GCR Code: 4te5zrg

EIE228 – IoT for Automation

For V Semester Minors in ROBOTICS & AUTOMATION

UNIT 2- Prototyping Devices and Protocols

Dr. RM. Kuppan Chetty.,

M.Tech (sensors)., Ph.D (IITM-Robotics), M.I.E., SMIEEE., MIET., C.Eng.,

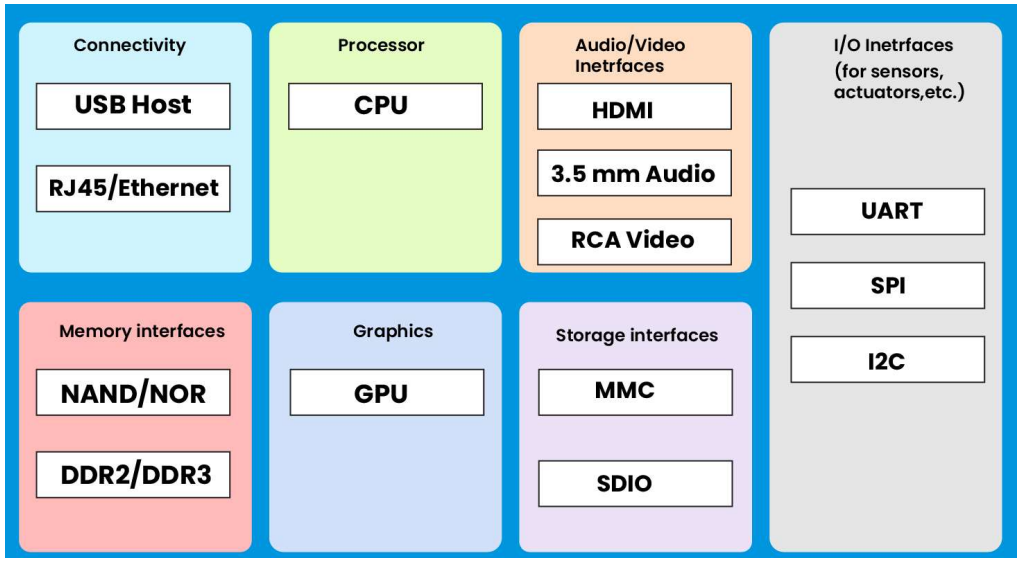
Professor; School of Mechanical Engineering

Physical Design of IoT

- The "Things" in IoT usually refers to IoT devices which have unique identities and can perform remote sensing, actuating and monitoring capabilities.
- Encompasses the individual nodes and the protocols used for creating an enterprise's IoT ecosystem

IoT devices can:

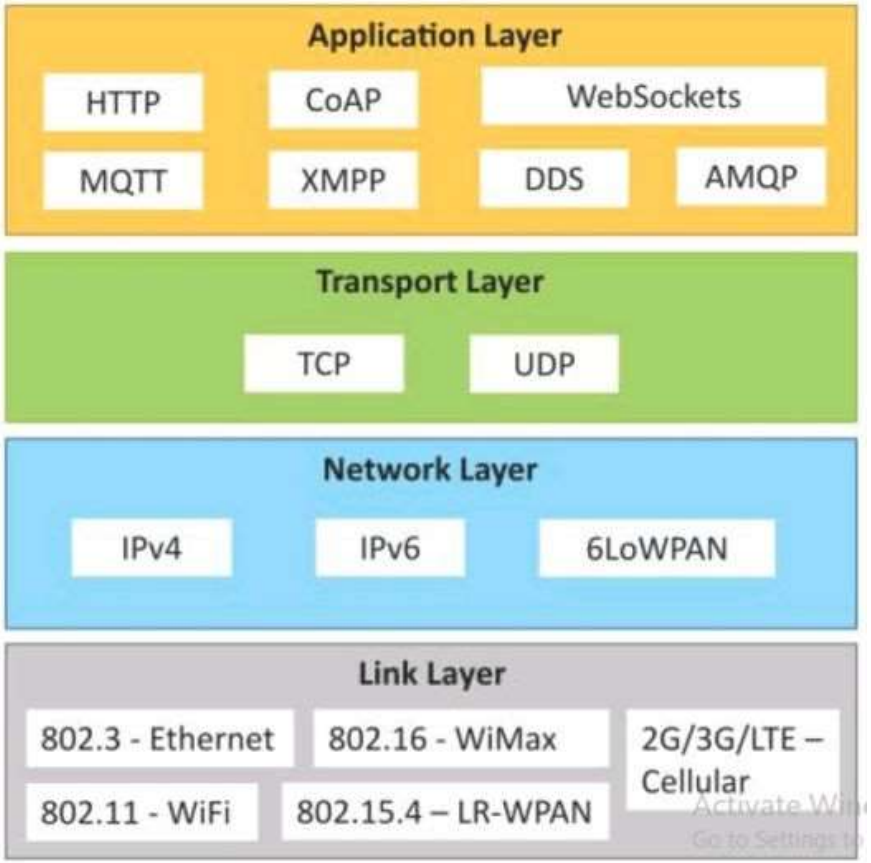
- Exchange data with other connected devices and applications (directly or indirectly), or
- Collect data from other devices and process the data locally or
- Send the data to centralized servers or cloud-based application back-ends for processing the data, or
- Perform some tasks locally and other tasks within the IoT infrastructure, based on temporal and space constraints



Generic block diagram of an IoT Device

IoT Protocols

- Link Layer
  - 802.3 – Ethernet ; 802.11 – WiFi ; 802.16 – WiMax
  - 802.15.4 – LR-WPAN
  - 2G/3G/4G
- Network/Internet Layer
  - IPv4, IPv6
  - 6LoWPAN
- Transport Layer
  - TCP
  - UDP
- Application Layer
  - HTTP
  - CoAP
  - WebSocket
  - MQTT
  - XMPP
  - DDS
  - AMQP



Logical Design of IoT

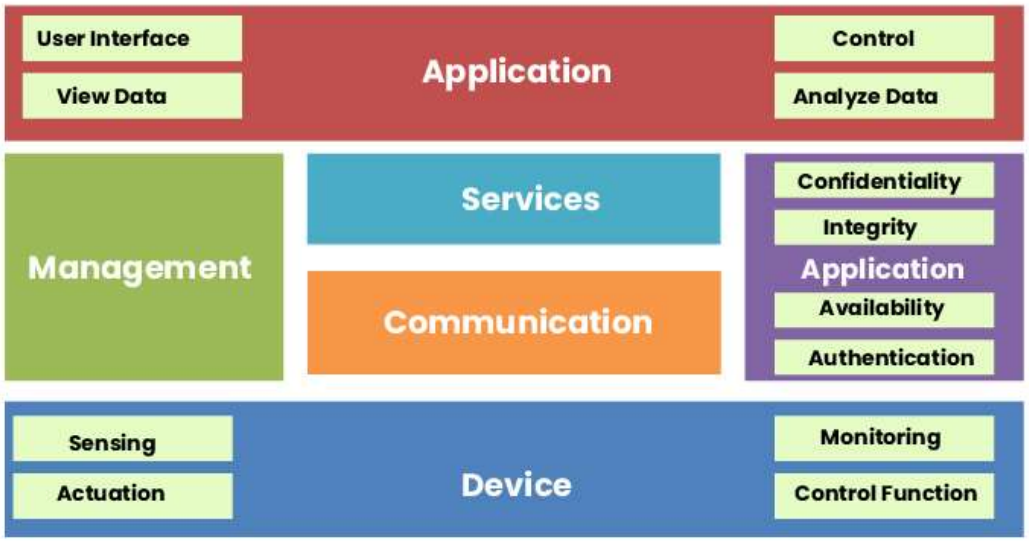
- It is the actual design of the IoT system. It illustrates the assembling and configuration of the components i.e. computers, sensors, and actuators.
- The logical design of IoT is composed of:
  - IoT functional blocks
  - IoT communications models
  - IoT communication APIs

IoT functional blocks

- Device, Communication, Services, Management, Security and Applications

IoT Communication Models

- Request Response Model, Publish Subscribe Model, Push Pull Model and Exclusive Pair Model



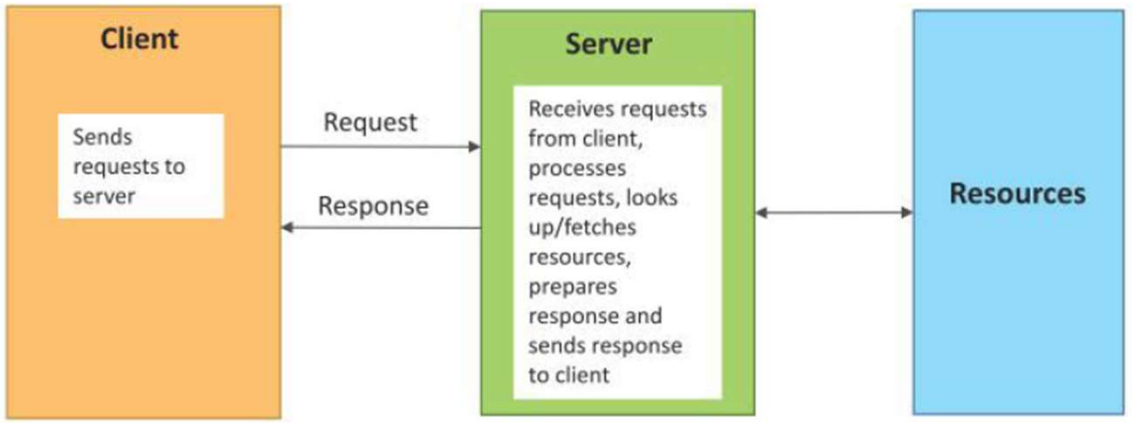
IoT Communication API

- REST based
- Web Socket

IoT Communication Model

Request-Response communication model

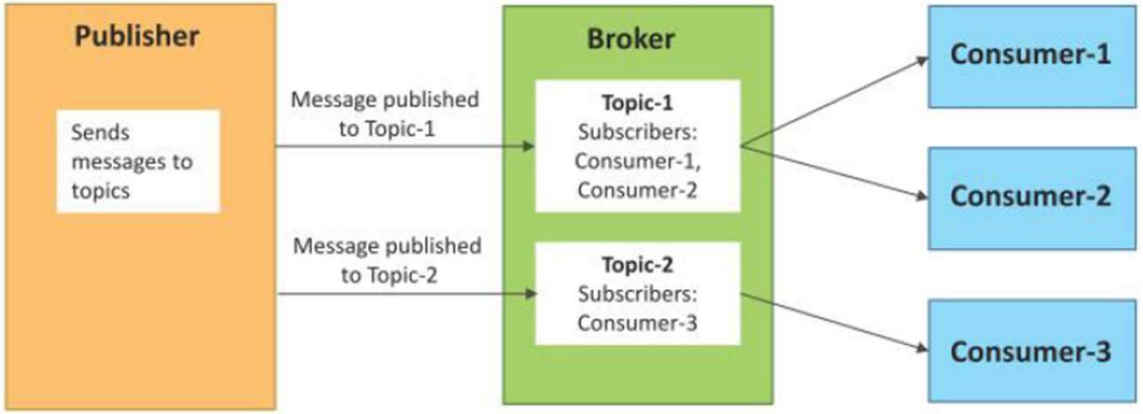
- Request-Response is a communication model in which the client sends requests to the server and the server responds to the requests.
- When the server receives a request, it decides how to respond, fetches the data, retrieves resource representations, prepares the response, and then sends the response to the client.



IoT Communication Model

Publish-Subscribe communication model

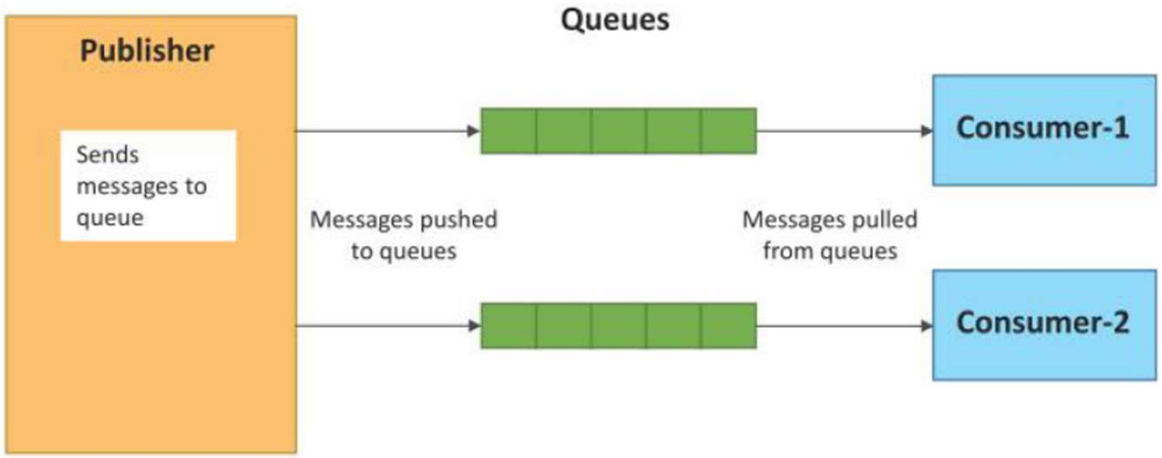
- Publish-Subscribe is a communication model that involves publishers, brokers and consumers.
- Publishers are the source of data. Publishers send the data to the topics which are managed by the broker. Publishers are not aware of the consumers.
- Consumers subscribe to the topics which are managed by the broker.
- When the broker receives data for a topic from the publisher, it sends the data to all the subscribed consumers.



IoT Communication Model

Push-Pull communication model

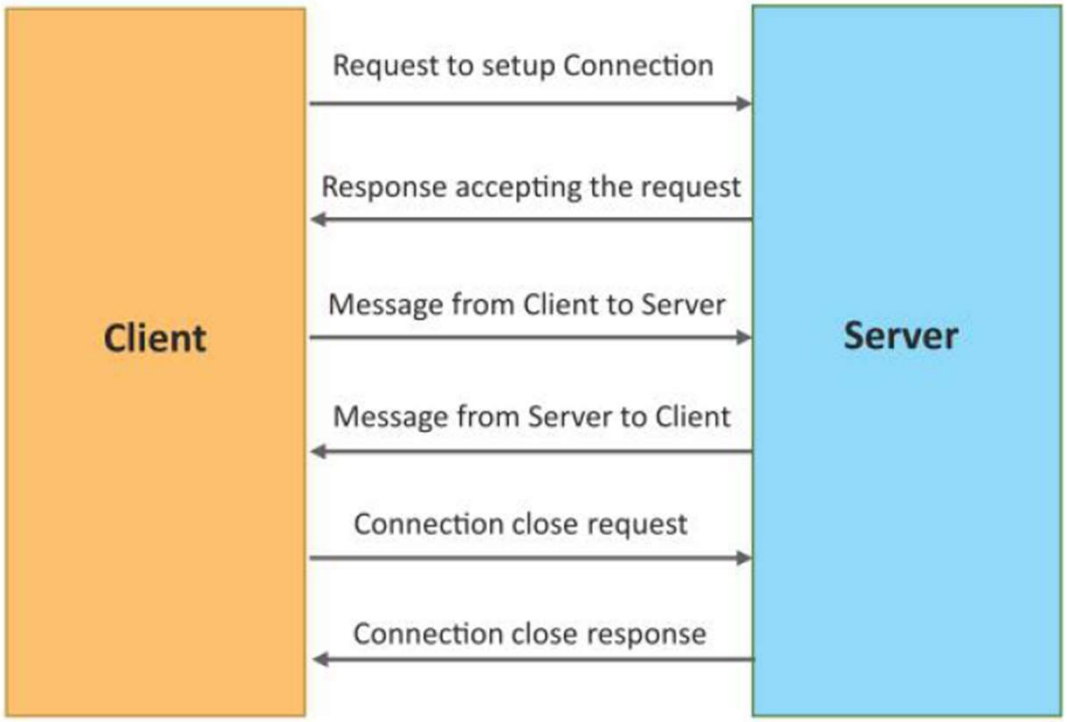
- Push-Pull is a communication model in which the data producers push the data to queues and the consumers pull the data from the queues. Producers do not need to be aware of the consumers.
- Queues help in decoupling the messaging between the producers and consumers.
- Queues also act as a buffer which helps in situations when there is a mismatch between the rate at which the producers push data and the rate rate at which the consumers pull data.



IoT Communication Model

Exclusive Pair communication model

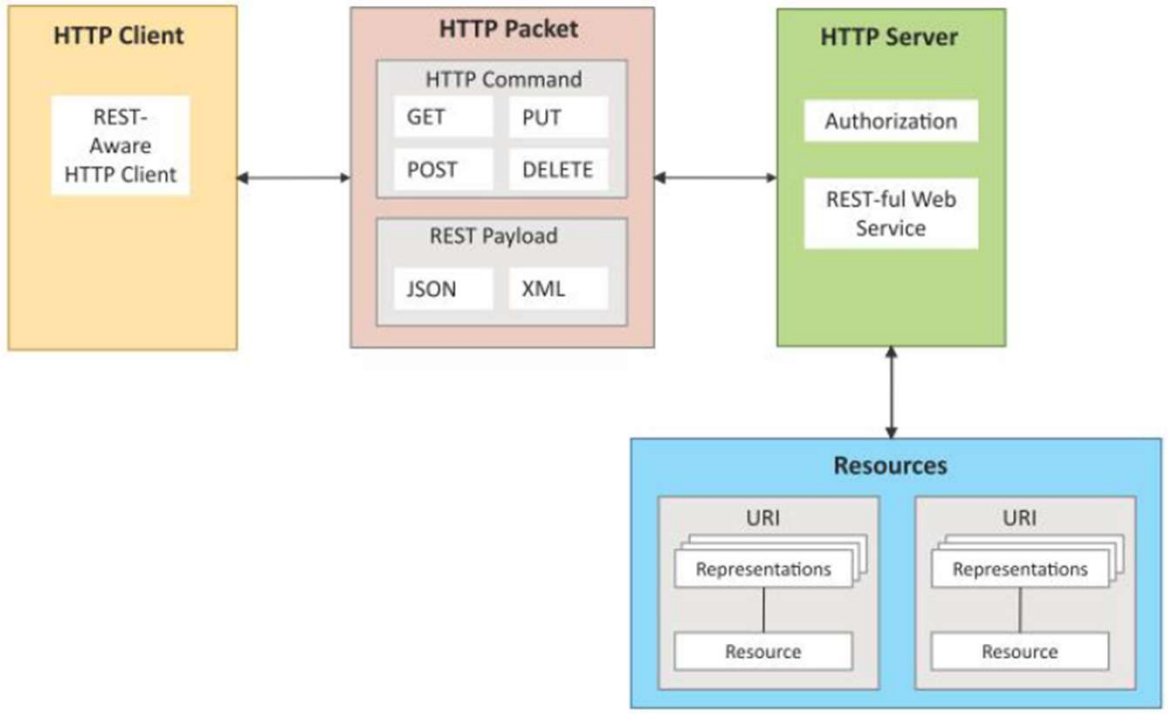
- Exclusive Pair is a bidirectional, fully duplex communication model that uses a persistent connection between the client and server.
- Once the connection is setup it remains open until the client sends a request to close the connection.
- Client and server can send messages to each other after connection setup.



IoT Communication APIs

REST-based Communication APIs

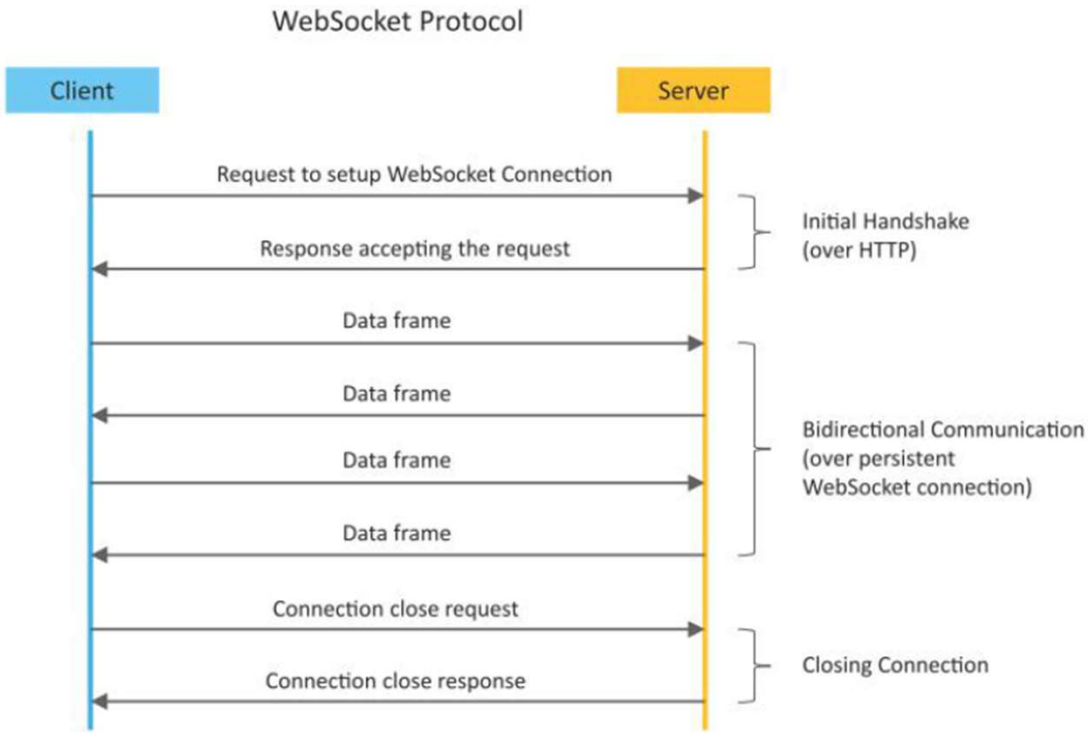
- Representational State Transfer (REST) is a set of architectural principles by which you can design web services and web APIs that focus on a system's resources and how resource states are addressed and transferred.
- REST APIs follow the request-response communication model.
- The REST architectural constraints apply to the components, connectors, and data elements, within a distributed hypermedia system.



IoT Communication APIs

WebSocket-based Communication APIs

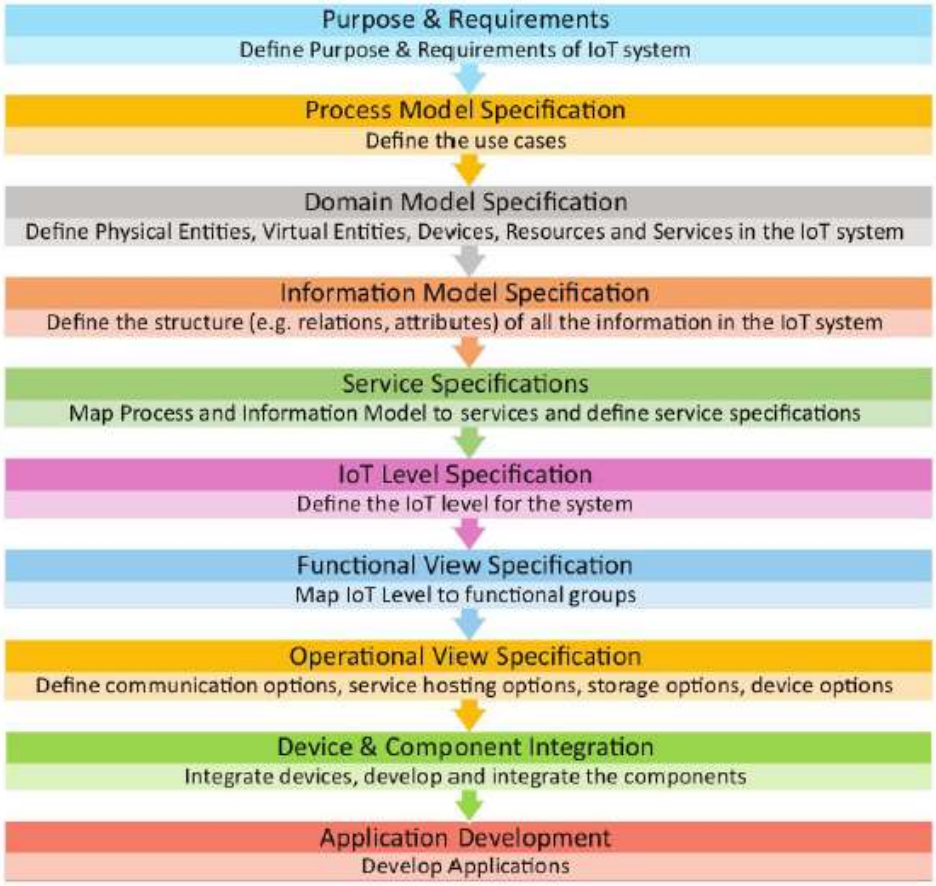
- WebSocket APIs allow bi-directional, full duplex communication between clients and servers.
- WebSocket APIs follow the exclusive pair communication model



IoT Design Methodology

IoT Design Methodology that includes:

- ❑ Purpose & Requirements Specification
- ❑ Process Specification
- ❑ Domain Model Specification
- ❑ Information Model Specification
- ❑ Service Specifications
- ❑ IoT Level Specification
- ❑ Functional View Specification
- ❑ Operational View Specification
- ❑ Device & Component Integration
- ❑ Application Development



Step 1: Purpose & Requirements Specificatio

The first step in IoT system design methodology is to define the purpose and requirements of the system.

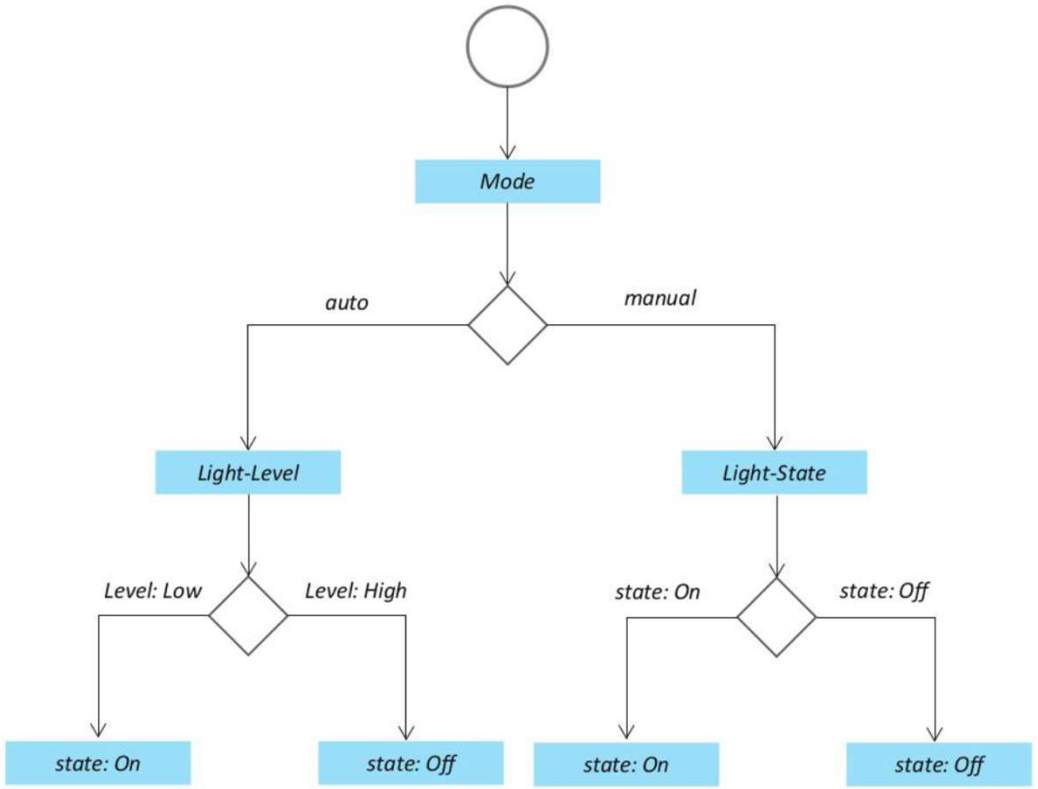
The system purpose, behavior and requirements (such as data collection requirements, data analysis requirements, system management requirements, data privacy and security requirements, user interface requirements, ...) are captured.

Example: Smart Home Automation

- **Purpose:** Controlling of lights in a home remotely through web applications
- Behavior : Auto and manual modes.
  - In auto mode - system measures the light level in the room and switches on the light when it gets dark.
  - In manual mode, the system provides the option of manually and remotely switching on/off the light.
- **System Management Requirement:** provide remote monitoring and control functions.
- **Data Analysis Requirement:** perform local analysis of the data.
- **Application Deployment Requirement:** application should be deployed locally on the device but should be accessible remotely.
- **Security Requirement:** The system should have basic user authentication capability.

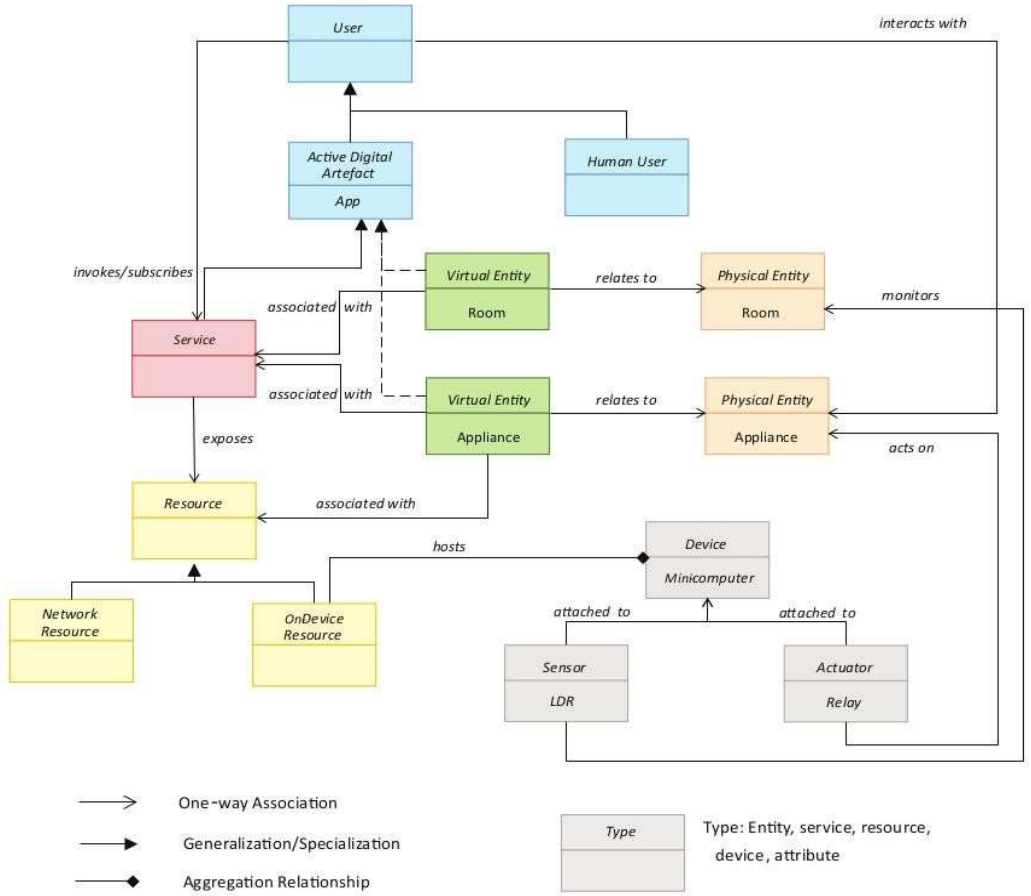
Step 2: Process Specification

- The second step in the IoT design methodology is to define the process specification.
- In this step, the use cases of the IoT system are formally described based on and derived from the purpose and requirement specifications.



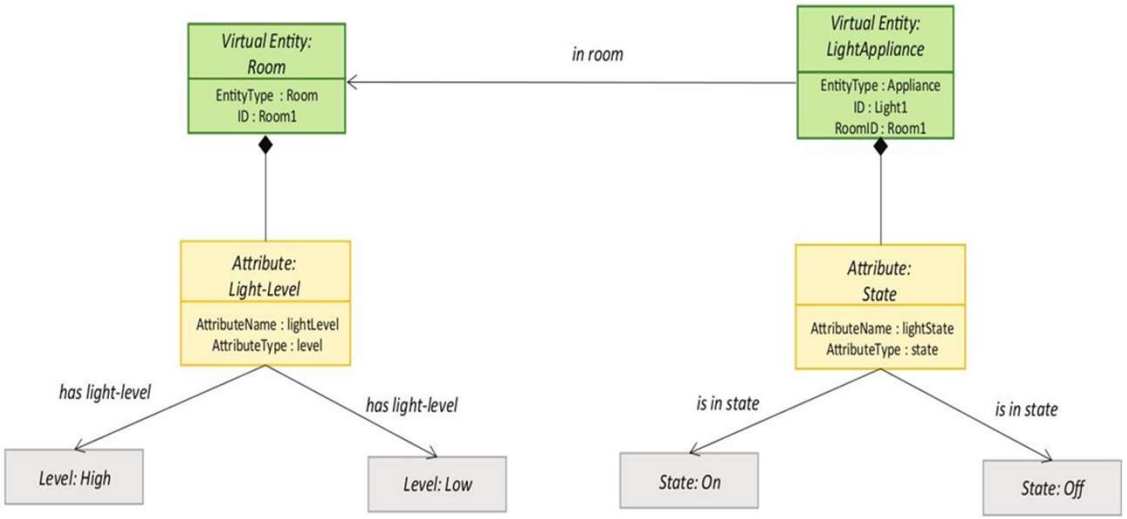
Step 3: Domain Model Specification

- Defines the Domain Model. It describes the main concepts, entities and objects in the domain of IoT system to be designed.
- Defines the attributes of the objects and relationships between objects.
- Provides an abstract representation of the concepts, objects and entities in the IoT domain, independent of any specific technology or platform.
- **Physical entity** (a room, a light, an appliance, a car etc.), **Virtual Entity** (representation of physical entity in digital world), **Device** (mini-computer – providing medium of interaction between physical and virtual entity), **Resource** (Software components – OS) and **Services** (interface for interactions)
- IoT system designers can get an understanding of the IoT domain for which the system is to be designed.



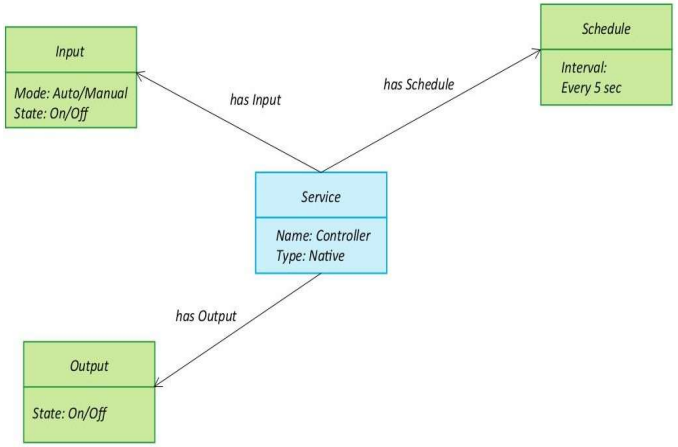
Step 4: Information Model Specification

- Defines the Information Model. The structure of all the information in the IoT system, for example, attributes of Virtual Entities, relations, etc.
- Information model does not describe the specifics of how the information is represented or stored.
- Listing of the Virtual Entities defined in the Domain Model is necessary to define the information model.
- Adds more details to the Virtual Entities by defining their attributes and relations.

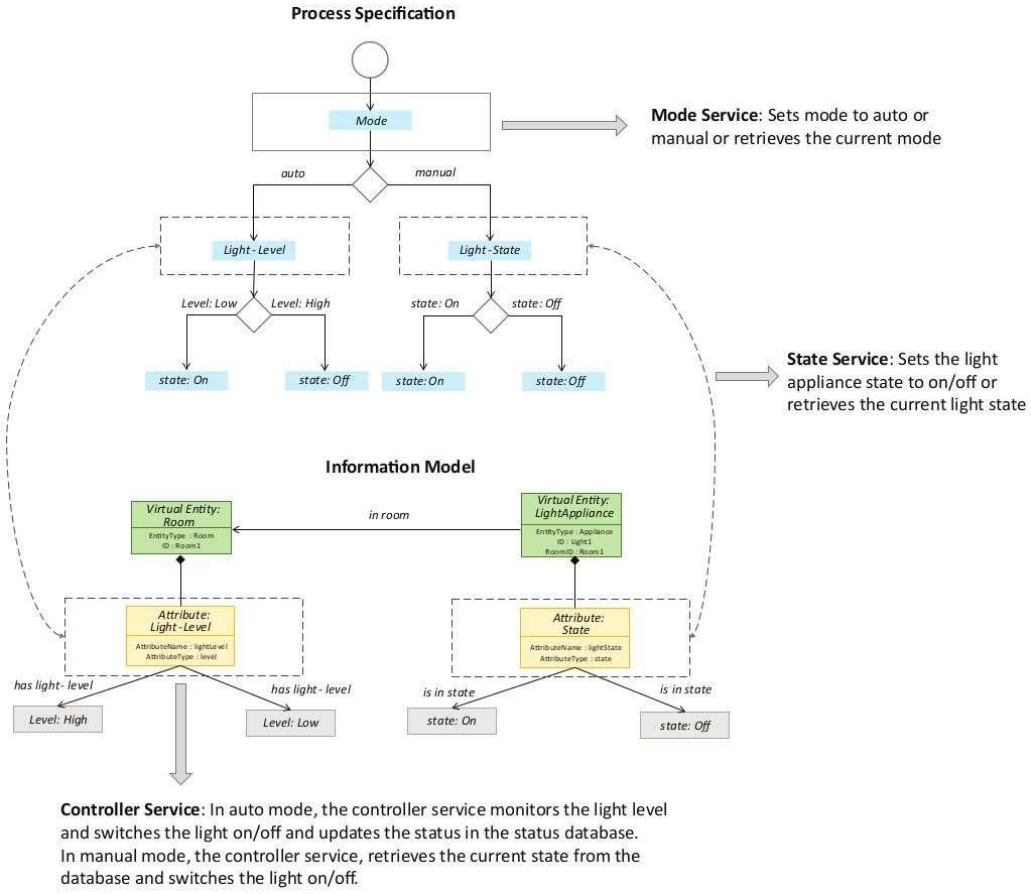


Step 5: Service Specifications

- The fifth step in the IoT design methodology is to define the service specifications.
- Service specifications define the services in the IoT system, service types, service inputs/output, service endpoints, service schedules, service preconditions and service effects.



Controller Service of home automation IoT system

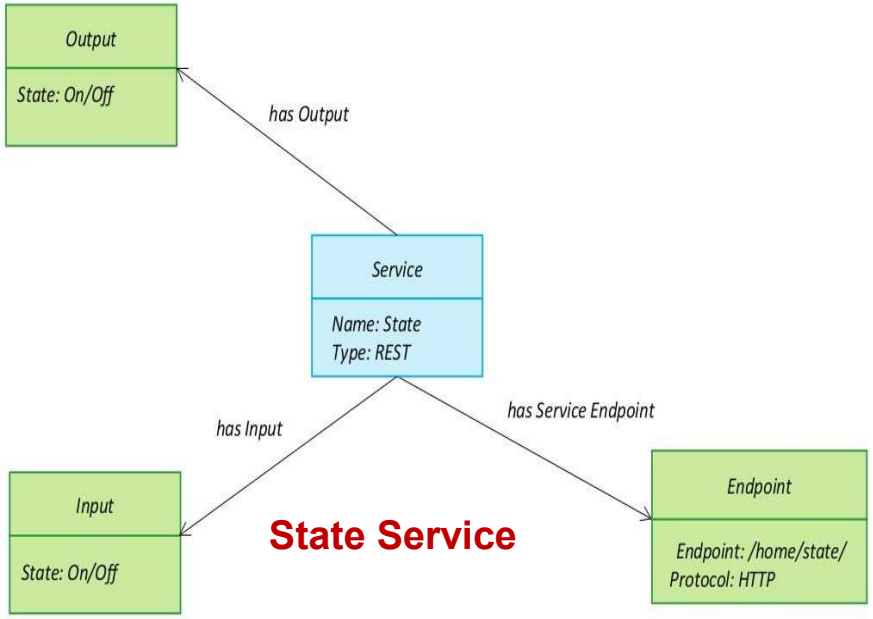


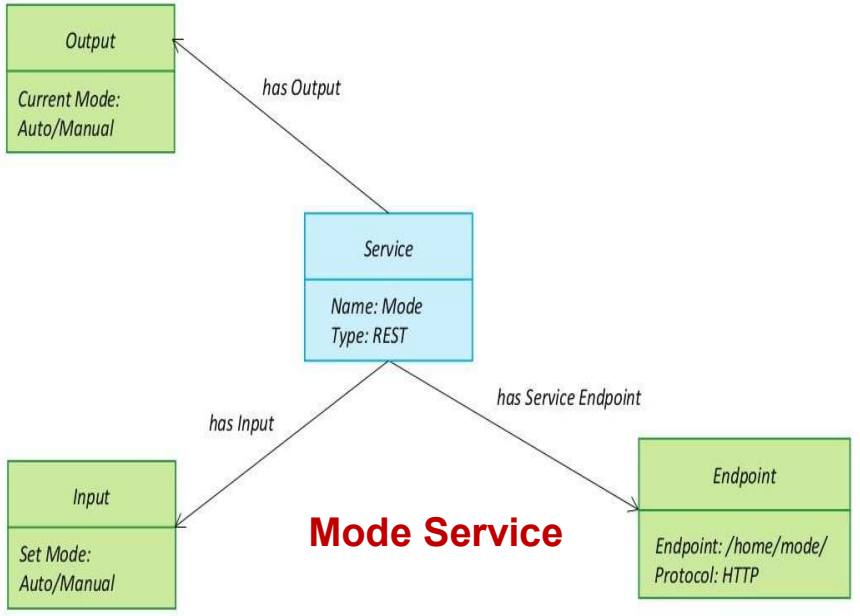
Mode Service: Sets mode to auto or manual or retrieves the current mode

State Service: Sets the light appliance state to on/off or retrieves the current light state

Controller Service: In auto mode, the controller service monitors the light level and switches the light on/off and updates the status in the status database. In manual mode, the controller service, retrieves the current state from the database and switches the light on/off.

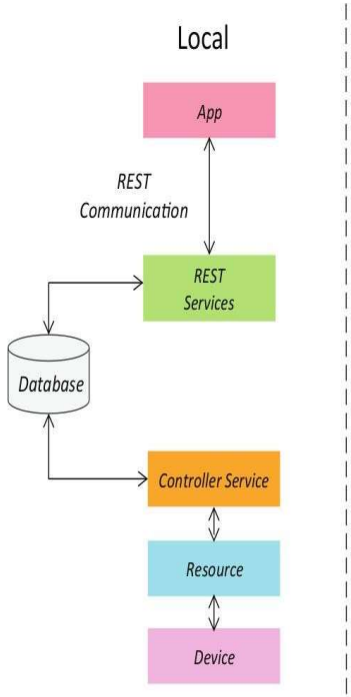
Step 5: Service Specifications





Step 6: IoT level Specification

- The sixth step in the IoT design methodology is to define the IoT level for the system.



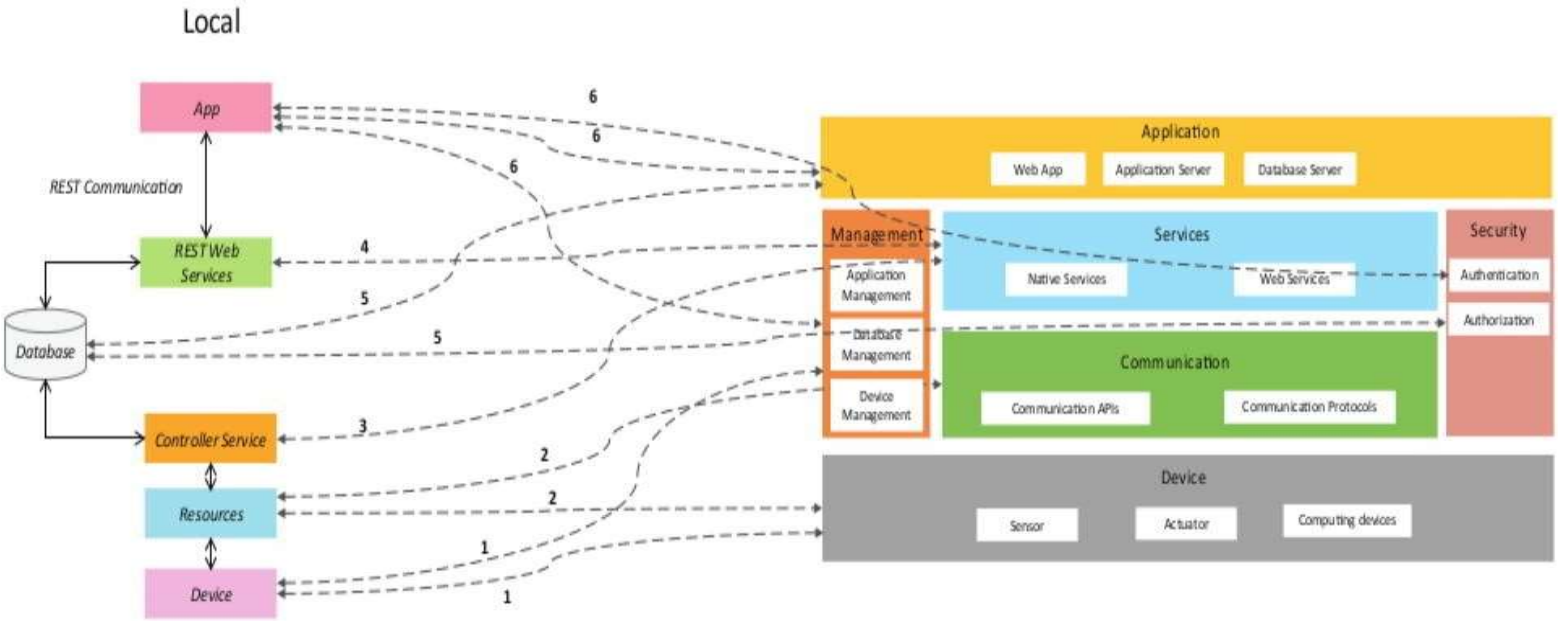
Deployment design of the home automation IoT system

Monitoring Node
performs analysis, stores data

Step 7: Functional View Specification

- Functional View Definition (FV).
- Defines the functions of the IoT systems grouped into various Functional Groups (FGs).
- Each Functional Group either provides functionalities for interacting with instances of concepts defined in the Domain Model or provides information related to these concepts.
- FG included in a FV are
  - Device
  - Communication
  - Services
  - Management
  - Security
  - Applications

Step 7: Functional View Specification



1. IoT device maps to the Device FG (sensors, actuators devices, computing devices) and the Management FG (device management)

2. Resources map to the Device FG (on-device resource) and Communication FG (communication APIs and protocols)

3. Controller service maps to the Services FG (native service). Web Services map to Services FG (web services)

4. Web Services map to Services FG (web services)

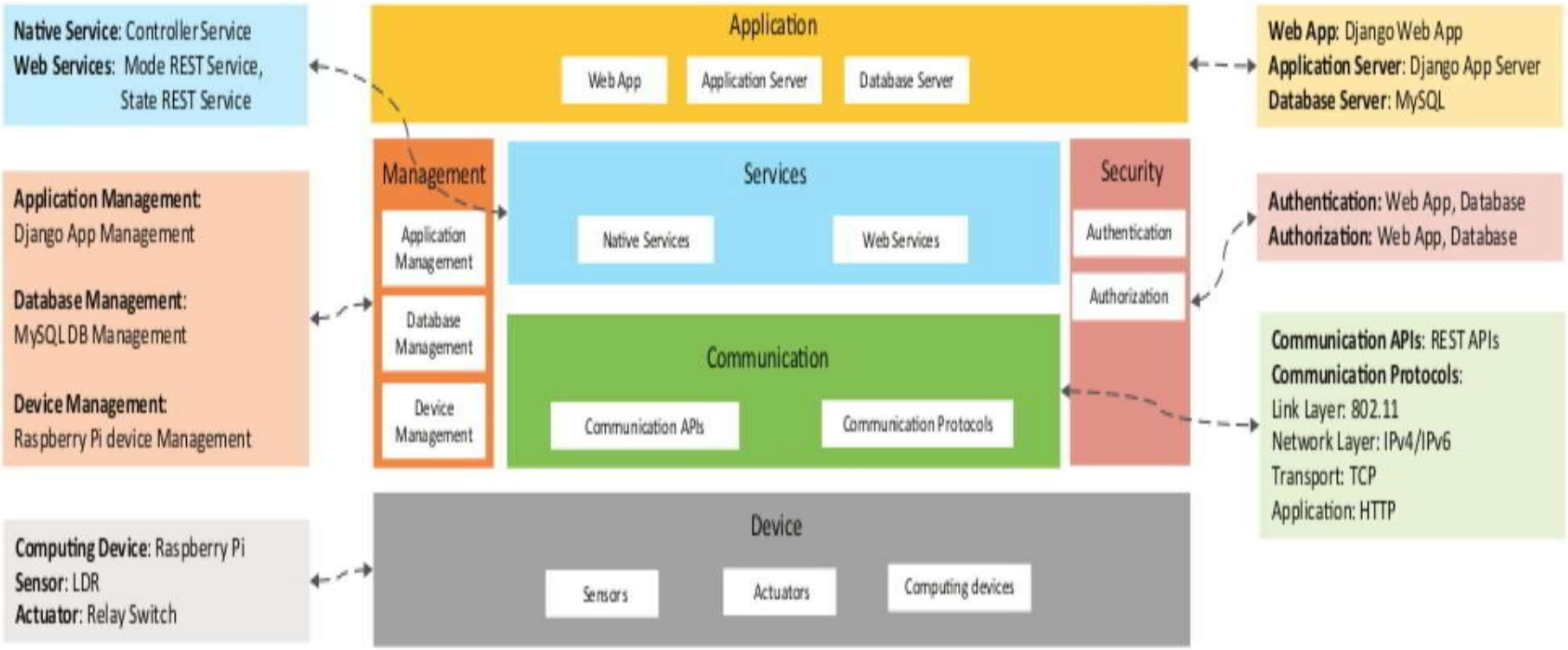
5. Database maps to the Management FG (database management) and Security FG (database security)

6. Application maps to the Application FG (web application, application and database servers), Management FG (app management) and Security FG (app security)

Step 8: Operational View Specification

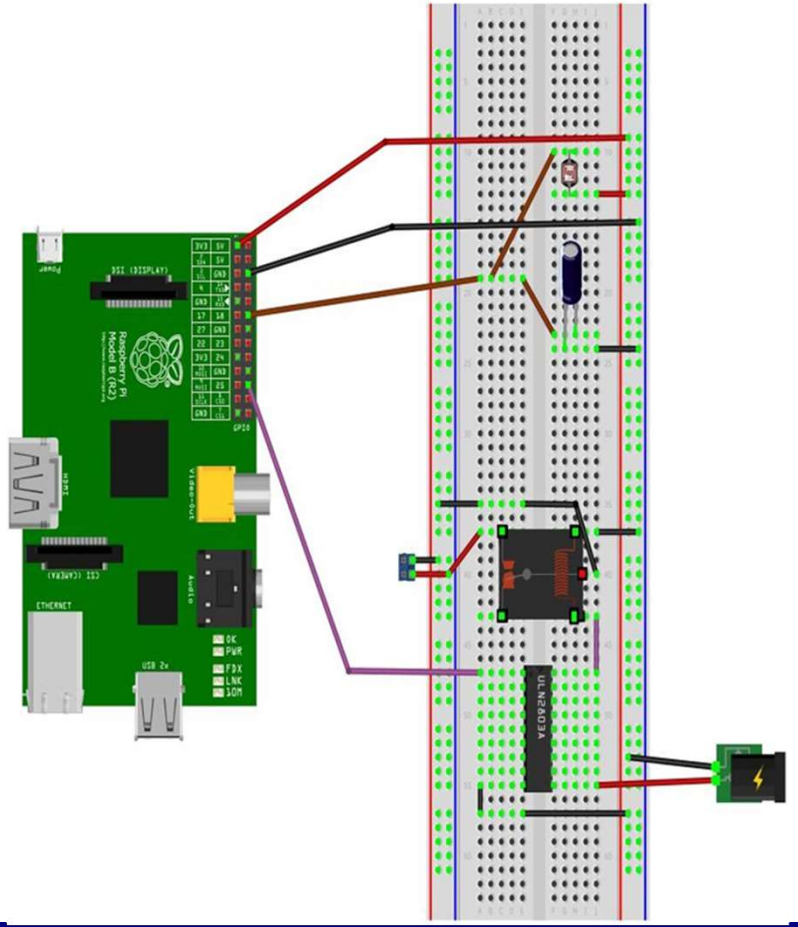
- The eighth step in the IoT design methodology is to define the Operational View Specifications.
- Various options pertaining to the IoT system deployment and operation are defined, such as, service hosting options, storage options, device options, application hosting options, etc
- Devices: Controller (Arduino, Pi. Etc.), sensors and actuators
- Communication API: REST API
- Protocol: Link Layer – 802.11, Network: IPv4/IPv6, Transport – TCP, Application: HTTP
- Services: all 3 services (Cont., Mode., State.)
- Application: Web based
- Security: Authentication and authorization
- Management: Application, Data Base and Device Management

Step 8: Operational View Specification



Step 9: Device & Component Integration

- The ninth step in the IoT design methodology is the integration of the devices and components.

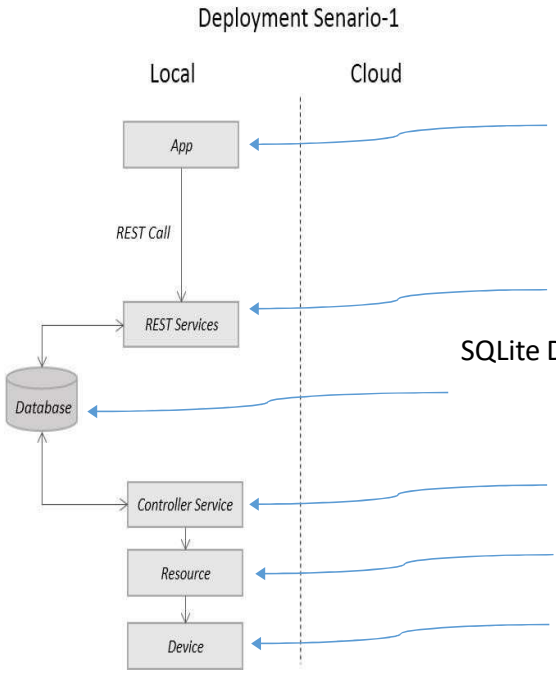


Step 10: Application Development

- The final step in the IoT design methodology is to develop the IoT application.
  - Auto
    - Controls the light appliance automatically based on the lighting conditions in the room
  - Light
    - When Auto mode is off, it is used for manually controlling the light appliance.
    - When Auto mode is on, it reflects the current state of the light appliance.

Finally - Integrate the System

- Setup the device
- Deploy and run the REST and Native services
- Deploy and run the Application
- Setup the database



Django Application

REST services implemented with Django-REST framework

SQLite Database

Native service implemented in

Python OS unning on

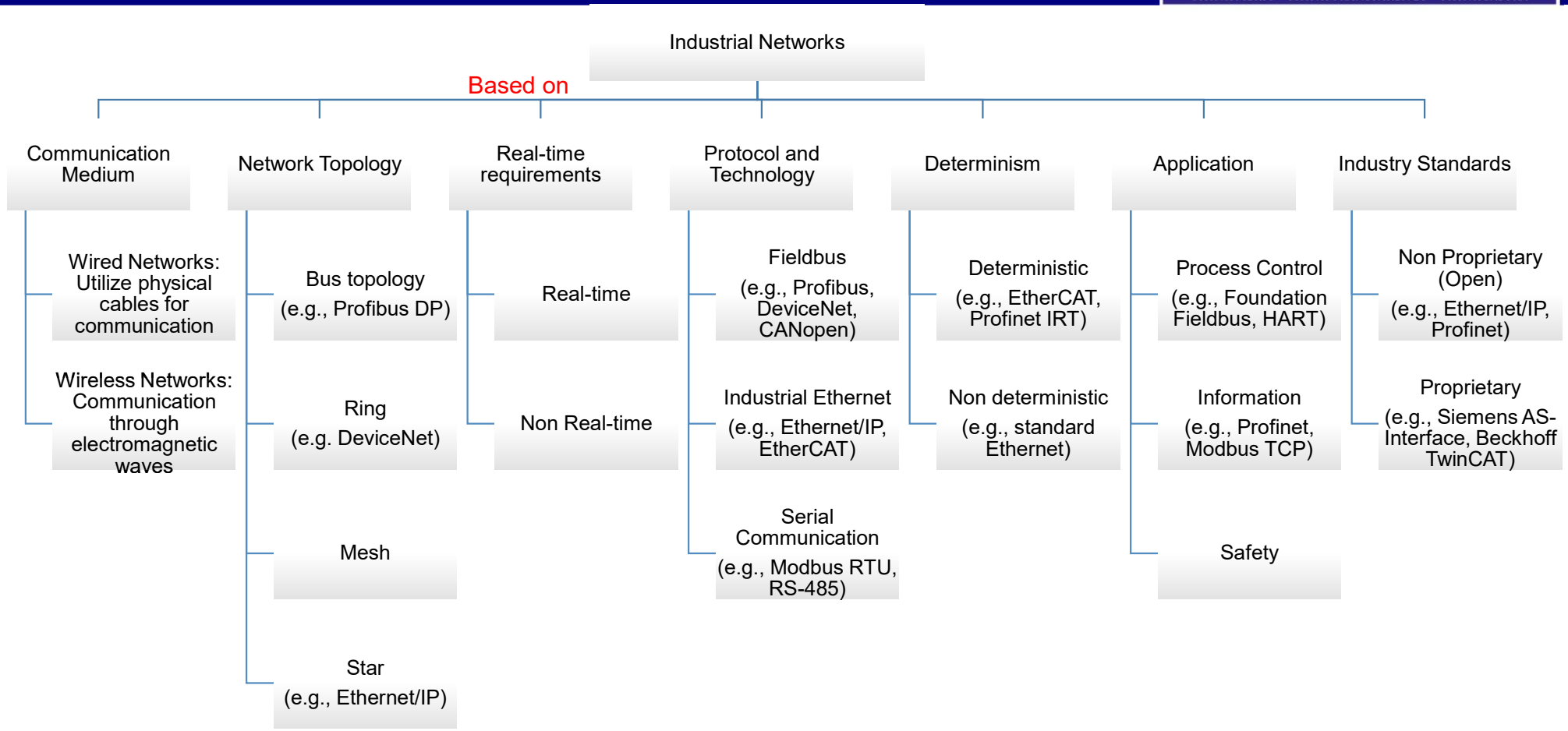
RaspberryPi

Raspberry Pi device to which sensors and actuators are connected

Industrial Network

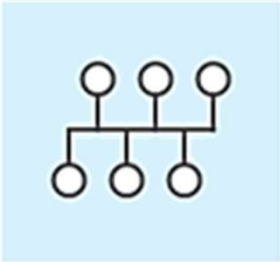
- Industrial networks are *communication systems* specifically designed for use in industrial environments to connect various devices and equipment, such as Programmable Logic Controllers (PLCs), Human-Machine Interfaces (HMIs), sensors, actuators, and other automation components
- These networks enable *data exchange* and *control signals* for automation processes.
- These are *essential for coordinating machinery* and *processes in manufacturing* and other industries.
- They are designed to meet industrial requirements such as *reliability, determinism,* and *real-time performance*.
- *Determinism* is a property of a network which ensures *consistent response times* by executing processes *within predictable time frames reliably*.
- Industrial networks are *robust* against *electromagnetic interference* and *harsh conditions*.
- *Communication protocols* like ***Ethernet/IP, Profibus, Modbus,*** and others are commonly used in industrial networks.
- These protocols define *rules for data transmission, addressing, error detection, and synchronization*.

Industrial Networks - Classification

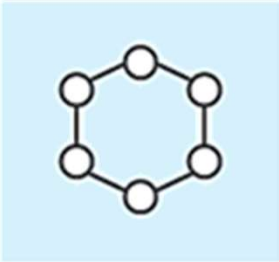


Industrial Networks – Classification

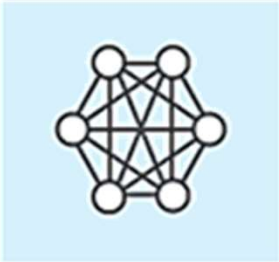
Based on Network Topology



Bus
Directly connects devices to each other and transmits data between links.

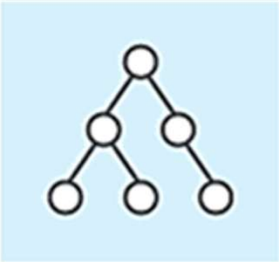


Ring
Connects devices next to each other in the form of a circle. Communication occurs unidirectionally or bidirectionally.



Mesh
Connects each device to every other device in the network.

Star
Features a central device which transmits data to other nodes in the system.



Tree
Connects devices down in a structure resembling a tree where parent nodes connect to child nodes.

Hybrid
Consists of at least two different types of network topology.

Industrial Networks – Classification

Based on Bandwidth and Range

	SIZE	DEVICES	CONNECTIVITY
PAN Devices connected around an individual	Small area of coverage; usually around 10 meters	IoT devices, tablets, wearables	Bluetooth, Zigbee, USB
LAN Devices connected to each other in one physical location	Up to hundreds of meters	Computers, smartphones, access points, routers, switches	Fiber optics, Ethernet, wireless, cellular
MAN Several LANs connected in a municipality	Up to 50 km	Computers, smartphones, access points, routers, switches	Fiber optics, Ethernet, wireless, cellular
Campus Several LANs connected in a general area	Between 1 km and 5 km	Computers, smartphones, access points, routers, switches	Fiber optics, Ethernet, wireless, cellular
WAN Connection of LANs linked around the world	Global	Computers, gateways, routers, switches	Fiber optics, Ethernet, wireless, cellular
CDN Connection of servers linked around the world to distribute rich media content	Global	Proxy servers, origin servers	Points of presence
VPN Virtual private network overlay on an existing network	Global	Computers, smartphones, tablets	Virtual connections with tunneling protocols

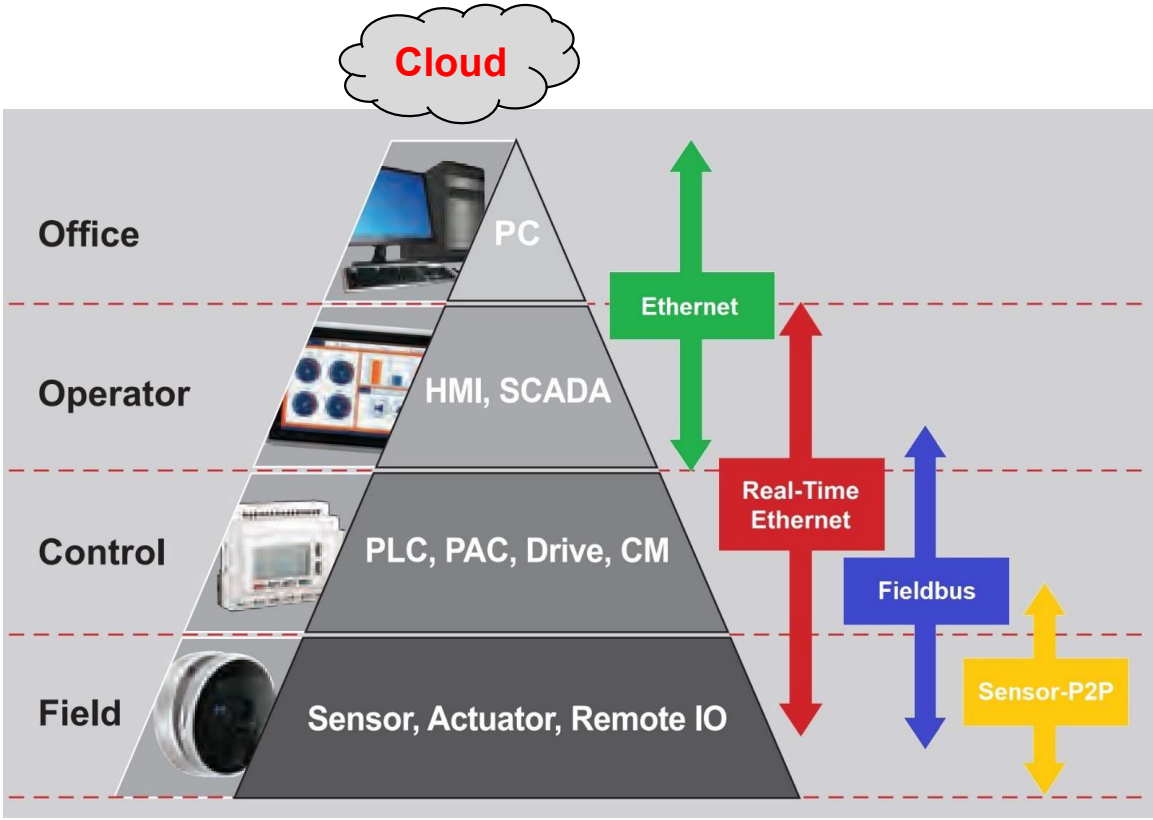
Industrial Networks – Classification

Based on Communication Protocol Technology

Sensor Networks	Fieldbus	Industrial Ethernet	Wireless
❖ An industrial networking solution used in PLC, DCS, and PC-based automation. ❖ An open technology supported by a multitude of automation equipment vendor ❖ ASi-3 ❖ ASi-5 ❖ IO-link	❖ Open system protocol that can run on a variety of physical layers, widely used in industrial control applications. ❖ Type of serial communication, most commonly used with RS232 and RS485 ❖ CANopen ❖ Profibus ❖ Modbus-RTU/ASCII ❖ DeviceNet	❖ Use of Ethernet in an industrial environment with protocols that provide determinism and real-time control. ❖ Etherenet/IP ❖ Profinet ❖ Modbus-TCP ❖ EtherCAT ❖ PowerLink ❖ OPCUA ❖ CC-Link IE	❖ Bluetooth ❖ WLAN

Industrial Network - Technologies

- Industry was once dominated by *serial networks* that utilized the technology that served the contemporary needs of the marketplace.
- Higher speeds and greater throughput are now needed to facilitate new, more sophisticated slave devices, end-node equipment, together with new applications.
- Thus, industrial communications equipment has been transitioning to faster, *deterministic Ethernet-based technology* and communications protocols, yet no specific communications protocol has come to dominate the industry.



Industrial Network - Technologies

Traffic Characteristics

- The traffic characteristics of IoT endpoints vary widely depending on the **application's demands and the nature of the devices.**
- Some applications compromise on packet loss, latency, and jitter (e.g., a meteorological monitoring application),
- Others have tight availability and latency(e.g., a jet engine control application).

Determinism

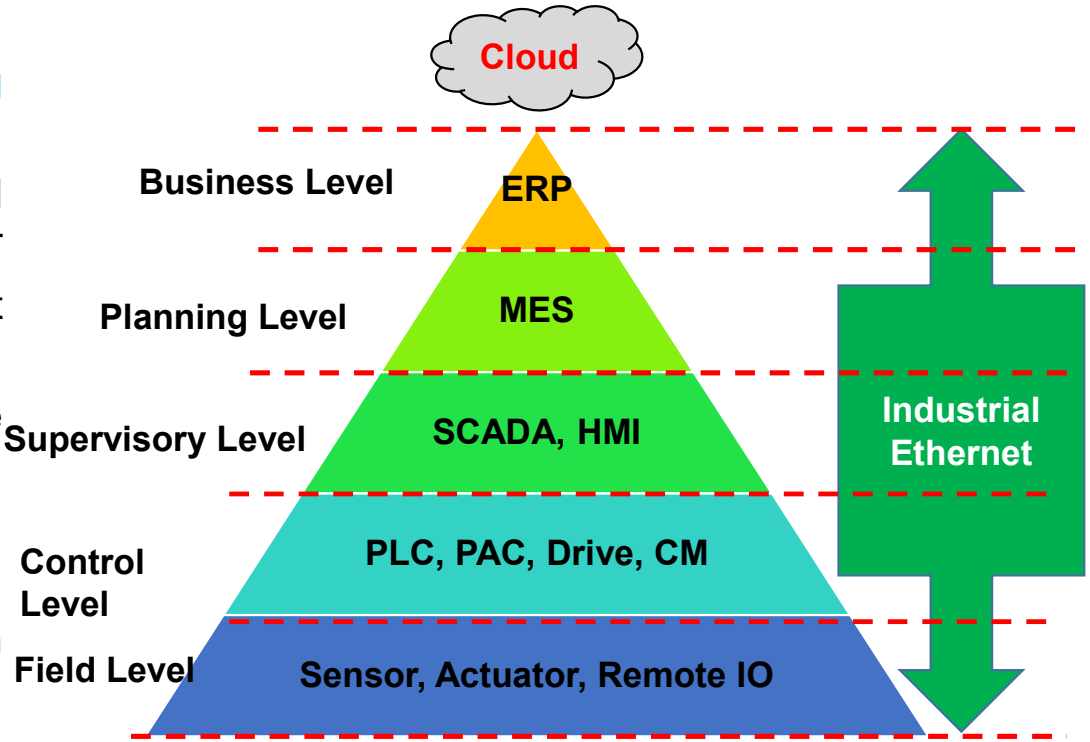
- All real-time use cases and applications share a common requirement to support real-time transfer: *the time taken for each packet to traverse a path from its source to its destination should be determined; that is, the process must be deterministic*.
- A network is said to support determinism if the worst-case communication latency and jitter of messages are realistic

Challenge for Determinism

- Migration of special-purpose non-packet-based technologies (e.g., HDMI, CAN bus, Profibus, etc.) to IP technologies to support new applications and also, existing IP network applications over the same physical network.

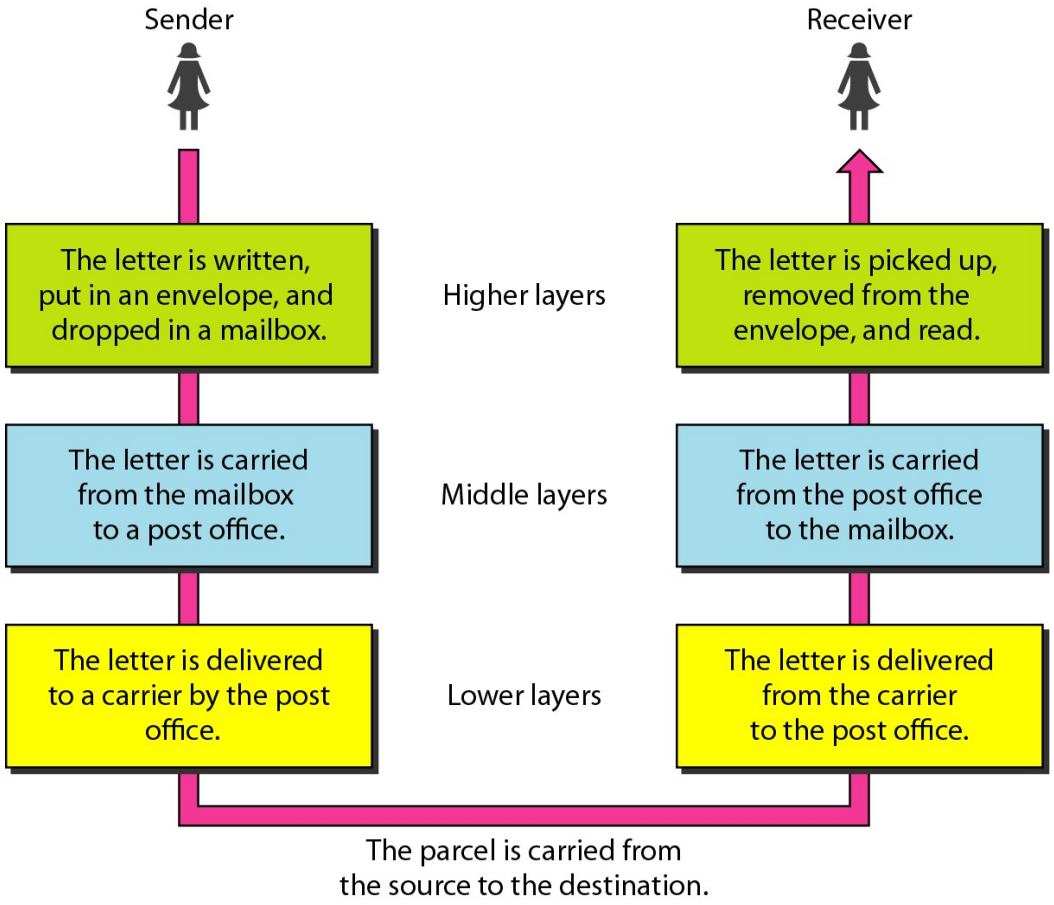
Industrial Network - Technologies

- *Ethernet is cost effective* and *ubiquitous*, offering common physical links with increased speed.
- Thus, many industrial communication systems are *moving to Ethernet based solutions*.
- In 2017, *Industrial Ethernet* dominated the market, representing **53% of industrial networking solutions**, with a notable growth rate of 22%. Leading Industrial Ethernet protocols included EtherNet/IP (15%), PROFINET (12%), EtherCAT (7%), Modbus-TCP, and POWERLINK (each at 4%).
- *Fieldbus technologies* accounted for 42% of the market, with PROFIBUS DP and Modbus-RTU each at 12%, DeviceNet and CANopen at 4%, and CC-Link at 6%.
- *Wireless technologies* experienced a 32% growth rate, making up 6% of the market, with WLAN protocols at 4% and Bluetooth at 1%.



Layered Tasks in Communication

- We use the *concept of layers* in our daily life.
- As an example, let us consider two friends who communicate through *postal mail*.
- The process of sending a letter to a friend would be complex if there were no services available from the post office.



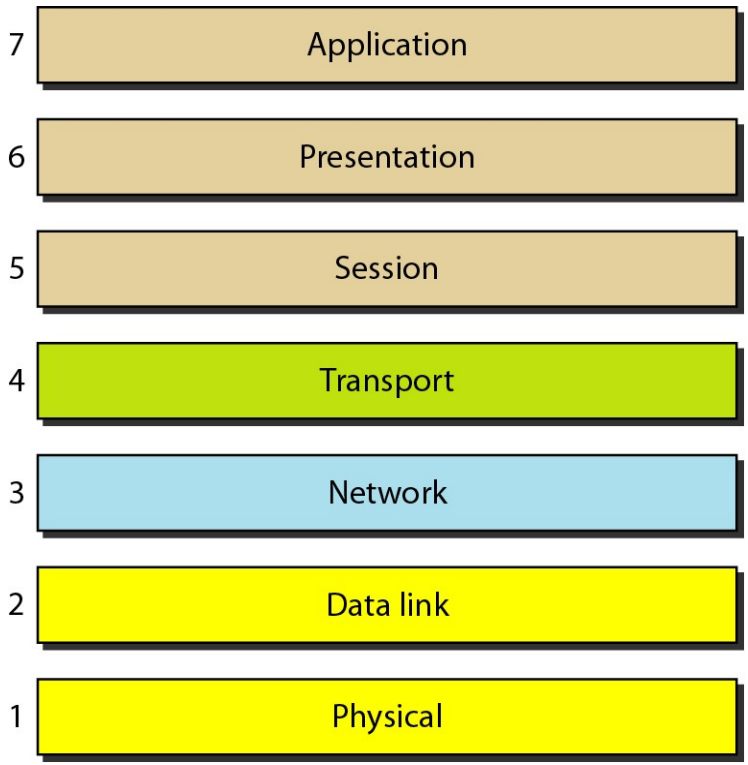
Open Systems Interconnection Model

- OSI model is based on the proposal developed by the *International Standards Organization (ISO)*.
- This model is called *ISO OSI (Open Systems Interconnection) Reference model* because it deals with *connecting open systems* (systems that are open for communication with other systems)

Principle on which OSI model designed

- A *layer* should be *created* where *different level of abstraction is needed*.
- *Each layer* should perform a well *defined function*.
- The *function of each layer* should be chosen according to *the internationally standardized protocols.*
- The *number of layers* should *be large enough* that *distinct functions should not be put in the same layer* and small enough that the architecture does not become very complex.

Open Systems Interconnection (OSI) Model



Human-computer interaction layer, where applications can access the network services

Ensures that data is in a usable format and is where data encryption occurs

Maintains connections and is responsible for controlling ports and sessions

Transmits data using transmission protocols including TCP and UDP

Decides which physical path the data will take

Defines the format of data on the network

Transmits raw bit stream over the physical medium

Seven layers of the OSI model

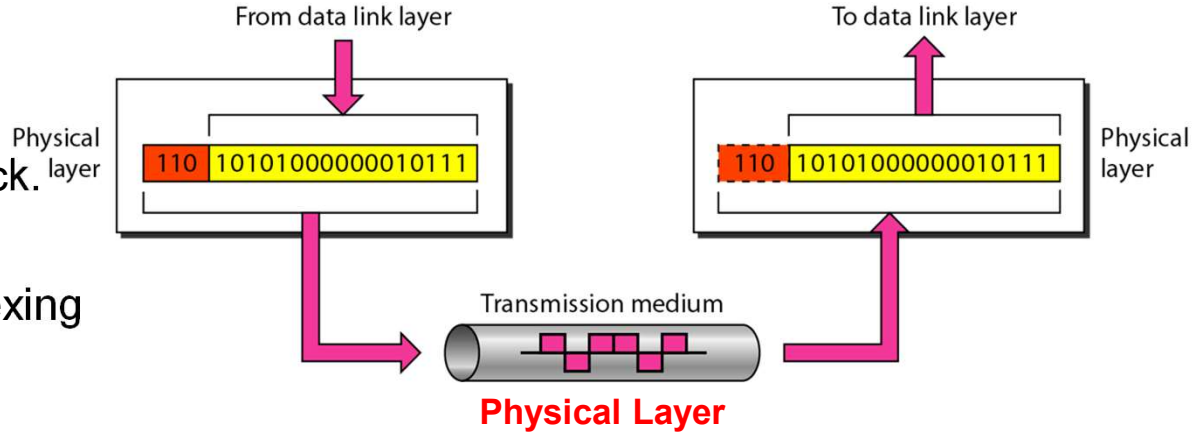
Open Systems Interconnection (OSI) Model

Physical Layer and Data Link Layer (DLL)

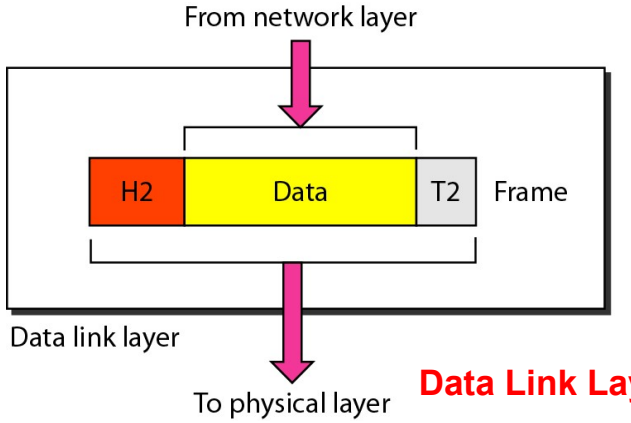
- **Physical layer** is the bottom layer of OSI Model.
- It is responsible for the *actual physical connection* between the devices.

Functions of the physical layer:

- Transforming bits into signals
- Provides synchronization of bits by a clock.
- It defines the transmission rate.
- It can use different techniques of multiplexing



Physical Layer

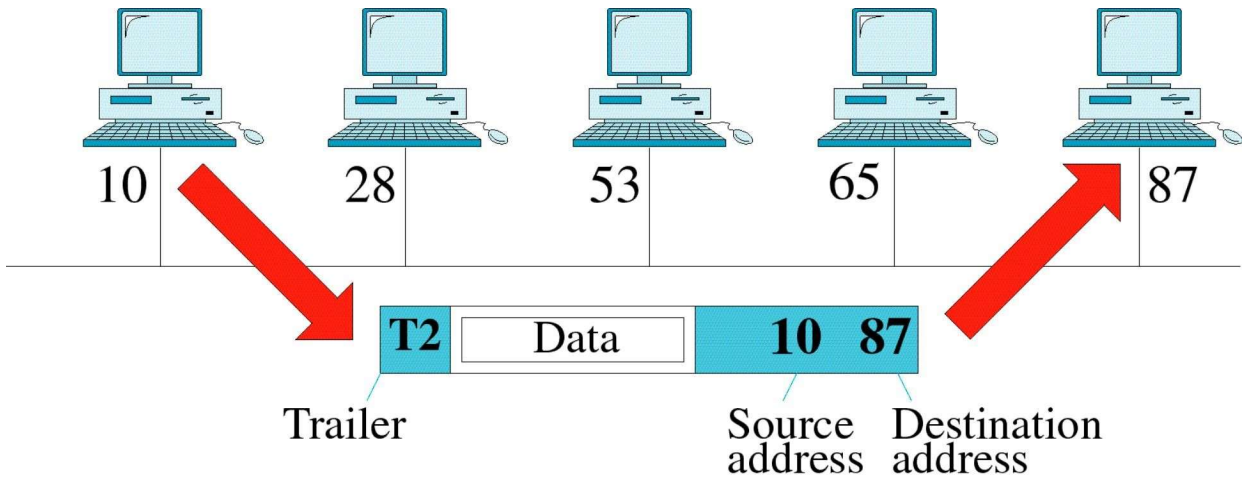


Data Link Layer

- **Data Link Layer (DLL)** receives the data from the network layer, adds the physical address & passes them to the physical layer

Functions:

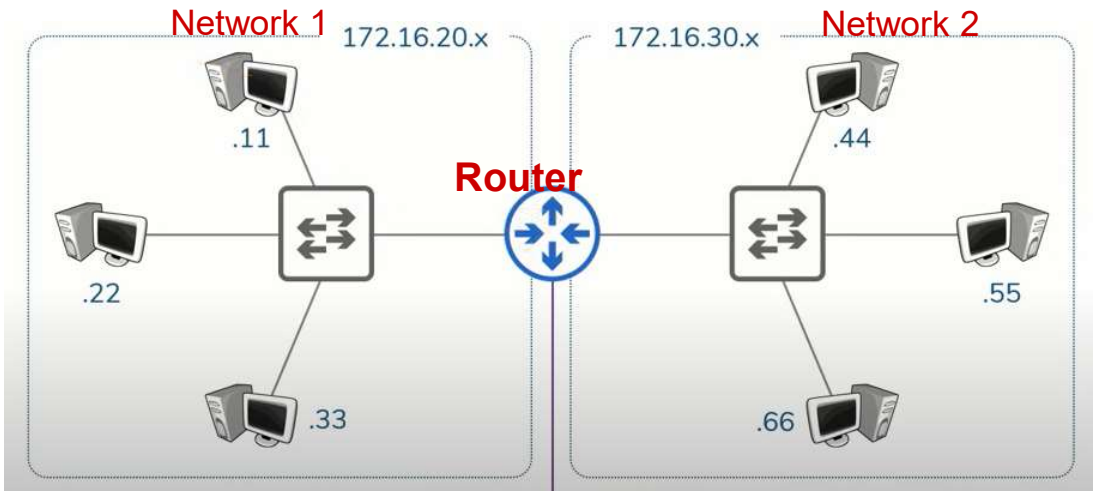
- **Framing:** DLL divides the bits received from the N/W layer into frames. (Frame contains all the addressing information necessary to travel from Source to Destination).
- **Physical addressing:** After creating frames, DLL adds the physical address of the sender/receiver (MAC address) in the header of each frame.

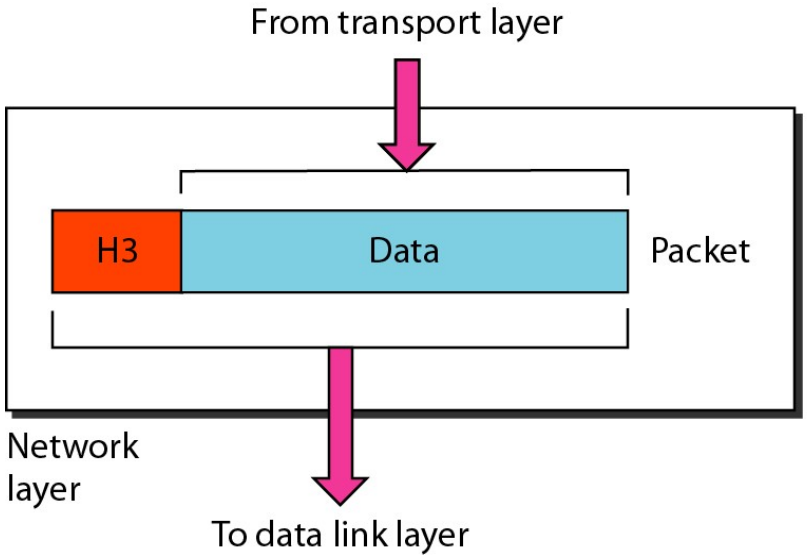


Open Systems Interconnection (OSI) Model

Network Layer

- It is responsible for the *source-to-destination delivery of a packet* across multiple networks.
- If two systems are attached to different networks with devices like *routers*, then N/W layer is used.





- **Functions:**
- **Routing:** routers & gateways route the packet to the final destination. The network layer ensures that the packe gets its point of origin to its final destination.
- **Logical Addressing**: N/W layer adds the Logical (network) address of the sender/receiver to each packet.

Open Systems Interconnection (OSI) Model

Transport Layer and Session Layer

Transport Layer provides two types of services:

- **Connection-Oriented Transmission:** The receiving devices send an acknowledgment back to the source after a packet is received. (Example: **TCP**)
- **Connectionless Transmission:** In this type of transmission the receiving devices do not send an acknowledgment back to the source. It is a faster transmission method. (Example: **UDP**)

Functions:

- **Segmentation of message into packet & reassembly of packets into message.**
- **Port addressing:** Computers run several processes. The transport layer header includes a port address with each process.

❖**Session Layer** has the responsibility of *beginning, maintaining, and ending the communication* between two devices, called session.

❖It also provides for *orderly communication between devices by regulating the flow of data*.

Functions:

- **Dialog Control:** This function determines which device will communicate first & the amount of data to be sent.
- **Dialog separation:** The process of adding checkpoints/markers to the stream of data

Transport Layer protocols

Transmission Control Protocol(TCP)

- This *connection-oriented protocol* requires a session to get established between the source and destination before exchanging data.
- *You can view it as an equivalent to a traditional telephone conversation, in which two phones must be connected and the communication link established before the parties can talk.*
- ability to transport large volumes of data into smaller sets of packets.
- ensures reassembly in a correct sequence, flow control and retransmission of lost packets.
- These benefits occur with the cost of overhead per packet and per session, potentially impacting overall packet per second performances and latency.

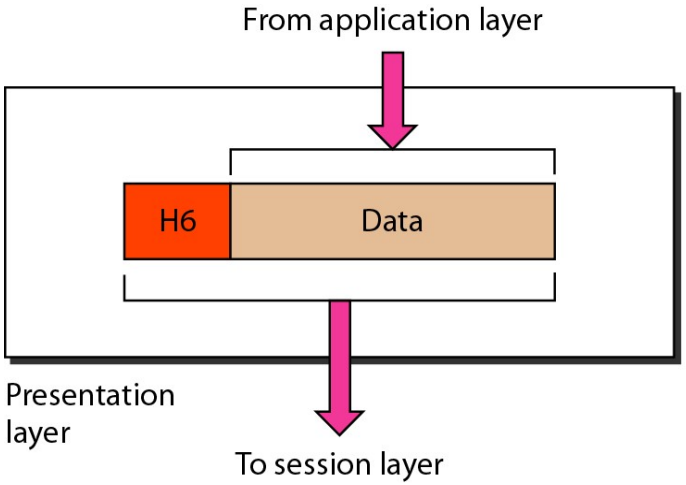
User Datagram Protocol (UDP)

- With this *connectionless protocol*, data can be quickly sent between source and destination—but with no guarantee of delivery.
- *This is analogous to the traditional mail delivery system, in which a letter is mailed to a destination.*
- *Confirmation of the reception of this letter does not happen until another letter is sent in response.*
- UDP is most often used in the context of network services for *real-time data traffic*, including *voice or video over IP*.
- In these cases, performance and latency are more important than packet retransmissions because re-sending a lost voice or video packet does not add value.

Open Systems Interconnection (OSI) Model

Presentation Layer

- Presentation layer is the sixth layer of OSI Model.
- It is concerned with the syntax & semantics of the information exchanged between the two devices.
- It was designed for data encryption, decryption, and compression.



Presentation Layer

Functions:

- **Data Presentation or Translation:** Because different computers use different encoding systems. It ensures that the data being sent is in the format that the recipient can process.
- **Data Encryption and compression:** PL provides this facility by which hides the information from everyone except the person who originally sent & the intended recipient. It also shrinks large amounts of data into smaller pieces i.e. it reduces the size of data.

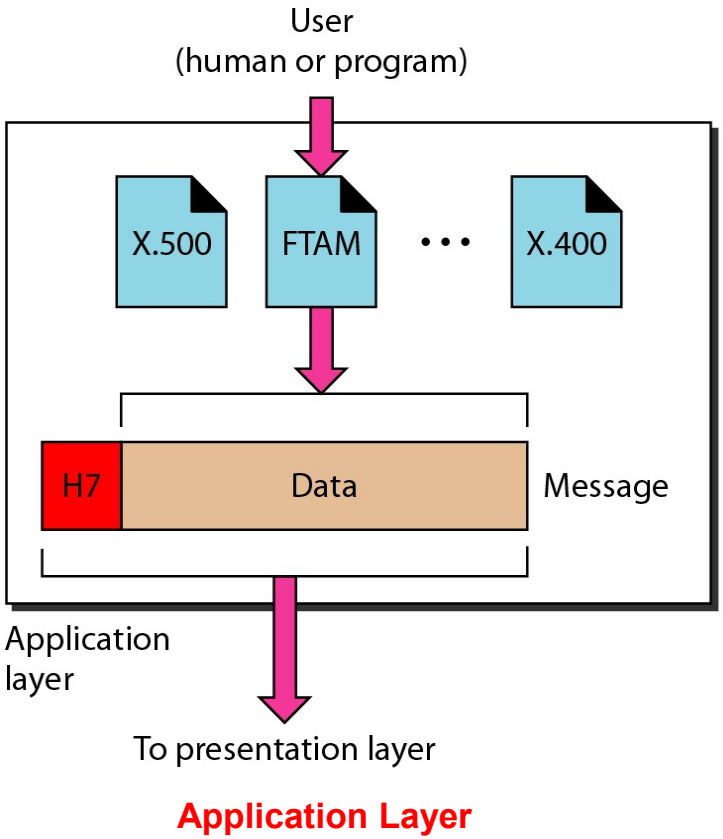
Open Systems Interconnection (OSI) Model

Application Layer

- It is the topmost i.e. seventh layer of the OSI Model. It enables the user to access the network.
- It provides user interface & support for services such as e-mail, file transfer, and access to the World Wide Web.
- So it provides services to different user applications.

Functions

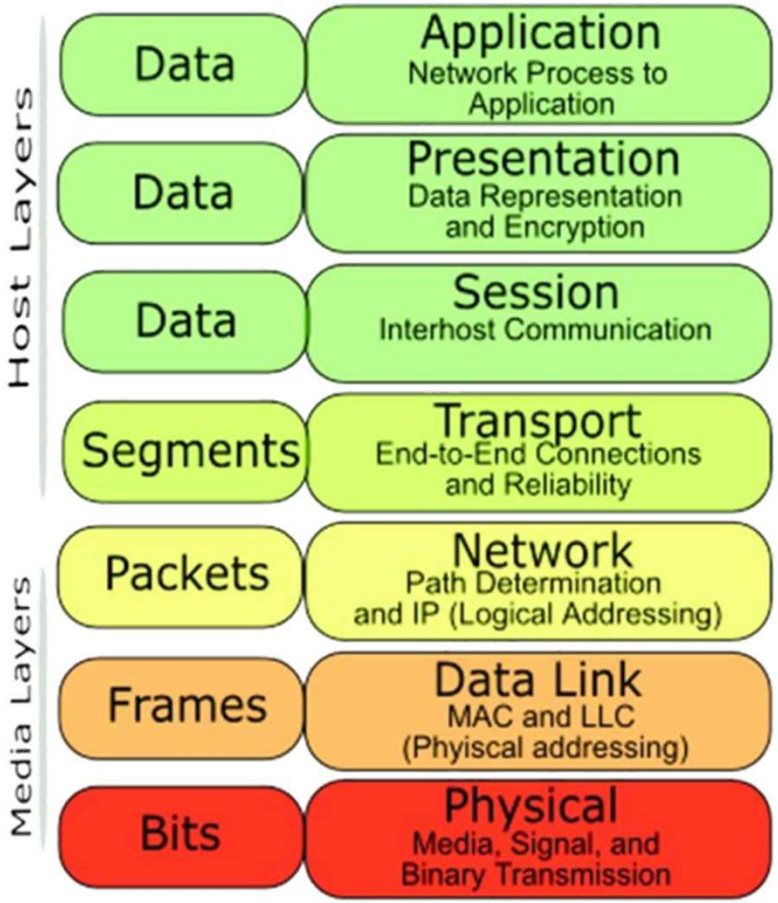
- **Mail Services:** This application provides various e-mail services.
- **File transfer & Remote log-in:** A user can log into a remote computer and access the resources of that computer



Application Layer

Open Systems Interconnection (OSI) Model

Summary

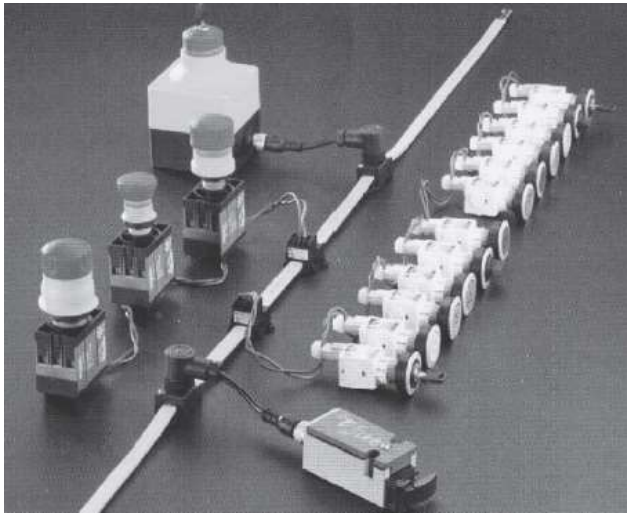


7	Application Layer	Human-computer interaction layer, where applications can access the network services
6	Presentation Layer	Ensures that data is in a usable format and is where data encryption occurs
5	Session Layer	Maintains connections and is responsible for controlling ports and sessions
4	Transport Layer	Transmits data using transmission protocols including TCP and UDP
3	Network Layer	Decides which physical path the data will take
2	Data Link Layer	Defines the format of data on the network
1	Physical Layer	Transmits raw bit stream over the physical medium

Industrial Network of PLCs

The Actuator Sensor-Interface Network

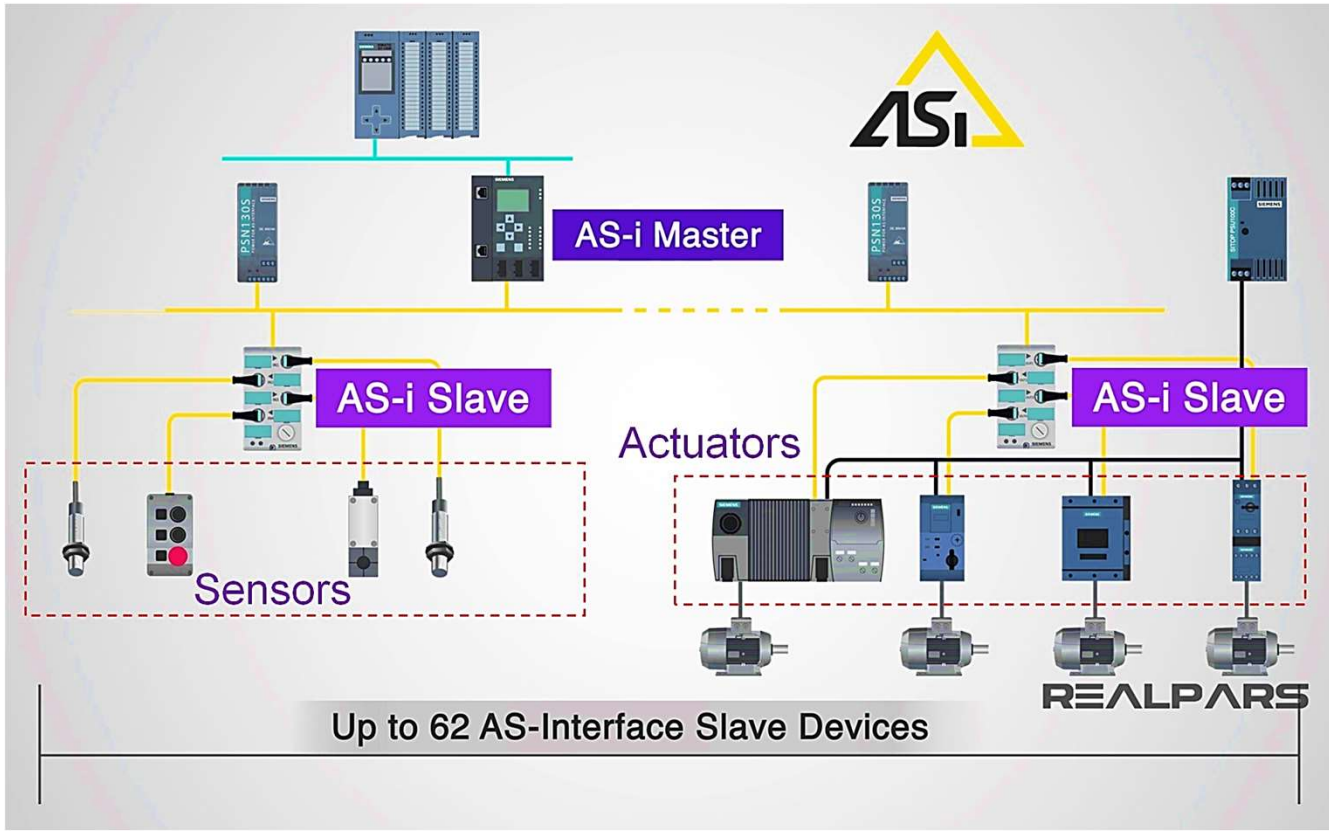
- In an industrial process, all actuators and sensors should be connected to the programmable automation system
- AS-I network streamlines industrial wiring by employing a single two-conductor cable for both *power and data transfer*, *minimizing complexity.*
- The special *yellow cable* of the AS-I network simplifies installation as it can be *connected without removing insulation, ensuring easy and efficient wiring*.
- Enhanced *data transmission security* is provided, while *enabling seamless interconnection* of sensors, actuators, and push buttons.
- AS-I network supports multiple topologies including *bus, star, ring, and tree* configurations, offering flexibility for different industrial process requirements.



Industrial Network of PLCs

The Actuator Sensor-Interface Network

- AS-I protocol works in *master-slave format*.
- Consists of an AS-I master module that can communicate with *up to 62 slave IO modules*.
- Each slave device supports up to *eight inputs and eight outputs* at a time simultaneously. So, a total of *992 IO's (496 inputs and 496 outputs)* can be handled properly by the master module.

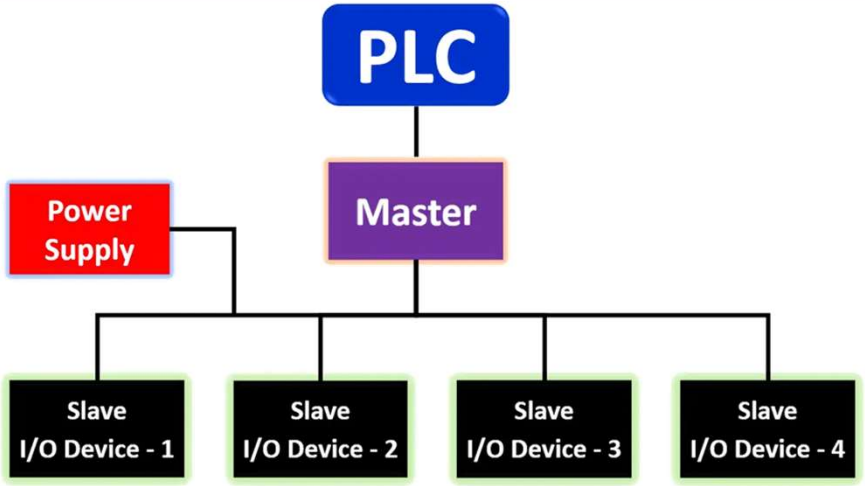


Industrial Network of PLCs

The Actuator Sensor-Interface Network

It consists **four** main components

1. Master module,
2. Slave module,
3. Power supply, and
4. Interface cables



1. Master

- Main component that *communicates the data* to PLC.
- *Controls data flow* between slaves and PLC in a *cyclic manner*, *manages troubleshooting data transfer to the PLC*.

2. Slave

- The *slave* modules are the actual modules where the *IO's are connected*.
- Each slave device is identified by a *device ID*.

Watch this video to understand more about AS-I: https://www.youtube.com/watch?v=S97rhReEnbo

3. Interface System

- Two standard cables are used in this system, namely **Yellow cable** carrying power (30 V) and data; and **Black cable** carrying 24V to actuators.
- A special piercing technology (mechanically keyed cable with connector) makes the *connection safe and easy* to use- *snap-in connectors*
- Due to the *mechanical pin key* connection, there is *no* chance of loose wiring or reverse polarity; as only the corresponding, pin will fit inside the connector.
- The maximum distance between the devices is *100 m*. Can be *extended to 600 m* by using extenders and repeaters

4. Power Supply

- Provides constant *30V DC* to power the Master & Slave systems
- Works as a *data decoupler-* Separates power and data which are simultaneously transferred

Watch this video to understand more about AS-I: https://www.youtube.com/watch?v=S97rhReEnbo

Difference between Ethernet & Serial Communication

- **High Scalability & High Expandability:** The number of RS232 serial devices that communicate with the computer is limited.
  - ✓As long as there is a network information port, you can directly connect the relevant Ethernet devices to Ethernet.
- **Configuration Ability:** When the computer needs to expand the number of RS232 ports, it is necessary to stop the computer.
  - ✓To add corresponding Ethernet equipment to the Ethernet, it is sufficient to connect the Internet cable to the nearby information port, and then configure it in the software.
- **Ease of use and maintenance:** Whether it is RS232 or RS485, it is generally only communicated with a single computer. It is difficult to form a dual-server. Once a problem occurs, the entire system may collapse.
  - ✓Industrial Ethernet switches are used to lay out an industrial network with redundant links. Once a link fails, it can self-heal and recover in 20 ms.
- **Suitability for flexible wiring**

Industrial Network Protocols

Fieldbus Protocols

- Fieldbus protocols are communication protocols used in industrial automation to *connect and control devices in a distributed system*.

Characteristics

- Digital communication protocols.
- Enable real-time control and monitoring.
- Support various field devices like sensors, actuators, and controllers.

Common Fieldbus Protocols

- **PROFIBUS**: Widely used in manufacturing and process automation.
- **Modbus**: Simple, open protocol used for serial communication.
- **FOUNDATION Fieldbus**: Supports both process control and factory automation.
- **DeviceNet**: Designed for connecting industrial devices in a network.

Advantages

- Increased efficiency through centralized control.
- Real-time monitoring and diagnostics.
- Reduced wiring complexity and cost.

Challenges

- Compatibility issues between different protocols.
- Initial setup and configuration complexity.
- Vulnerability to cyber threats.

Industrial Network Protocols

Industrial Ethernet Protocols

- Industrial Ethernet protocols are specialized communication protocols used in industrial automation for *reliable, high-speed data exchange*.

Characteristics

- Based on Ethernet standards with enhancements for industrial environments.
- Support real-time control and communication.
- Enable integration with enterprise networks.

Common Industrial Ethernet Protocols

- **Ethernet/IP**: A widely used protocol based on TCP/IP and CIP (Common Industrial Protocol).
- **PROFINET**: Developed by PROFIBUS International, combines Ethernet and IT standards with real-time communication.
- **Modbus TCP**: Extends the Modbus protocol over TCP/IP for Ethernet-based communication.
- **EtherCAT**: Real-time Ethernet protocol with high performance and low latency.

Advantages

- High-speed data transfer for real-time control.
- Scalability and flexibility for diverse industrial applications.
- Integration with existing Ethernet infrastructure.

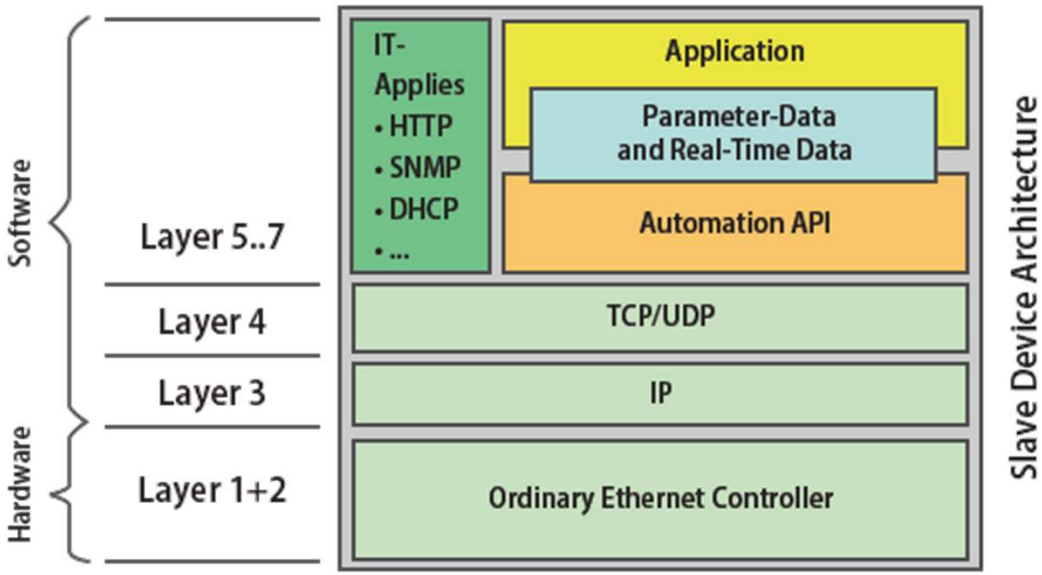
Challenges

- Network reliability in harsh industrial environments.
- Compatibility issues between different protocols.
- Security concerns in interconnected networks.

Note: Industrial Ethernet protocols are replacing other protocols in industrial automation due to *their higher performance, seamless integration with IT networks, real-time communication capabilities, interoperability, flexibility, cost efficiency, and adherence to industry standards*.

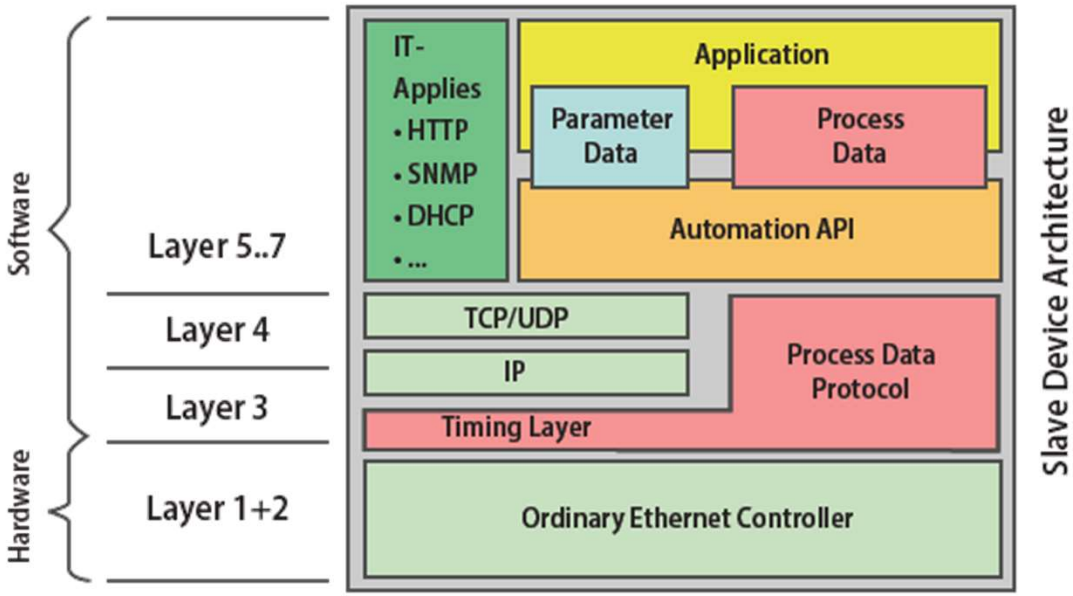
Ethernet/IP, Modbus TCP, and Standard PROFINET:

- Completely *TCP/UDP/IP* based.
- Utilize standard Ethernet switches and controllers.
- *No* specific *modifications to MAC layer*.
- Process data transmission over TCP/UDP/IP.



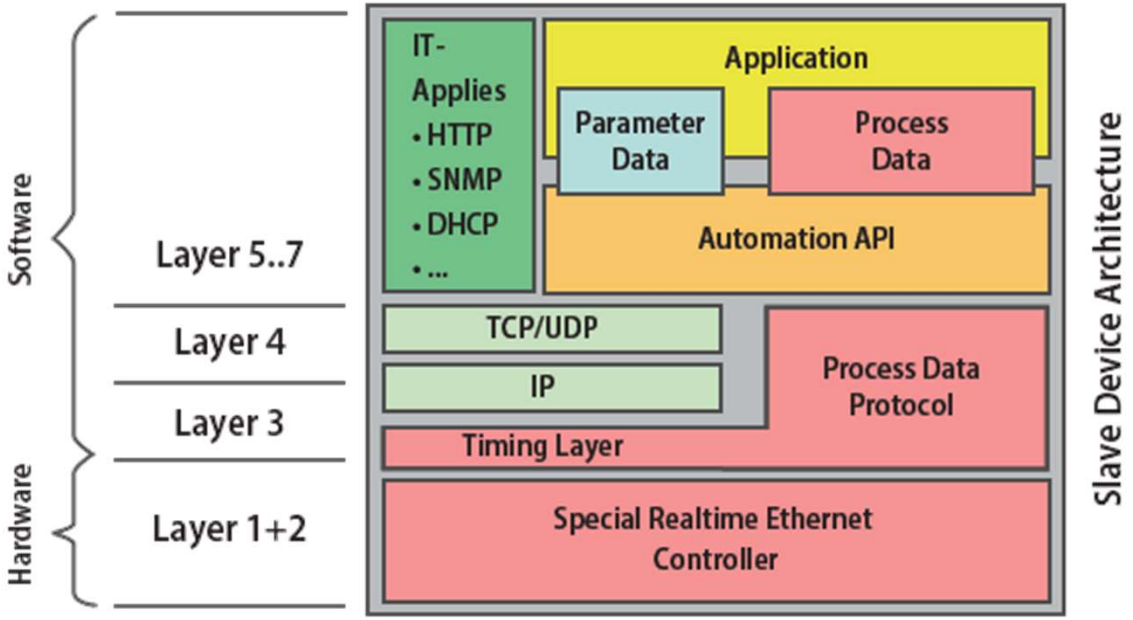
PROFINET RT and POWERLINK:

- Process data over *TCP/UDP/IP*.
- *Timing* controlled by a process data driver.
- *Compatible* with standard *Ethernet switches, hubs, and controllers*.



PROFINET IRT, CC-Link IE, SERCOS, and EtherCAT :

- Process data over *TCP/UDP/IP*.
- *Timing* controlled by a *process data driver*.
- Require *special Ethernet link layer/MAC hardware* for deterministic communication.
- Special hardware needed for protocol slave devices to ensure determinism.



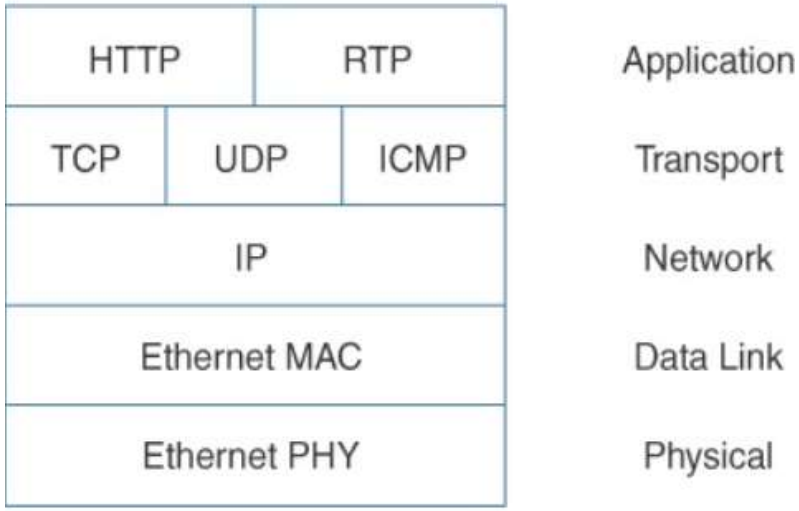
Encapsulation Mechanisms:

- Nearly all protocols provide mechanisms to encapsulate and carry standard Ethernet frames.
- Examples include Ethernet-Over-EtherCAT (EOE).

SCADA system- Communication architecture

- To enable SCADA to be IP compactable, the following IP protocol stack should be considered

IP Protocol Stack



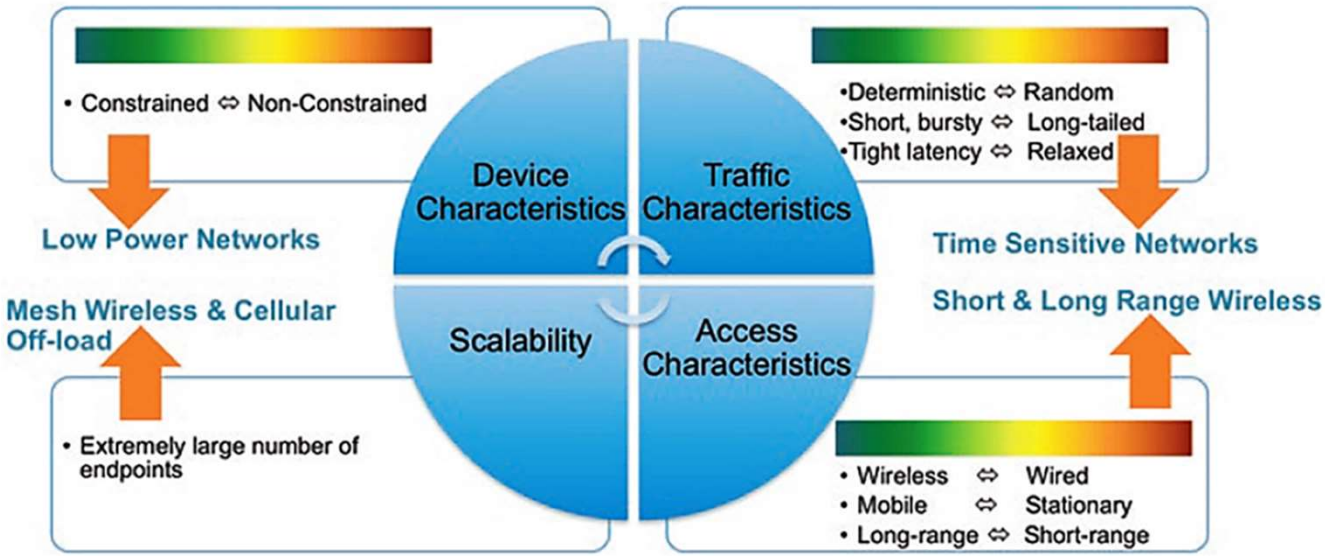
- Ethernet packet structure for transmitting data

Ethernet Header	IP Header	TCP/UDP header	Data

IoT Requirements for Networking (IoT Characteristics)

Major requirements for networking in IoT include:

- Support for Constrained devices- Device Characteristics
- Determinism- Traffic Characteristics
- Legacy device support
- Wireless Spectrum- Access Characteristics
- Scalability



Support for Constrained devices- Device Characteristics

- What are **constrained devices**?
- Small devices with limited processing, memory, and power resources are referred to as constrained devices
- A constrained device *is limited in one or more* of the following dimensions:
  - ✓ Maximum code complexity (ROM/Flash).
  - ✓ Size of run-time state and buffers (RAM).
  - ✓ Amount of computation feasible in a specific period of time (“processing power”).
  - ✓ Available power resources.

Fundamentals of IoT Networking

- The constrained devices typically have *limited energy resources* to spend on *processing and communication*
- *Network communication* is typically *more power-consuming* when compared to local processing.
- The Link layer, in particular, has a significant impact on constrained devices as this layer is responsible for *50% to 80%* of the communication energy.
- Constrained devices are classified into:
  - Class 0
  - Class 1
  - Class 2

Name	Data size	Code size
Class 0	<<10 KB	<<100 KB
Class 1	~10 KB	~100 KB
Class 2	~50 KB	~250 KB

Fundamentals of IoT Networking

Class 0 Devices

- *Constrained in memory and processing power:* Less than 10 KB of memory and less than 100 KB of flash processing and storage capacity. Typically *battery-powered*.
- *Devices do not have the resources to connect to an IP network:* Influence the services of proxies or gateways for connectivity.
- Do not have resources associated to security mechanisms.
- E.g.: *a push-button that sends 1 byte of information when changing its status*

Class 1 Devices

- *Constrained in terms of code space and processing capacity:* ~ 10KB RAM and 100 KB flash
- *Capable of connecting to an IP* (Internet Protocol) network directly without the help of gateways and provide necessary security functions.
- E.g.: *Environmental sensors*

Class 2 Devices

- Devices are less constrained when compared to the first two classes and are capable of running the same IP stack that runs on general computer nodes today.
- Have *more than 50 KB* of memory and *250 KB of* flash
- E.g.: *Smart power meter*

Industrial Network - Technologies

Traffic Characteristics

- The traffic characteristics of IoT endpoints vary widely depending on the *application's demands and the nature of the devices*.
- Some applications *compromise* on *packet loss, latency, and jitter* (e.g., a meteorological monitoring application).
- Others have *tight availability and latency* (e.g., a jet engine control application).

Determinism

- All real-time use cases and applications share a common requirement to support real-time transfer: *the time taken for each packet to traverse a path from its source to its destination should be determined; that is, the process must be deterministic*.
- A network is said to support determinism if the worst-case communication latency and jitter of messages are realistic

Challenge for Determinism

- *Migration* of special-purpose non-packet-based technologies (e.g., HDMI, CAN bus, Profibus, etc.) to IP technologies to support new applications and also, existing IP network applications over the same physical network.

Legacy Device Support

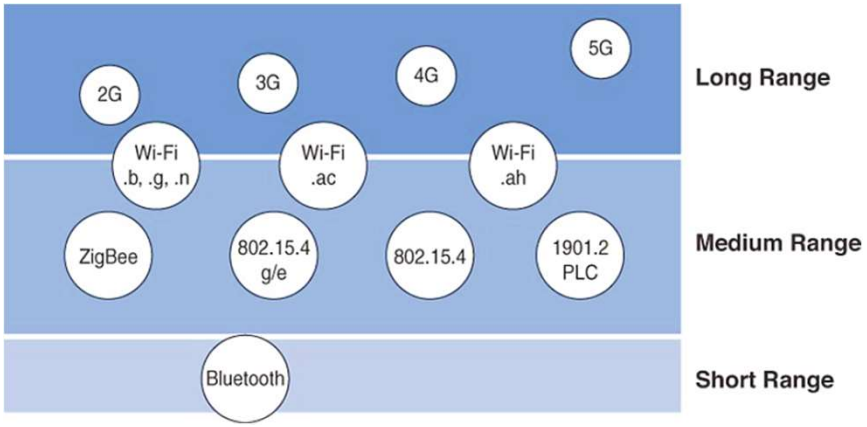
- Supporting legacy devices in an IT organization is not usually a big problem. If someone's computer or operating system is outdated, they simply upgrades.
- In OT systems, end devices are likely to be on the network for a very long time—sometimes decades
- For example, a factory may replace machines only once every 20 years—or perhaps even longer! It does not want to upgrade *multi-million-dollar machines just so it can connect them to a network for better visibility and control*.
- However, *many of these legacy machines might support older protocols, such as serial interfaces, and use RS-232.*

Wireless Spectrum- Access Characteristics

- The access characteristics of IoT endpoints become increasingly diverse as the footprint of the network grows beyond traditional IT environments
- To accommodate this diversity, new Link layer protocols that form the foundation of field area network (FAN), and personal area network (PAN) technologies are required.
- Modern wireless solutions come under *3 categories*

1. Short Range

- The classical wired example is a serial cable.
- Wireless short-range technologies are often considered as an alternative to a serial cable.
- Supports 10 meters of maximum distance between two devices.



- E.g.: **IEEE 802.15.1 Bluetooth** and **IEEE 802.15.7 Visible Light Communications**
- These communication methods are found in only a minority of IoT installations. They are not mature enough for production deployment.

Fundamentals of IoT Networking

2. Medium Range

- This range is the main category of IoT access technologies.
- The maximum distance is *generally less than 1 mile* between two devices.
- RF technologies do not have real maximum distances defined, as long as the radio signal is transmitted and received in the scope of the applicable specification.
- E.g.: *IEEE 802.11 Wi-Fi, IEEE 802.15.4, ZigBee and 802.15.4g WPAN.*
- Wired technologies such as *IEEE 802.3 Ethernet* and *IEEE 1901.2 Narrowband Power Line Communications (PLC)* may also be classified as medium range

3. Long Range

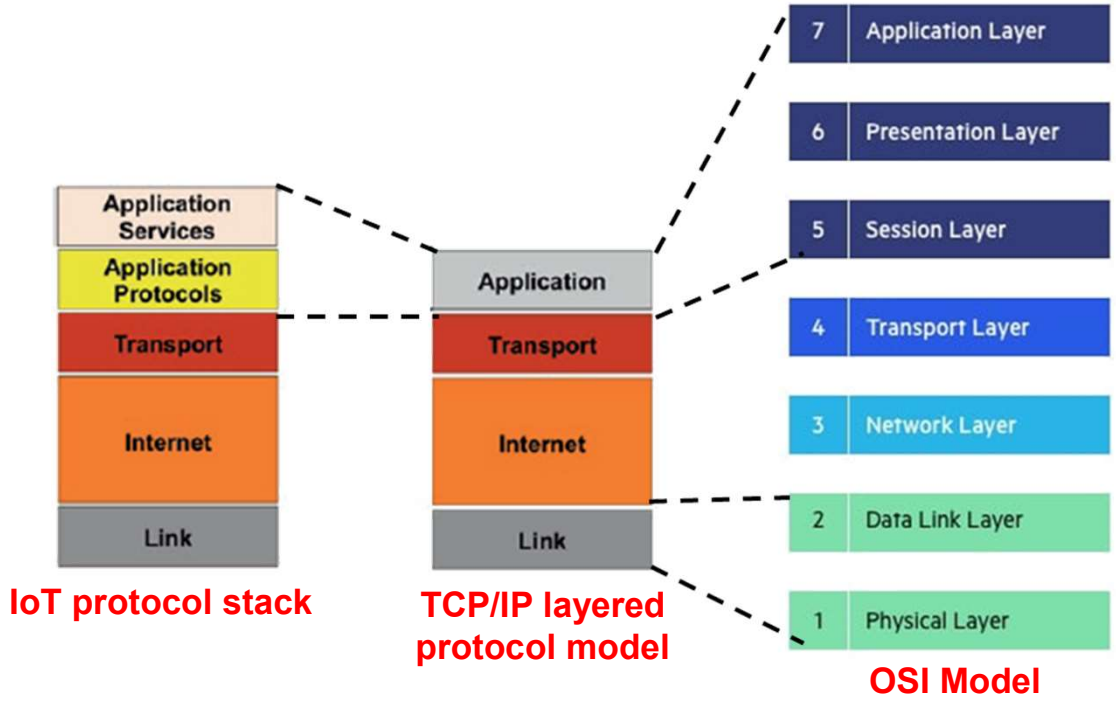
- Distances greater than *1 mile* between two devices require long-range technologies.
- E.g.: *Cellular networks (2G, 3G, 4G),* some applications of outdoor *IEEE 802.11 Wi-Fi* and *Low-Power Wide-Area (LPWA)* technologies.
- LPWA communications have the ability to communicate over a large area without consuming much power.
- These technologies are therefore *ideal* for *battery-powered IoT sensors*.

Scalability

- IoT scalability demands present interesting *challenges for the Link layer of the protocol st*ack, especially for wireless technologies.
- On the one hand, these technologies offer a number of appealing characteristics that make them a good fit for the IoT like:
  - Low upfront investments
  - Wide geographic coverage,
  - Fast deployment, and
  - No unsightly wires
- On the other hand, these technologies are *susceptible to scalability issues*.
- E.g.: cellular technologies are subject to the spectrum crunch problem

IIOT Networking and Protocols

- The IoT protocol stack can be visualized as an extension of the TCP/IP layered protocol model and is comprised of the following layers
  - Physical layer
  - Link layer
  - Network layer
  - Transport layer
  - Application Protocols layer
  - Application Services layer
- **The Application protocol layer** is responsible for the efficient communication between the applications and smart objects and are responsible for exchanging data between the devices.



- **The Application Services layer** acts as the middleman for the working of varied applications and provides the interoperability among the various applications and the devices.

Application Layer Challenges

- A challenge in IoT *data serialization* formats is the impact they have on *device resource utilization, especially in terms of energy consumption.*
- *Data serialization* is the process of converting data objects present in complex data structures into a byte stream for storage, transfer and distribution purposes on physical devices
- Data formats have an effect on device resource usage in two aspects:
  - Local processing demands and
  - Communication efficiency.
- The *local processing demands* include both the processing required to serialize memory objects into data encoded in messages and the processing required to parse the encoded messages into memory objects.
- The *communication efficiency* is a function of the compactness of the data serialization format and its efficiency to encode information in the least amount of message real estate.

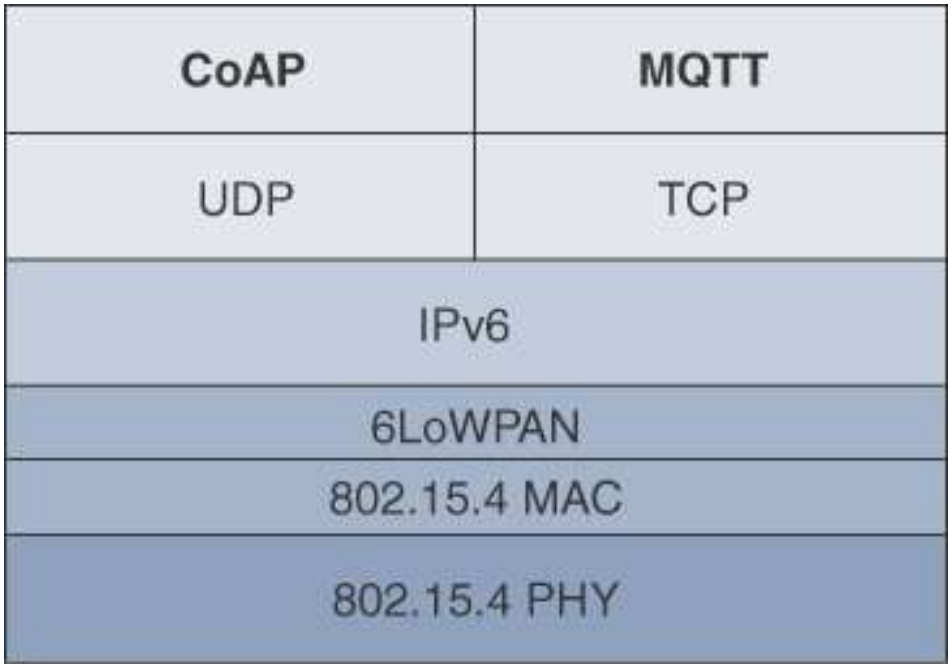
IIOT Networking and Protocols

- A third challenge in IoT data serialization formats is the impact they have on network *bandwidth utilization*.
- This *ties back* to the compactness of the format and its encoding efficiency, as discussed above.
- The *more verbose* that the data format is, the *more message space* that it will consume on the wire to carry the same amount of information, which leads to less efficient use of network bandwidth.
- For IoT, especially when devices are connected over low-bandwidth wireless links, the data serialization format of application protocols should be chosen carefully to maximize the use of the available bandwidth.
- When considering constrained networks and/or a large-scale deployment of constrained nodes, verbose web-based and data model protocols (HTTP), may be too heavy for IoT applications.

- **HTTP is a synchronous protocol.** The client waits for the server to respond. That is a requirement for web browsers, but it comes at the cost of poor scalability. In the world of IoT, the large number of devices and most likely an unreliable / high latency network have made synchronous communication problematic. An asynchronous messaging protocol is much more suitable for IoT applications. The sensors can send in readings, and let the network figure out the optimal path and timing for delivery to its destination devices and services.
- **HTTP is one-way.** The client must initiate the connection. In an IoT application, the devices or sensors are typically clients, which means that they cannot passively receive commands from the network.
- **HTTP is a 1-1 protocol.** The client makes a request, and the server responds. It is difficult and expensive to broadcast a message to all devices on the network, which is a common use case in IoT applications.
- **HTTP is a heavy weight protocol** with many headers and rules. It is not suitable for constrained networks

IIOT Networking and Protocols

- To address this problem, the IoT industry is working on new lightweight protocols that are better suited to large numbers of constrained nodes and networks.
- Two of the most popular protocols are **CoAP** and **MQTT**.

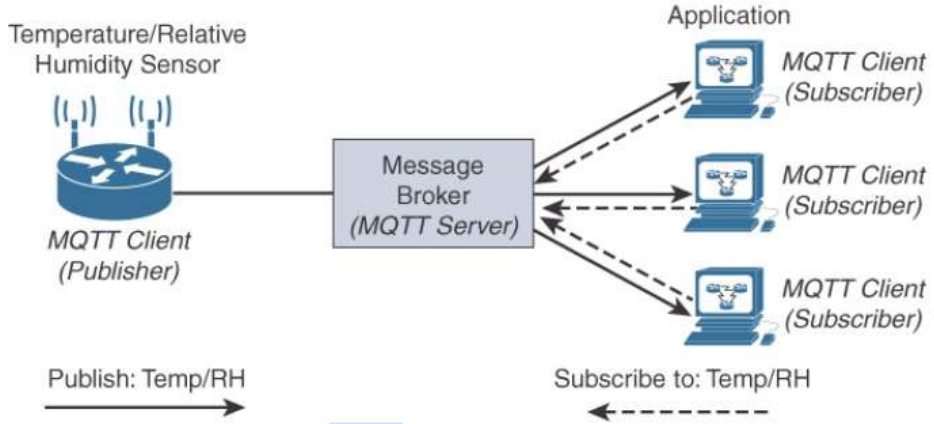


Message Queuing Telemetry Transport (MQTT)

- At the end of the 1990s, engineers from IBM and Arcom developed *Message Queuing Telemetry Transport (MQTT)-* a *reliable, lightweight and cost-effective* protocol to monitor and control a large number of sensors and their data from a central server location, as typically used by the oil and gas industries.
- It is a client/server and *publish/subscribe* framework based on the TCP/IP architecture
- *Publish/Subscribe framework:* An asynchronous framework for exchanging messages between publishers (senders of messages) and subscribers (specific receivers). This pattern involves the publisher and the subscriber relying on a **message broker** that relays messages from the publisher to the subscribers.
- *Message broker:* Intermediate computer program module that translates a message from the formal messaging protocol of the sender to the formal messaging protocol of the receiver.

MQTT Publish/Subscribe Framework

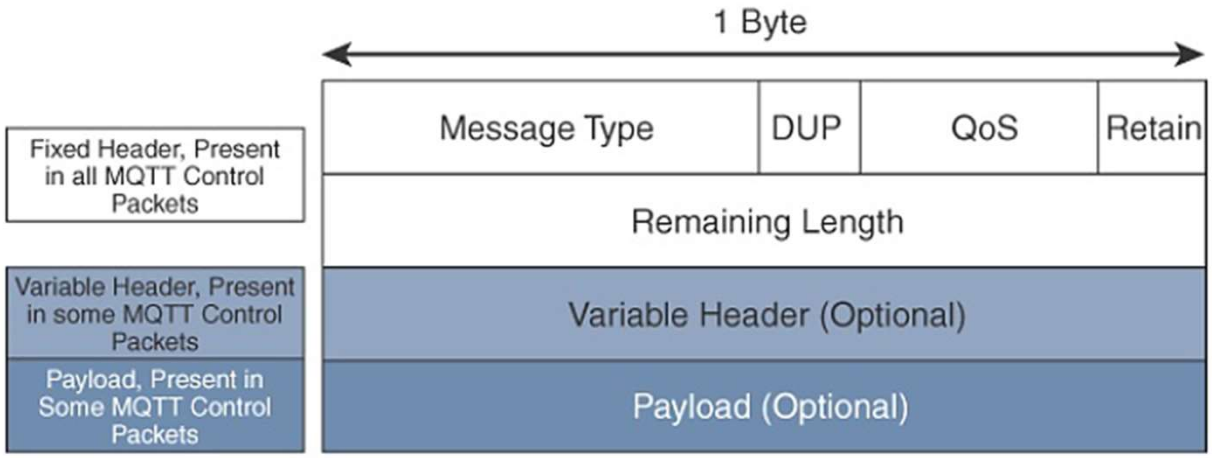
- An MQTT client can act as a publisher to send data to an MQTT server acting as an MQTT message broker
- The MQTT client on the left side is a temperature and relative humidity sensor that publishes its Temp/RH data.



- The MQTT server accepts network connection along with Temp/RH data from the publisher, handles subscription and un-subscription processes and pushes the data to MQTT subscribers.
- With MQTT, clients can subscribe to all data (using a wildcard character) or specific data from the information tree of a publisher.
- *The message broker decouples the data transmission between publishers and subscribers.*
- *The publishers and subscribers do not need to know about each other.*
- ***Benefit of decoupling:*** Message broker ensures that information can be buffered and cached in case of network failures.
- Also means that *publishers and subscribers do not have to be online at the same time.*

MQTT Message Format

- *MQTT is a lightweight protocol:* each control packet consists of a 2-byte fixed header with optional variable header fields and an optional payload.
- Control packet can contain a *payload up to 256 MB*.



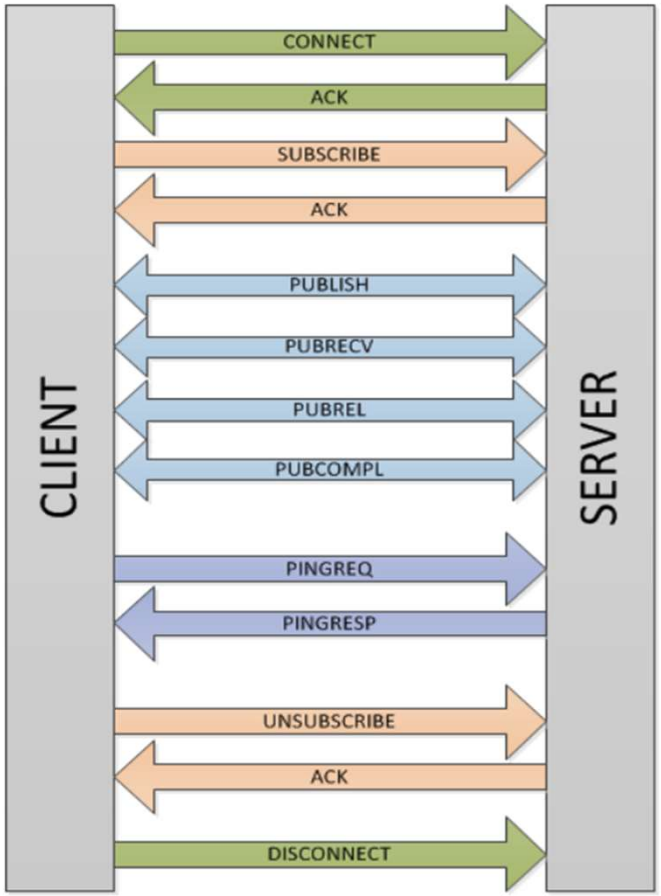
1. *Message Type*: Identifies the type of MQTT packet within a message.
   - There are 14 different types of control packets
   - Each of them has a unique value that is coded into the Message Type field.
   - Values 0 and 15 are reserved

2. *DUP (Duplication Flag):* This flag, when set, allows the client to notate that the packet has been sent previously, but an acknowledgment was not received.

IIOT Networking and Protocols

Message Types

Message Type	Value	Flow	Description
CONNECT	1	Client to server	Request to connect
CONNACK	2	Server to client	Connect acknowledgement
PUBLISH	3	Client to server Server to client	Publish message
PUBACK	4	Client to server Server to client	Publish acknowledgement
PUBREC	5	Client to server Server to client	Publish received
PUBREL	6	Client to server Server to client	Publish release
PUBCOMP	7	Client to server Server to client	Publish complete
SUBSCRIBE	8	Client to server	Subscribe request
SUBACK	9	Server to client	Subscribe acknowledgement
UNSUBSCRIBE	10	Client to server	Unsubscribe request
UNSUBACK	11	Server to client	Unsubscribe acknowledgement
PINGREQ	12	Client to server	Ping request
PINGRESP	13	Server to client	Ping response
DISCONNECT	14	Client to server	Client disconnecting



3. QoS (Quality of Service Header): Allows for the selection of three different QoS levels

4. Retain Flag: Only found in a *PUBLISH message*. Notifies the server to hold onto the message data. This allows new subscribers to instantly receive the last known value without having to wait for the next update from the publisher.

5. Remaining Length: specifies the *number of bytes in the MQTT packet* following this field

MQTT Quality of Service (QoS)

- The MQTT protocol offers *three levels of quality of service (QoS)*.
- QoS for MQTT is implemented when exchanging application messages with publishers or subscribers.
- The delivery protocol is symmetric: client and server can each take the role of either sender or receiver.
- The delivery protocol is concerned solely with the delivery of an application message from a single sender to a single receiver.
- The 3 QoS are:
  - QoS 0: 'at most once'
  - QoS 1: 'at least once'
  - QoS 2: 'exactly once'

IIOT Networking and Protocols

1. QoS 0: 'at most once'- (Fire & Forget)

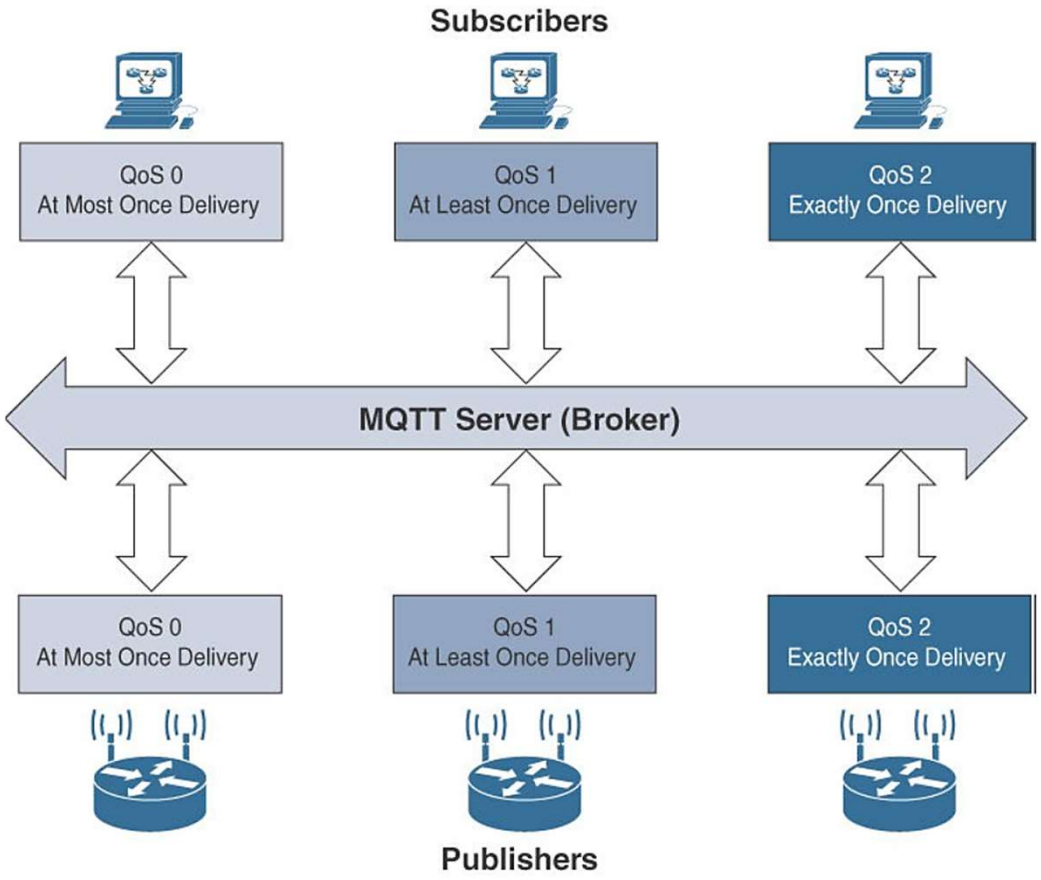
- This is a *best-effort and unacknowledged data service* referred to as *"at most once"* delivery.
- The publisher sends its message *one time* to a server, *which transmits it once* to the subscribers.
- *No response is sent by the receiver,* and *no retry is performed by the sender*.
- The message arrives at the receiver *either once or not at all.*

2. QoS 1: 'at least once'- (Acknowledged delivery)

- This QoS level ensures that the message delivery between the publisher and server and then between the server and subscribers *occurs at least once*.
- In PUBLISH and PUBACK packets, a *packet identifier is included* in the variable header.
- If the message is not acknowledged by a PUBACK packet, *it is sent again*.
- This level *guarantees "at least once"* delivery.

3. QoS 2: 'exactly once'- (Assured delivery)

- This is the highest QoS level, used when *neither loss nor duplication of messages is acceptable*.
- There is an increased overhead associated with this QoS level because each packet contains an optional variable header with a packet identifier.
- Confirming the receipt of a PUBLISH message requires a two-step acknowledgment process.
- The *first step* is done through the *PUBLISH/PUBREC packet pair*, and the *second* is achieved with the *PUBREL/PUBCOMP packet pair.*
- *This level provides a "guaranteed service" known as "exactly once" delivery,* with no consideration for the number of retries as long as the message is delivered once.



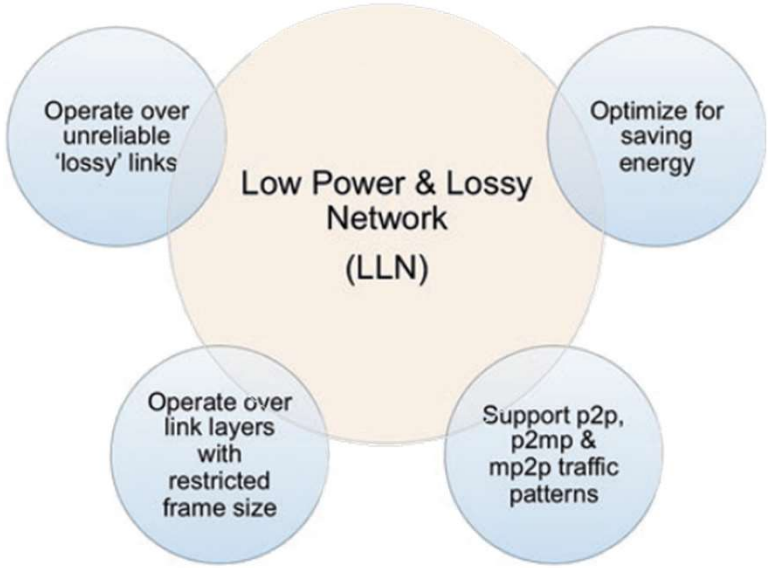
Internet layer Challenges

- Many IoT deployments constitute what is referred to as *low-power and lossy networks (LLNs).*
- LLNs comprise of several thousand of constrained embedded devices with *limited power, memory, and processing resources.*
- They are interconnected using a variety of Link-layer technologies, such as *IEEE 802.15.4* (Low-rate Wireless Personal Area Network) , *Bluetooth, Wi-Fi, or power-line communication* (PLC) links.
- There is a wide scope of applications for LLNs, including
  - Industrial monitoring,
  - Building automation (HVAC, lighting, access control, fire),
  - Connected homes, healthcare,
  - Urban sensor networks (e.g., smart grid), and
  - Asset tracking

IIOT Networking and Protocols

- LLNs present the following five challenges to the Internet layer of the protocol stack:

1. Optimized for saving energy. Various techniques are used to that effect, including employing extended sleep cycles, where the embedded devices only wake up and connect to the network when they have data to send.
2. Traffic patterns within LLNs include point-to-point, point-to-multipoint, and multipoint-to-point flows. As such, unicast and multicast considerations should be taken into account when designing protocols for this layer.
3. LLNs will typically be employed over Link layer technologies characterized with restricted frame sizes.



4. Links within LLNs may be inherently unreliable with time-varying loss characteristics. The protocols need to offer high reliability under those characteristics.

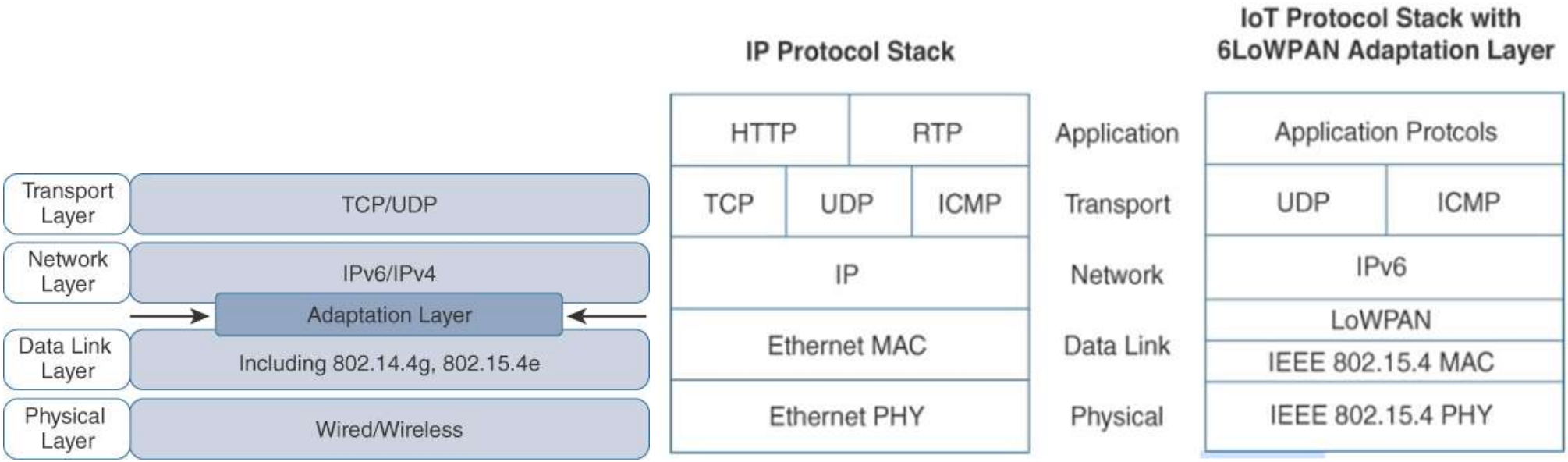
- Protocols for LLNs should take into account the link speeds and the device capabilities.
- If the devices are battery-powered, then protocols that require frequent communication will deplete the nodes' energy faster.

6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks)

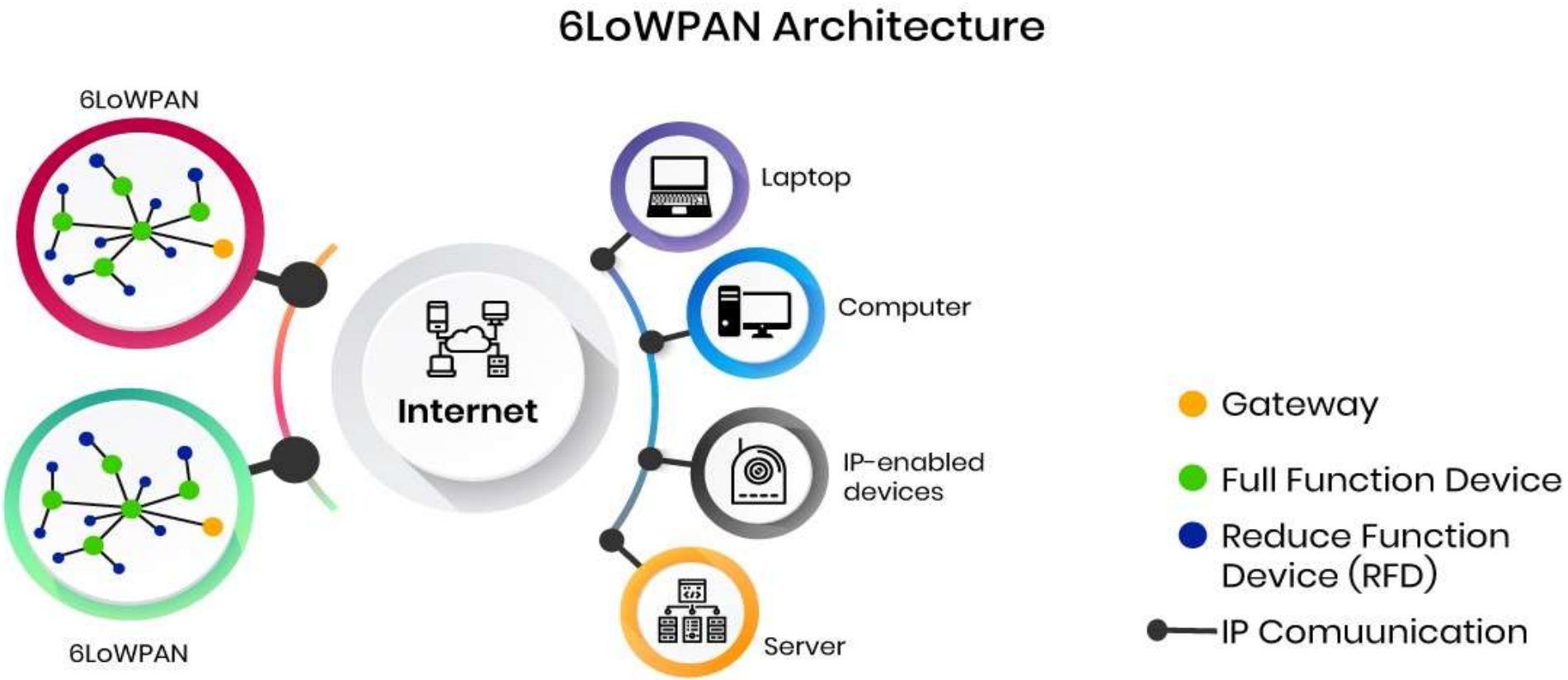
- One of the challenges imposed by IoT on the Internet layer is the adaptation of the internet layer's functions to **Link-layer technologies with restricted frame size**.
- Considering the case of adapting IPv6 functionalities to IEEE 802.15.4 Link layer, the maximum frame size of IEEE 802.15.4 Link layer is **127 bytes**.
  - **25 bytes** need to be reserved for the frame header
  - **21 bytes** for link-layer security.
  - Maximum **81 bytes** remaining for filling in IPv6 packet
- But, IPv6 requires packet transmission with **1280 bytes.**
  - Header itself is **40 bytes**
  - IPv6 does not perform segmentation and reassembly of packets; this function is left to the end stations or to lower layer protocols.
- To solve this issue, may need to have an **adaptation layer** for running IPv6 over IEEE 802.15.4 networks

IIOT Networking and Protocols

- ***6LoWPAN*** *is a standard protocol* that operates in the adaptation layer which realizes IPv6 communication on wireless networks composed of *low-power wireless modules*.
- The initial focus of the 6LoWPAN working group was to *optimize the transmission of IPv6 packets* over constrained networks such as IEEE 802.15.4.



IIOT Networking and Protocols

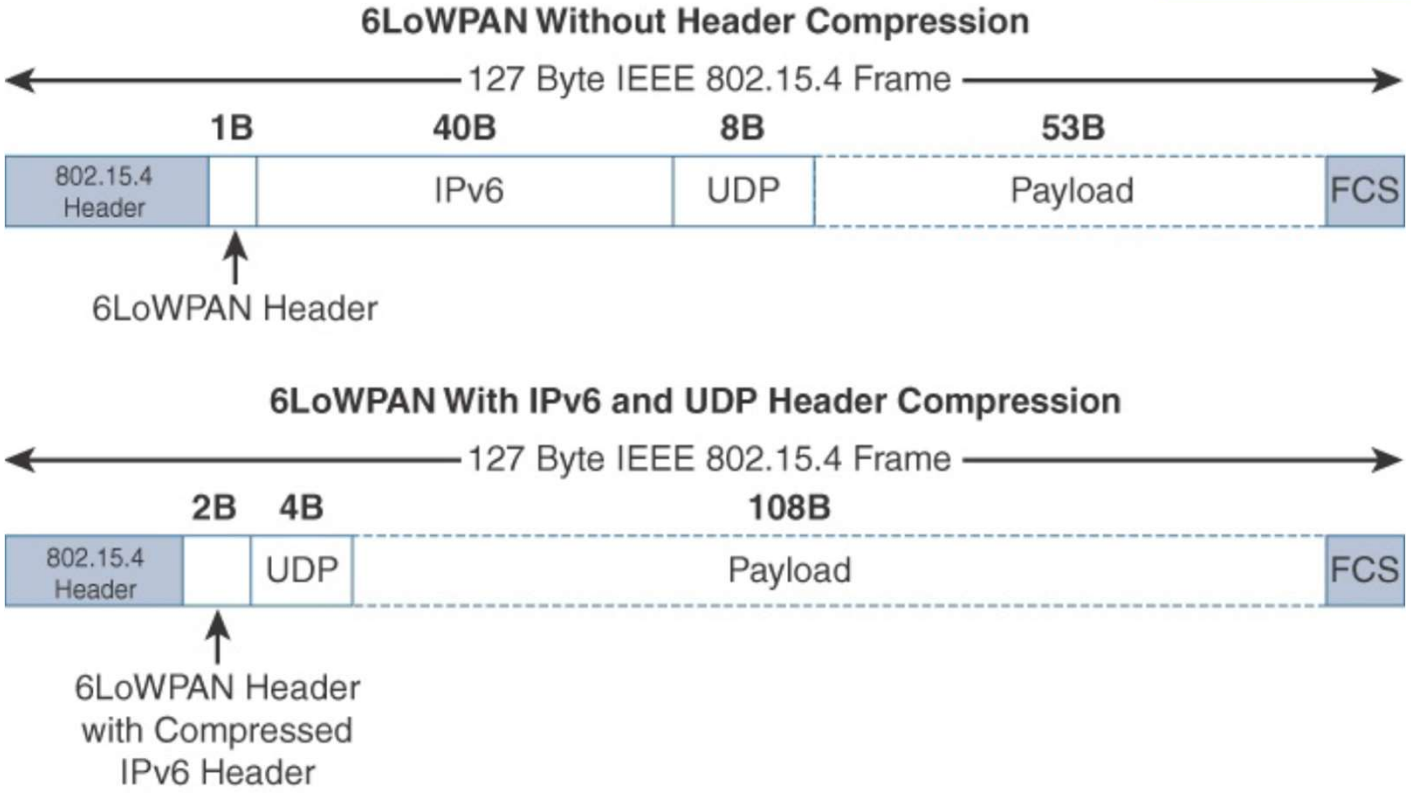


IIOT Networking and Protocols

- The adaptation layer has three main functions:
  - ✓ *IPv6 header compression,*
  - ✓ *IPv6 fragmentation and reassembly, and*
  - ✓ *Mesh addressing /Routing.*

1. IPv6 header compression

- IPv6 header compression is very important to *reduce overhead and increase application payload space.*
- Header compression for 6LoWPAN is *only defined for an IPv6 header and not IPv4*.
- The 6LoWPAN protocol does not support IPv4.
- Internet Engineering Task Force (IETF) working group has released several *Request for Comments (RFC)*
- RFC 6282 defines how to compress the IPv6 and UDP headers efficiently *using improved header compression (IPHC) and next header compression(NHC) methods.*
- This capability shrinks the size of *IPv6's 40-byte headers* and ***UDP's 8-byte*** headers down as low as *6 bytes* combined in some cases.

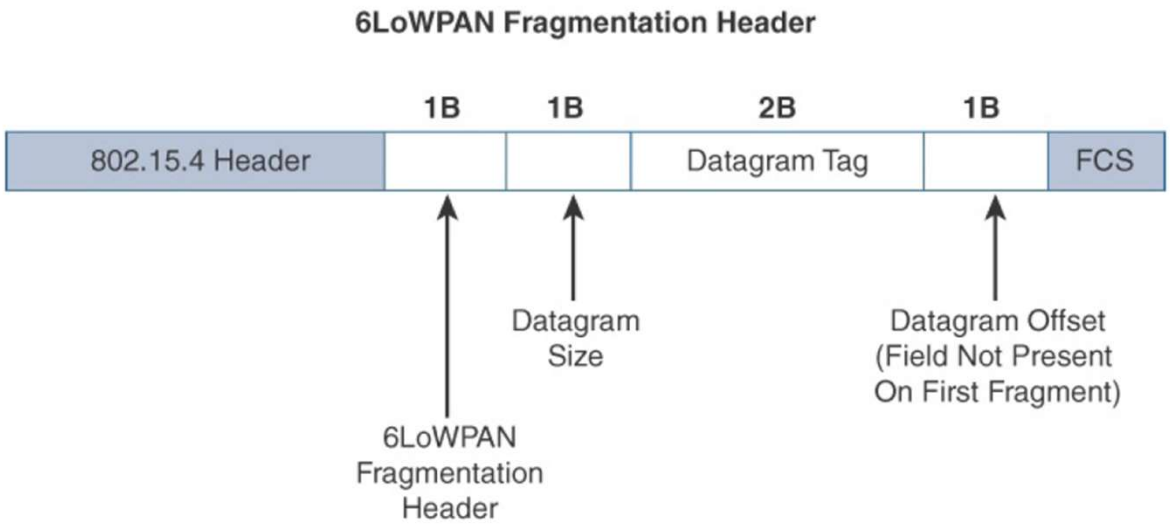


- **Frame check sequence (FCS)** is a calculation based on the data in the frame that is used by the receiving side to confirm the integrity of the data in the frame.

2. IPv6 fragmentation and reassembly

- The major function of the adaptation layer is **IPv6 fragmentation and reassembly.**
- When an IPv6 packet does not fit into a single IEEE802.15.4 data frame, the packet is divided into fragments
- Each of which is sent over a single IEEE 802.15.4 frame.
- When all fragments are received at the destination, the IPv6 packet is reassembled and delivered up to the network layer.
- The 6LoWPAN fragmentation header field itself uses a unique bit value to identify that the subsequent fields behind it are fragment fields as opposed to another capability, such as header compression.

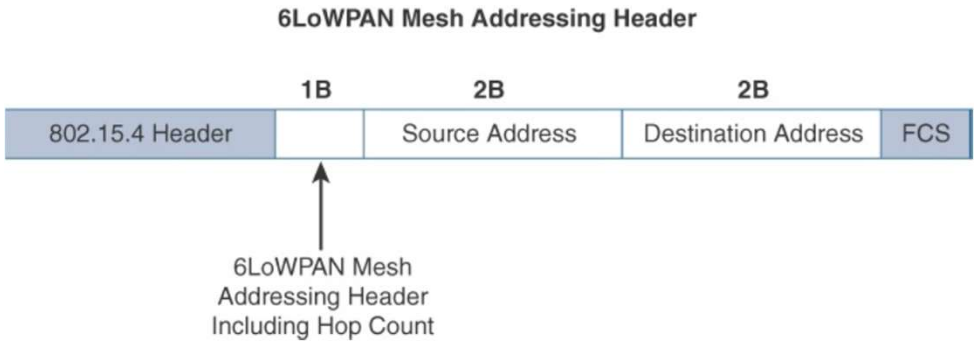
- The fragment header utilized by 6LoWPAN is composed of *three* primary fields:
  - *Datagram Size:* The 1-byte Datagram Size specifies the size of the defragmented payload.
  - *Datagram Tag* identifies each set of fragments of a payload.
  - *Datagram Offset* field delineates how far into a payload a particular fragment occurs.



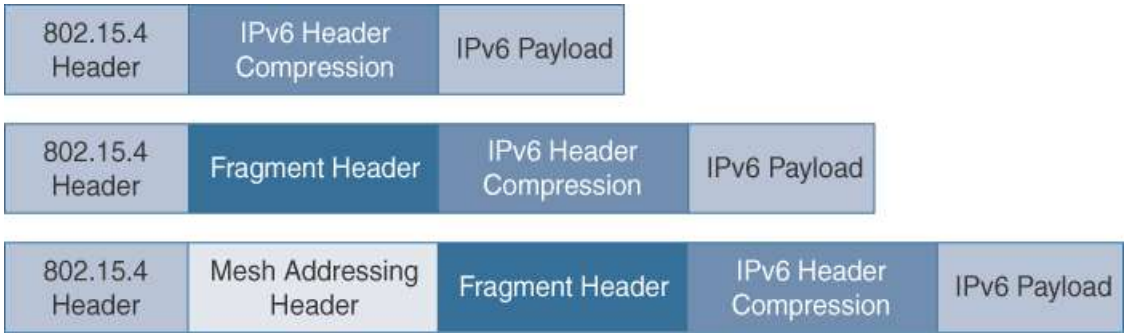
- In the first fragment, the Datagram Offset field is not present because it would simply be set to 0.
- Hence, the *first fragmentation header* for an IPv6 payload being only *4 bytes long*.
- *Remaining fragments* have a *5-byte header field* so that the appropriate offset can be specified.

3. Mesh Addressing/Routing

- The *purpose* of the 6LoWPAN mesh addressing function is to *forward packets over multiple hops.*
- *Hops:* the trip a data packet takes from one router or intermediate point to another in the network.
- The hop limit for mesh addressing also provides an upper limit on how many times the frame can be forwarded.



- Each hop *decrements* this value *by 1* as it is forwarded. *Once the value hits 0, it is dropped and no longer forwarded.*
- The *Source Address* and *Destination Address* fields for mesh addressing are *IEEE 802.15.4 addresses* indicating the *endpoints* of an IP hop.



6LoWPAN Message format

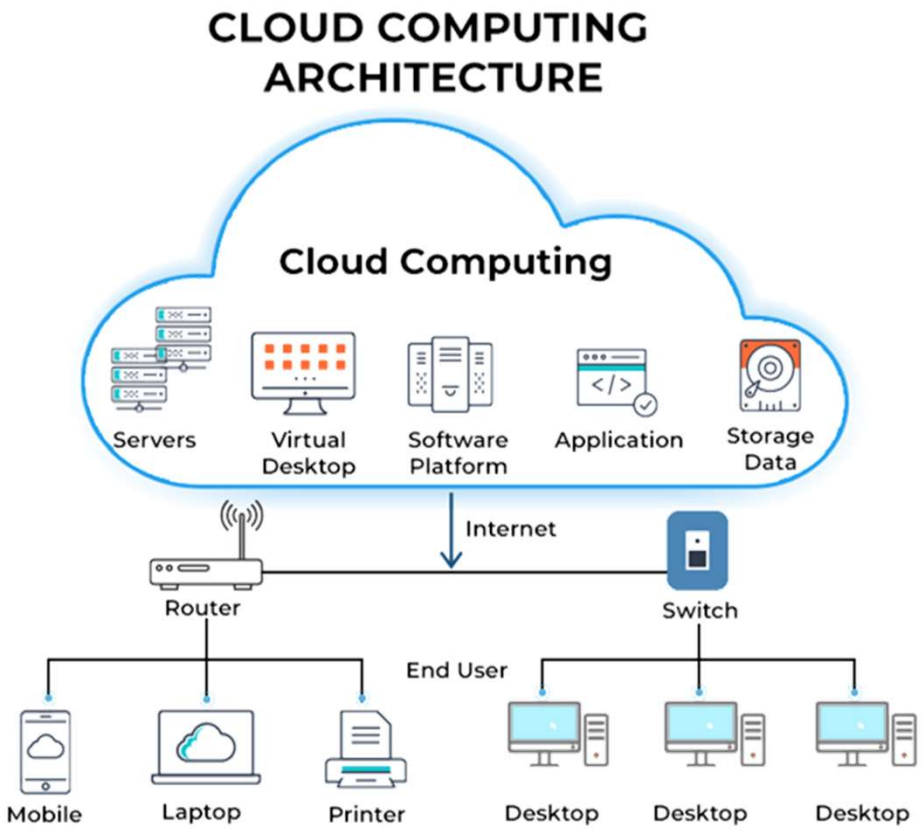
Cloud computing

- Cloud computing is the on-demand availability of **computer system resources**, especially **data storage (cloud storage)** and **computing power**, without direct active management by the user.
- Up until recently, enterprises were forced to deploy and manage their own computing infrastructures.
- Cloud computing, which was introduced in 2008, allows enterprises to outsource their computing infrastructure fully or partially to public cloud providers
- Cloud provides instant availability,
  - Visualization
  - Reliability
  - Scalability

  without the hassles of maintenance or infrastructure.
- **Public Cloud** providers deliver cloud services, on-demand, over the Internet. Enterprises pay only for the CPU cycles, storage, or bandwidth they consume.
- **Private Cloud** solutions are deployed by enterprises in their own data centers and deliver computing services to their internal sub-businesses/users.

Cloud computing

- Instead of storing files on a storage device or hard drive, a user can save them on the cloud, making it possible to access the files from anywhere, as long as they have access to the web.
- Cloud can be divided into *two different layers*:
  - Front-End
  - Back-End
- **Front-End:** The *layer with which users interact.* This layer enables a user to access the data that has been stored in the cloud through cloud computing software.
- **Back-End:** The *layer made up of software and hardware*, i.e., the computers, servers, central servers, and databases. This layer is the primary component of the cloud and is entirely responsible for storing information securely.



Characteristics of Cloud computing

- **On-demand self-service:** A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.
- **Broad network access.:** Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g. mobile phones, tablets, laptops, and workstations)
- **Resource pooling:** The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. Resources include storage, processing, memory, and network bandwidth
- **Rapid elasticity:** If the level of computing and storage exceeds the allocated level, the cloud will dynamically stretch to accommodate the demand without any customer interaction.
- **Measured service:** Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g. storage, processing, bandwidth, and active user accounts.

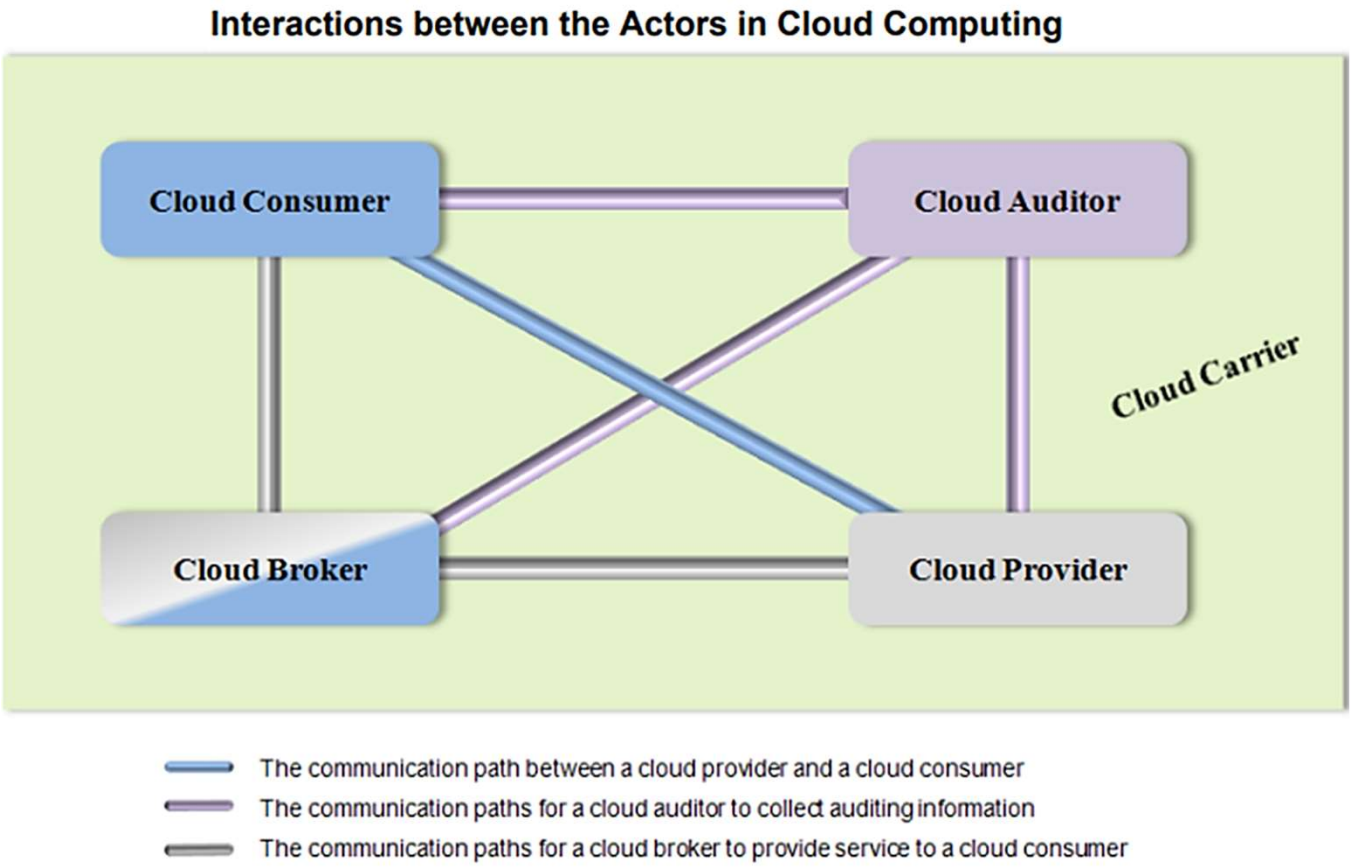
Actors of Cloud computing

- An actor is an entity (a person or an organization) that participates in a transaction or process and/or performs tasks in cloud computing.

Actor	Definition
Cloud Consumer	A person or organization that maintains a business relationship with, and uses service from, *Cloud Providers*.
Cloud Provider	A person, organization, or entity responsible for making a service available to interested parties.
Cloud Auditor	A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation.
Cloud Broker	An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between *Cloud Providers* and *Cloud Consumers*.
Cloud Carrier	An intermediary that provides connectivity and transport of cloud services from *Cloud Providers* to *Cloud Consumers*.

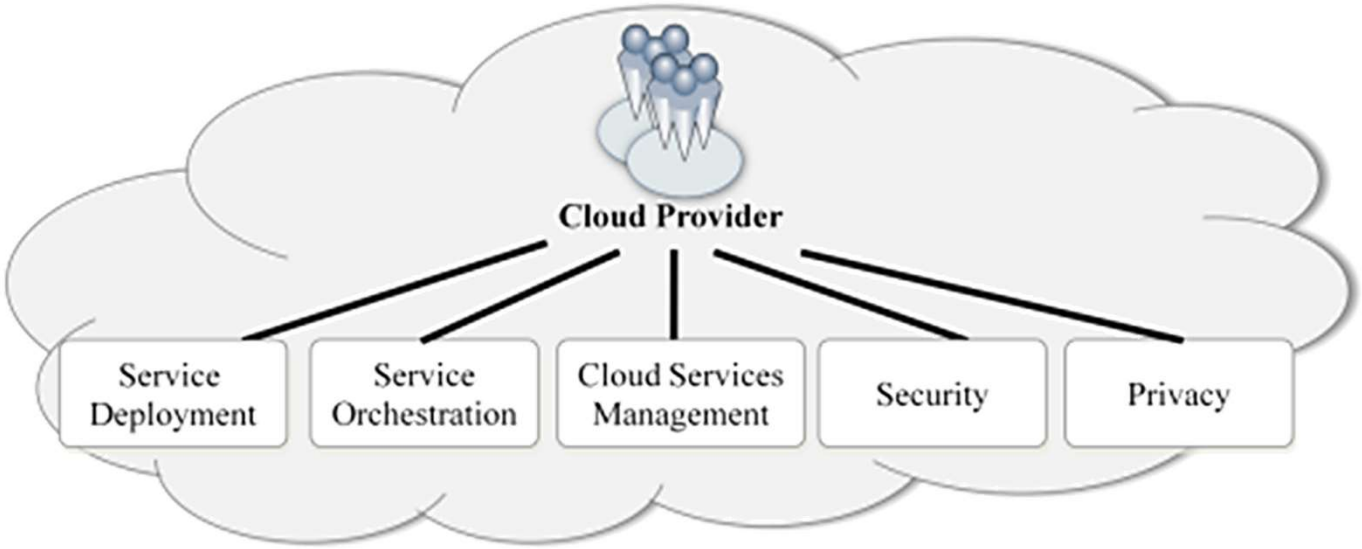
Cloud computing

- A cloud consumer may request cloud services from a cloud provider directly or via a cloud broker.
- A cloud auditor conducts independent audits and may contact the others to collect the necessary information.



Cloud Provider Activities

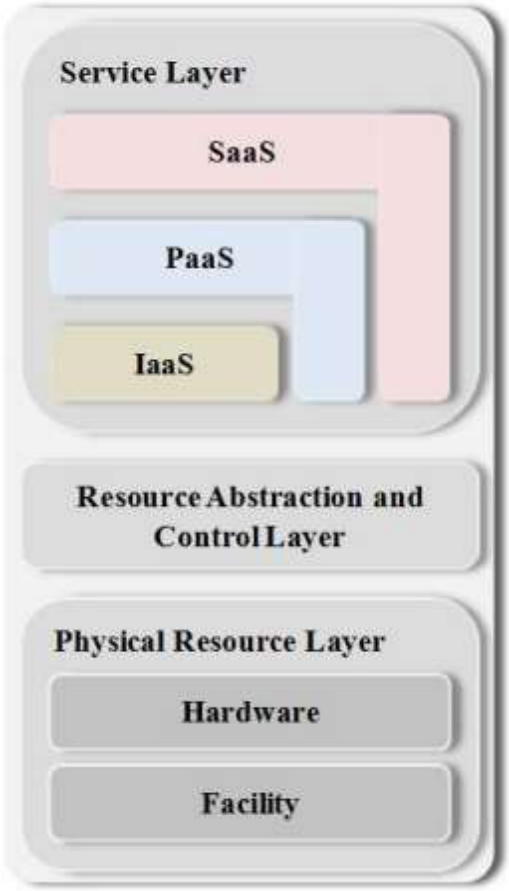
- A Cloud Provider acquires and manages the computing infrastructure required for providing the services, runs the cloud software that provides the services, and makes arrangements to deliver the cloud services to the Cloud Consumers through network access.
- A Cloud Provider's activities can be described in five major areas



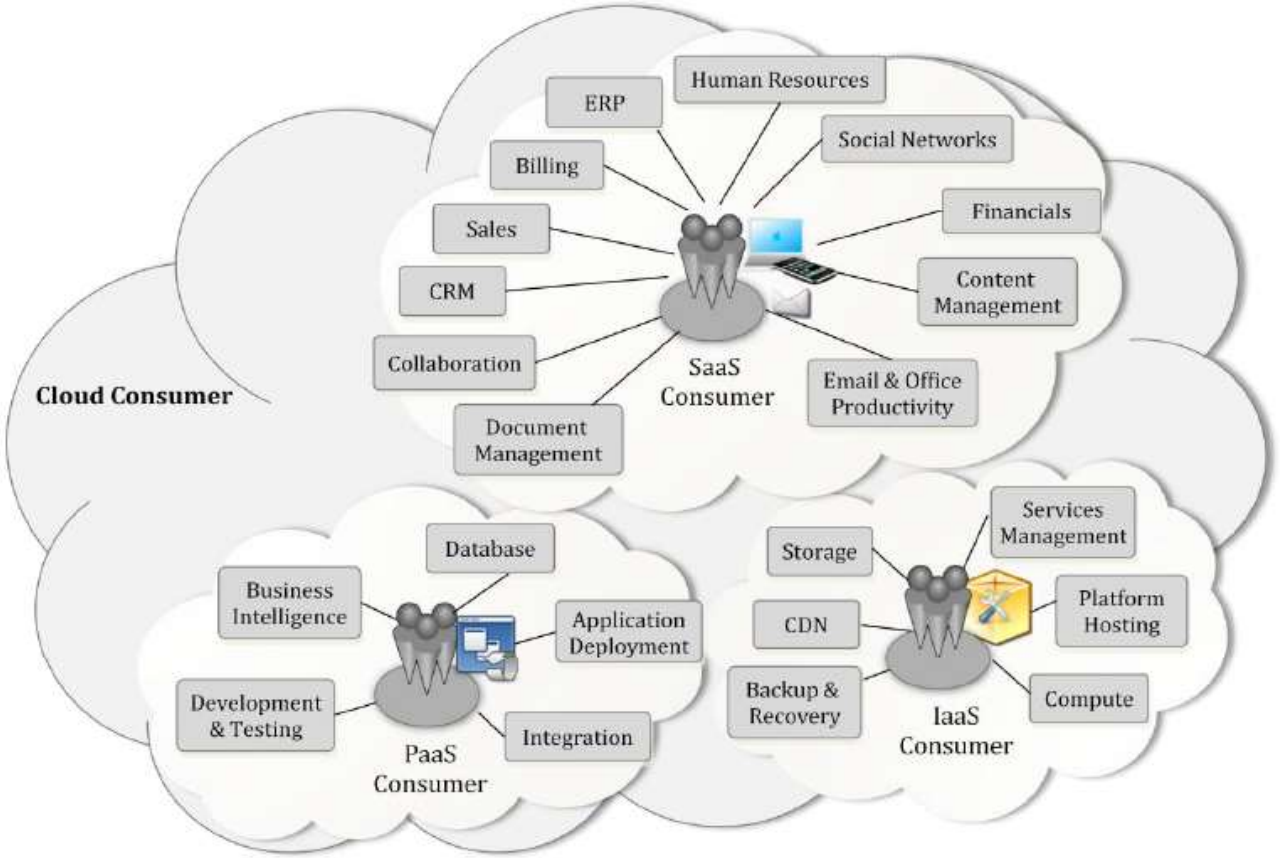
Cloud computing

Service Orchestration

- Service Orchestration refers to the *composition of system components* to support the Cloud Providers' activities in arrangement, coordination and management of computing resources in order to provide cloud services to Cloud Consumers
- There are three forms of services in *service layer*
  - ✓ **IaaS (Infrastructure-as-a Service):** Providing storage, software, and network online in the cloud
  - ✓ **PaaS (Platform-as-a Service):** Access to languages, libraries, APIs, and services
  - ✓ **SaaS(Software-as-a Service)** offers customized applications
- **Resource abstraction and control layer** contains system components that Cloud Providers use to *provide and manage access* to the *physical computing resources through software abstraction*.
- **Physical resource layer** includes *all the physical computing resources* like computers, networks, storage device, etc.



Cloud Services



Cloud computing

1. Software as a Service (SaaS)

- The capability provided to the consumer to use the provider's applications running on a cloud infrastructure.
- The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g. web-based email), or a program interface.
- The consumer does not manage or control the underlying cloud infrastructure including network, servers and operating systems.
- Have control over only certain user-specific application configuration settings.
- The consumers of SaaS can be
  - ✓ Organizations that provide access to software applications,
  - ✓ End users who directly use software applications
  - ✓ Software application administrators who configure applications for end-users.
- SaaS consumers can be billed based on
  - ✓ The number of end users,
  - ✓ The time of use,
  - ✓ The network bandwidth consumed,
  - ✓ The amount of data stored

2. Platform as a Service (PaaS)

- The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider.
- The consumer does not manage or control the underlying cloud infrastructure but has control over the deployed applications and possibly configuration settings for the application-hosting environment.

- PaaS consumers can be
  - ✓ Application developers who design and implement application software,
  - ✓ Application testers who run and test applications in cloud-based environments,
  - ✓ Application deployers who publish applications into the cloud, and
  - ✓ Application administrators who configure and monitor application performance.

Cloud computing

- PaaS consumers can be billed according to
  - Processing,
  - Database storage and
  - Network resources

  consumed by the PaaS application

- The PaaS Cloud Provider typically also supports PaaS Cloud Consumer by providing tools such as:
  - Integrated development environments (IDEs),
  - Development version of cloud software,
  - Software development kits (SDKs), deployment and management tools.

3. Infrastructure as a Service (IaaS)

- The capability provided to the consumer is to provide processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications.
- The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls).
- The consumers of IaaS can be
  - ✓ System developers,
  - ✓ System administrators and
  - ✓ IT managers who are interested in creating, installing, managing and monitoring services for IT infrastructure operations.

Cloud computing

- IaaS consumers are provisioned with the capabilities to access these computing resources and are billed according to:
  - ✓ The amount or duration of the resources consumed, such as CPU hours used by virtual computers,
  - ✓ Volume and duration of data stored,
  - ✓ Network bandwidth consumed,
  - ✓ Number of IP addresses
- The Cloud Provider runs the cloud software necessary to makes computing resources available to the IaaS Cloud Consumer through a set of service interfaces and computing resource abstractions, such as virtual machines and virtual network interfaces.
- The IaaS Cloud Consumer in turn uses these computing resources, such as a virtual computer, for their fundamental computing needs Compared to SaaS and PaaS Cloud Consumers
- An IaaS Cloud Consumer has access to more fundamental forms of computing resources