

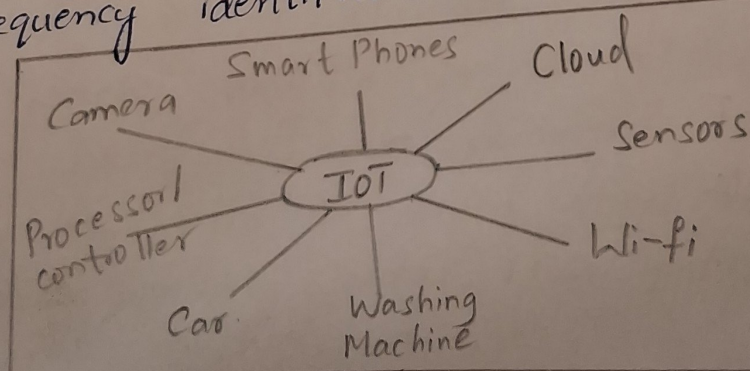
Unit - 1

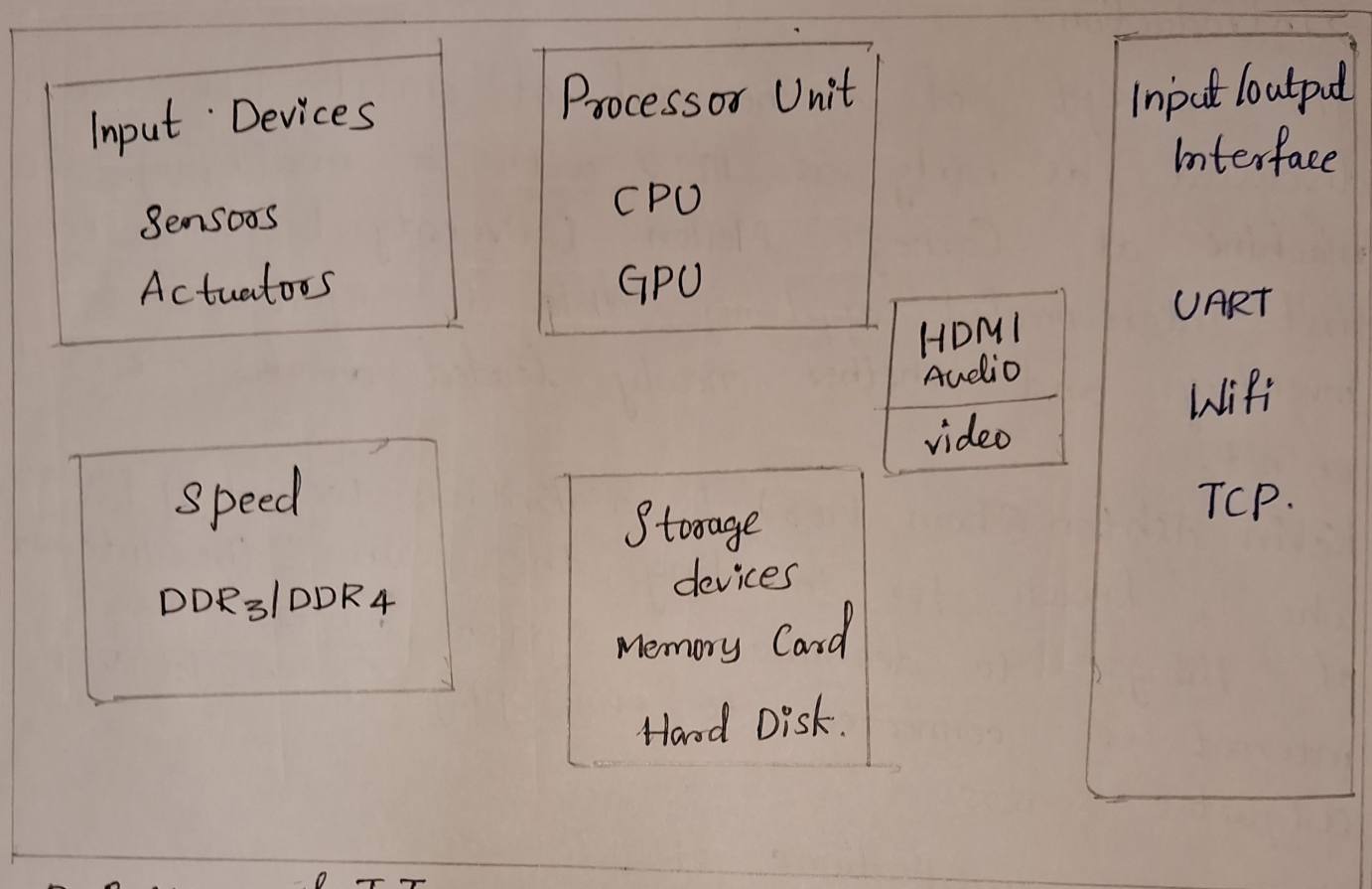
Introduction To Internet of Things (IoT)

⇒ Introduction:-

- The concept of a network of smart devices was discussed as early as 1982 with a modifying coke machine at Carnegie Mellon University becoming the first interconnected appliance, able to report its inventory and whether newly loaded drinks are cold or not.
- Kevin Ashton (born 1968) is a British technology pioneer who is known for inventing the term "the Internet of Things" to describe a system where the internet is connected to the physical world via ubiquitous sensors.
- The main applications of IoT are
 - i. Healthcare
 - ii. Transportation
 - iii. Home appliances
 - iv. Agriculture
 - v. Automotive Industries.
- The development of IoT involves many issues such as infrastructure, communications, interfaces, protocols and standards.

• The IoT concept was coined by a member of the Radio frequency identification (RFID) community in 1999.





→ Definition of IoT

It is the network of physical objects, - devices, instruments, vehicles, and other items embedded with electronics, circuits, software, sensors and network connectivity that enables these objects to collect and exchange data.

IoT refers to the interconnection via internet of computing devices embedded in everyday objects, enabled them to send and receive the data.

→ Physical design of IoT:-

The structure of an IoT can be divided into two parts

- i. Things in IoT
- ii. IoT protocols.

I. Things in IoT:-

- The "things in IoT" usually refers to IoT devices which have unique identities and can perform remote sensing, actuating and monitoring capabilities.

- IoT devices can exchange data with other connected devices and applications.

For example:-

Moisture sensor always monitors, the water level in the field that means whether the fields are wet or dry.

- An IoT device may consist of several interfaces/software/connectivity for connections to other devices, both wired and wireless.

These include

- i. I/O interfaces for sensors (Input and Output)
- ii. Interface for internet connectivities.
- iii. Memory and storage interfaces.
- iv. Audio or video interfaces.

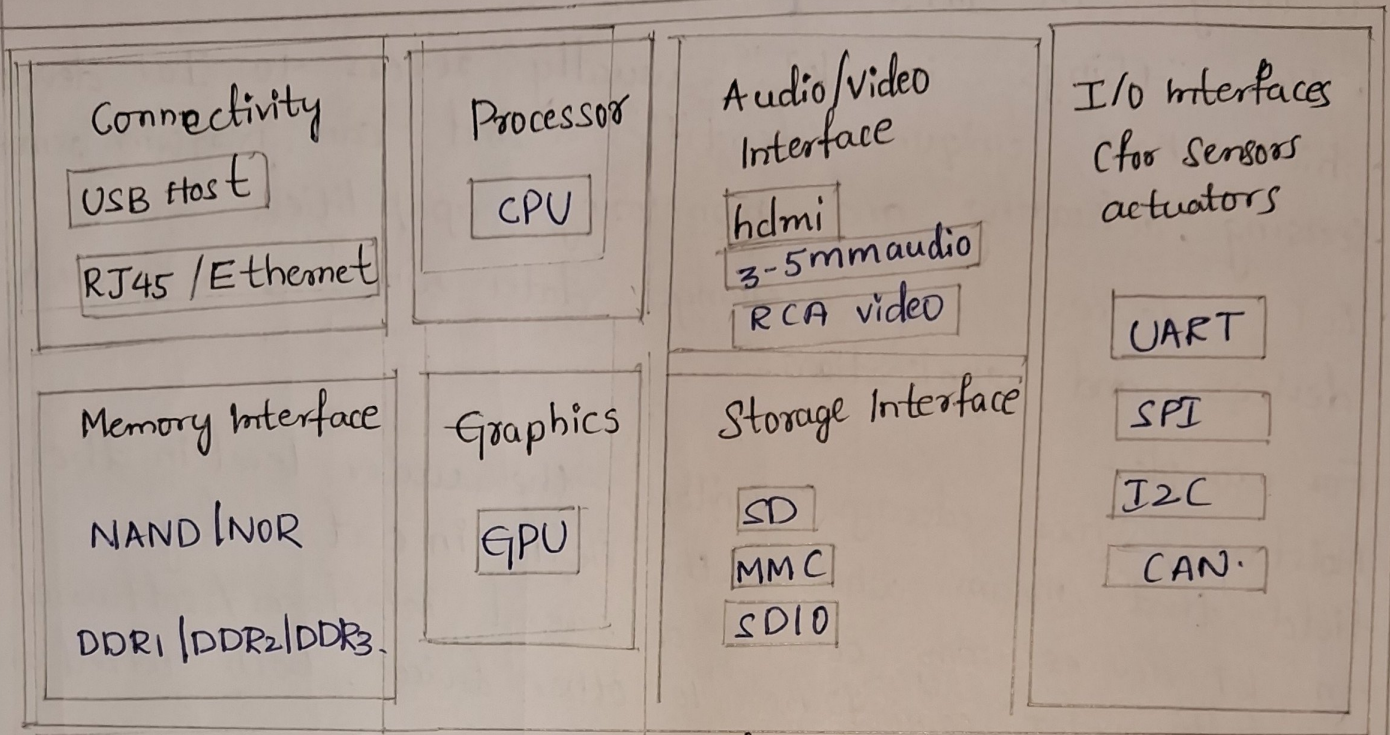
- An IoT device can collect various type of data from the onboard or attached sensors such as temperature, humidity, light intensity, ultra sonic sensors, etc.

The sensed data can be communicated either to other devices or cloud based server/storage.

- IoT devices can be connected to actuators that allow them to interact with other physical entities (non IoT devices) in the vicinity of the device.

Ex:- Relay switch connected to an IoT device can turn on/off based on the commands.

- IoT devices can also be varied types for instances, wearable sensors, smart watches, LED lights, automobile, and industrial machine.



— Generic Block Diagram of an IoT Device

ii. IoT Protocols:-

- The IoT devices are typically connected to the internet via IP (Internet Protocol) network.
- However devices such as Bluetooth and RFID allow IoT devices to connect locally.
- In these cases, there's a difference in power, range and memory used.
- Connection through IP networks are comparatively complex, required more memory and power from the IoT protocols/devices while the range is not a problem so IP networks are complex.
- On the otherhand non-IP networks demand comparatively power and memory but has a range limitation.
- As far as the IoT communication protocols or technologies are concerned, a mix of both IP and non-IP networks can be considered depending on usage.

Figure below shows the structure of IoT protocols. There are four layers in IoT protocols, they are.

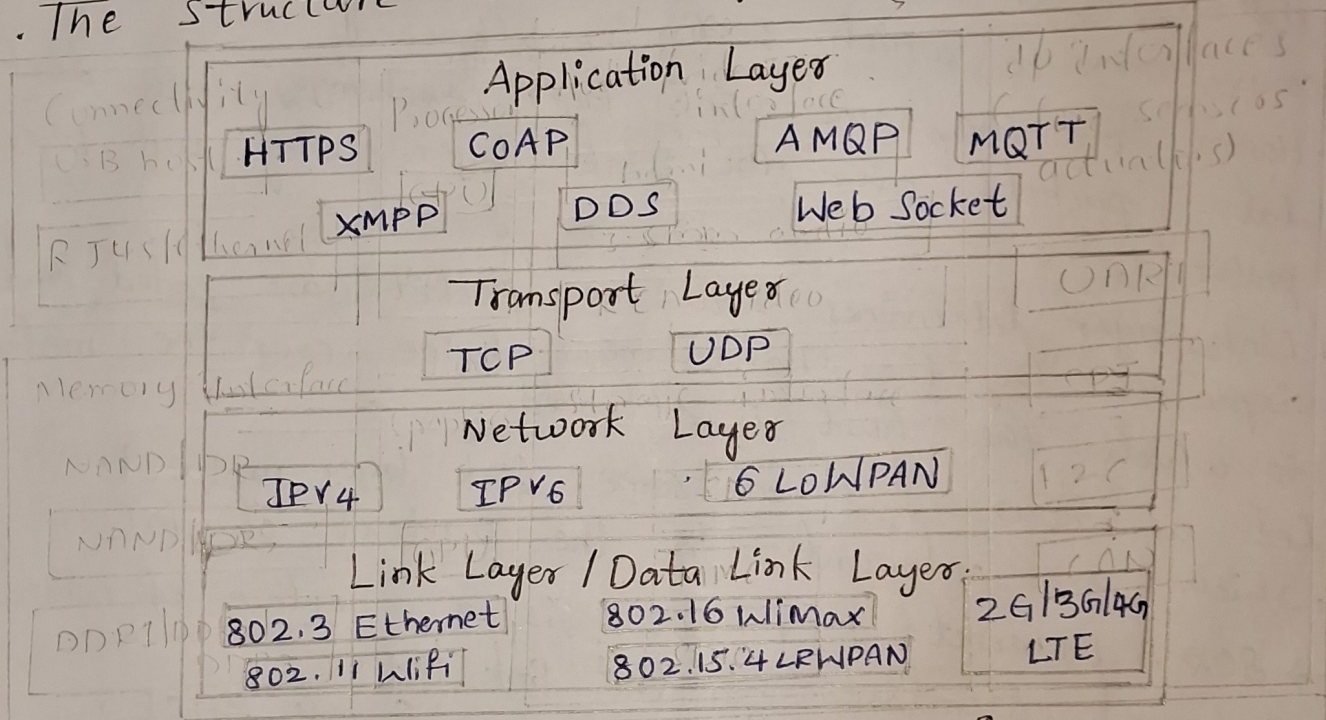
i. Link layer (Data Link Layer)

ii. Network layer

iii. Transport layer

iv. Application layer.

The structure of IoT protocols are shown below



Generic Block Diagram of an IoT Device

- ⊕ Link Layer:-
 - Link layer protocols determine how the data is physically sent over the network's physical layer or medium.
 - In computer network, the link layer is the lowest layer in the internet protocol suite, the networking architecture of the internet.
 - The link layer is the group of methods and communications protocols confined to the link that a host is physically connected to.
 - The link is the physical and logical network

component to interconnect hosts or nodes in the network and a link protocol is a suite of methods and standards that

- Link layer protocols determine how the data is physically sent over the network's physical layer or medium (e.g., copper wire, coaxial cable, or a radio wave).

- The scope of the link layer is the local network connection to which host is attached.

- Hosts on the same link layer exchange the data packets over the link layer using link layer protocols.

- Link layer determines how the packets are coded and signaled by the hardware device over the medium to which the host is attached.

- Let us now look at some link layer protocols which are relevant in the context of IoT.

→ 802.3 Ethernet: IEEE 802.3 is a collection of wired Ethernet standards for the link layer.

- For ex: 802.3 is the standard for 10 BASE5 Ethernet that uses coaxial cable as a shared medium.

- 802.3.i is the standards for 10 BASE-T ethernet over copper twisted pair connections.

- 802.3.j is the standard for 10BASE-F ethernet over fiber optic connections.

- 802.3 ae is the standard for 10G bits/s ethernet over fiber and so on.

- These standards provide data rates from 10 Mbps to 40 Gbps and higher.
- The shared medium in ethernet can be a coaxial cable twisted pair or an optical fiber.
- The shared medium carries the communication for all the devices on the network, thus data sent by one device can be received by all devices, subject to propagation conditions and transceiver capabilities.
- The specifications of the 802.3 standards are available on the IEEE 802.3 working group website.

i. 802.11 wifi:- IEEE 802.11 is a collection of wireless local area network communication standards including extensive description of the link layer.

Examples:-

- 802.11a operates in the 5GHz band.
- 802.11b & 802.11g operates in the 2.4GHz band.
- 802.11n operates in the 2.4/5GHz band.
- 802.11ac operates in the 5GHz band.
- 802.11ad operates in 60GHz band.
- These standards provide a data rates from 1 Mbps to upto 6.75 Gbps.

The specifications of the 802.11 standards are available on the IEEE 802.11 working group website.

ii. Mobile Communication for 2G/3G/4G:-

- There are different generations of mobile communications standards including 2G (GSM & CDMA), 3G (UMTS and CDMA 2000) and 4G (LTE).
- IoT devices based on these standards can

communicate over the cellular networks.
 Data rates for these standards range from 9.6 kbps (for 2G) to upto 100 mbps (for 4G) and are available from the 3GPP websites.

802.16

- * IEEE 802.16
- * Long distanced
- * Range 7km to 50km
- * Outdoor usage
- * 2.4 GHz & 5 GHz
- * 802.16, 16a, 16d, 16e, 16m
- * 100 mbps in a 20 MHz channel
- * More customers can be connected

802.11

- * IEEE 802.11 WLAN or wifi
- * limited area
- * 30m to 100m
- * Indoor usage.
- * 2.4 GHz, 3.5 GHz, 5.8 GHz
- * 802.11a, 11b, 11g, 11n, 11ac, 11ad etc.
- * 54 mbps in 20 MHz.
- * Limited.

S.No	Standard	Frequency	Speed.
1.	802.11	2.4 GHz	2 mbps
2.	802.11A	5 GHz	54 mbps
3.	802.11B	2.4 GHz	11 mbps
4.	802.11G	2.4 GHz	54 mbps
5.	802.11N	2.4 GHz / 5 GHz	600 mbps
6.	802.11ac	5 GHz	1300 mbps
7.	802.11ad	2.4 GHz / 5 GHz / 60 GHz	7 Gbps.

5
→ 802.16 - WiMax: IEEE 802.16 is a collection of wireless broadband standards, including extensive descriptions for the link layer (also called WiMax).

• WiMax standards provide data rates from 1.5 mbps to 1 Gbps. The recent update (802.16) provides data rates of 100 mbps for mobile stations and 1 Gbps for fixed stations.

• The specifications of the 802.11 standards are readily available on the IEEE 802.16 working group website.

• 802.16 is designed to deliver high speed internet access wirelessly, offering an alternative to traditional wired connection.

→ 802.15.4 - LR WPAN:-

• IEEE 802.15.4 is a collection of standards for low rate wireless personal area networks (LR WPANs).

• Low rate wireless personal area network.
• These standards form the basis of specification for high level communication protocols such as Zigbee.
• LR WPAN standards provide data rate from 40 kbps to 250 kbps.

• These standards provide low cost and low speed communication for power constrained devices.

• The specifications for the 802.15.4 standards are available on the IEEE 802.15 working group website.

• They are often used in applications like wireless sensor networks, hence automation and industrial control systems.

What is an IP Address:-

- Imagine every device on the internet as a house. For you to send a letter to a friend living in one of these houses, you need their house address.
- In this digital world, this house address is what we call an IP (Internet Protocol) address.
- It is a unique string of numbers separated by periods (IPv4) or colons (IPv6) that identifies each device connected to the internet or local network.

Types of IP Address:-

- IP address can be classified in two types.

IPv4 and IPv6

- IPv4:- This is the most common form of IP address.
- It consists of four sets of numbers separated by dots.

For Ex:- 192.168.141

Each set of numbers range from 0 to 255.

- The range is derived from the possible combinations of eight bits ($2^8 = 256$).

• Example of IPv4 address:- 192.168.1.1

- 192 is the first octet.
- 168 is the second octet.
- 1 is the third octet.
- 1 is the fourth octet.

• IPv6:- IPv6 addresses were created to deal with shortage of IPv4 address.

- They use 128 bits instead of 32, offering a vast greater number of possible addresses.
- These addresses are expressed in eight groups of

four hexadecimal digits, each group representing 16 bits. These groups are separated by colons.

Ex:- 2001:0db8:85a3:0000:0000:8a2e:0370:7334

Based on usage IP addresses classified into two types:-

Public IP and Private IP.

→ Public IP Address:-

• A public IP is assigned to every device that directly access the internet. This address is unique across the entire internet.

→ Private IP Address:- Home networks, office networks

• Internet protocol version 6 is the newest version of internet protocol and successor to IPv4.

• IPv6 uses 128-bit address scheme that allows total of 2^{128} (or) 3.4×10^{38} addresses.

→ 6LOWPAN:- 6LOWPAN (IPv6 over low power wireless personal area networks) brings IP protocol to the low power devices which have limited processing capability.

• 6LOWPAN operates in the 2.4 GHz frequency range and provide data transfer rates of 250 kbps.

• 6LOWPAN works with the 802.15.4 link layer protocol and defines compression mechanisms for IPv6 datagrams over IEEE 802.15.4-based networks.

IP address classifications:-

- 1) Class A (1.0.0.0 to 127.255.255.255) - (multinational companies)
- 2) Class B (128.0.0.0 to 191.255.255.255) - (organizations)
- 3) Class C (192.0.0.0 to 223.255.255.255) - small business or home networks.
- 4) Class D (224.0.0.0 to 239.255.255.255) - multicast groups.
- 5) Class E (240.0.0.0 to 255.255.255.255)

Reserved for experimental purposes and future use.

• Network / Internet Layers:-

• The network layers are responsible for sending of IP datagrams from the source networks to the destination network.

• This layer performs the host addressing and packet routing.

• The datagrams contain the source and destination address which are used to route them from the source to the destination across multiple networks.

• Host identification is done using hierarchical IP addressing schemas such as IPv4 and IPv6.

→ IPv4:- Internet Protocol version 4 (IPv4) is the most deployed internet protocol that is used to identify the devices on a network using a hierarchical addressing schema.

• IPv4 uses a 32-bit address scheme that allows total of 2^{32} or 4,294,967,296 addresses.

- As more and more devices got connected to the internet these addresses got exhausted in the year 2011.
- IPv4 has been succeeded by IPv6.
- The IP protocols establish connection on packet networks, but don't guarantee delivery of packets.
- Guaranteed delivery and data integrity are handled by the upper layer protocols (such as TCP).

→ Transport Layer:-

The transport layer protocols provide end-to-end message transfer capabilities independent of the underlying network.

The message transfer capability can be set up on connections either using handshakes (as in TCP) or without handshakes/acknowledgements (as in UDP).

The transport layer provides functions such as error control, segmentation, flow control and congestion control.

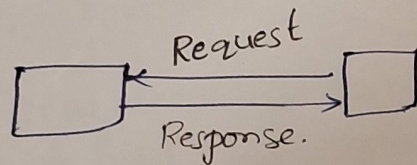
→ TCP:- Transmission Control Protocol is the most widely used transport layer protocol, that is used by web browsers (along with HTTP, HTTPS application layer protocols) email programs (SMTP application layer protocol) and file transfer protocol (FTP).

- TCP is a connection oriented and stateful protocol, while IP protocol deals with sending packets, TCP ensures reliable transmission of packets in-order.
- TCP also provides error detection capability so that duplicate packets can be discarded and lost packets are retransmitted.

- The flow control capability of TCP ensures that rate at which the sender sends the data is not too high for the receiver to process.
- It is a connection-oriented protocol for communications that helps in the exchange of messages between different devices over a network.
- It is one of the main protocols of the TCP/IP suite.
- It lies between the application layer and network layers which are used in providing reliable delivery services.
- The IP which establishes the technique for sending data packets between computers, works with TCP.
- * TCP establishes a reliable connection between sender and receiver using the three way handshakes (SYN, SYN-ACK, ACK) and it uses a four step handshake (FIN, ACK, FIN, ACK) to close the connections properly.
- * It ensures error free, in order delivery of data packets.
- * It uses acknowledgements (ACKs) to confirm receipt.
- * It prevents data overflow by adjusting the data transmission rate according to the receiver's buffer size.
- The congestion control capability of TCP helps in avoiding network congestion and congestion collapse which lead to degradation of network performance.

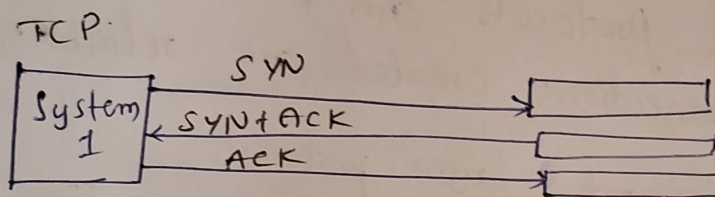
→ UDP:-

- Unlike TCP, which requires carrying out an initial setup procedure, UDP is a connectionless protocol.
- UDP is useful for time-sensitive applications that have very small data units to exchange and do not want the overhead of connection setup.
- UDP is a transaction oriented and stateless protocol.
- UDP does not provide guaranteed delivery, ordering of messages and duplicate elimination.
- Higher levels of protocols can ensure reliable delivery or ensuring connections created are reliable.
- UDP is a transport layer protocol.
- Unlike TCP, it is an unreliable and connection less protocol.
- So there is no need to establish a connection before data transfer.
- The UDP helps to establish to low latency and low tolerating connection over network.
- The UDP enables process to process communication.



- User Datagram Protocol (UDP) is one of the core protocols of the IP suite.
- It is a communication protocol used across the internet for time sensitive transmission such as video playback.
- Unlike TCP, UDP is connectionless and does not guarantee delivery, order or error checking, making it a light weight and efficient option for certain types of data transmission.

- * It prevents network congestion using algorithms like slow start, congestion avoidance, fast retransmit, and fast recovery.
- * TCP header uses checksum to detect corrupted data and requests retransmission if needed.
- * It is used in applications requiring reliable and ordered data transfer, such as web browsing, email and remote login.



For ex: When an user requests a web page on the internet, somewhere in the world, the server processes that request and sends back an HTML page to that user.

Features of TCP:-

- 1) Segment Numbering System.
- 2) Connection Oriented.
- 3) Full duplex
- 4) Flow Control
- 5) Congestion Control.

Application Layer:- Application layer protocols define how the applications interface with lower layer protocols to send the data over the network.

- The application data, typically in files, is encoded by the application layer protocol and encapsulated in the transport layer protocol which provides connection or transaction oriented communication over the network.
- Port numbers are used for application addressing (for ex port 80 for HTTP, port 22 for SSH etc).
- Application layer protocols enable process to process connections using ports.

i) **HTTP:-** Hypertext Transfer Protocol (HTTP) is the application layer protocol that forms the foundation of the World Wide Web (WWW).

- HTTP includes commands such as GET, PUT, POST, DELETE, HEAD, TRACE, OPTIONS, etc.

The protocol follows a request-response model where a client send requests to a server using the HTTP commands.

- HTTP is a stateless protocol and each HTTP request is independent of the other requests.
- An HTTP client can be a browser or an application running on the client.

ii. **CoAP:-**

- The full form of CoAP is Constrained Application Protocol. It is an application layer protocol for machine to machine (M2M) applications, meant for constrained environments with constrained devices and

constrained networks.

- Like HTTP, CoAP is a web transfer protocol and uses a request-response model, however it runs on top of UDP instead of TCP.

- CoAP uses a client server architecture where clients communicate with servers using connectionless datagrams. data graphs, CoAP is designed to easily interface with HTTP.

- Like HTTP, CoAP supports methods such as GET, PUT, POST and DELETE.

- CoAP draft specifications are available on IETF constrained environment (CORE) working group website.

iii. Web Socket:-

- Websocket protocol allows full duplex communication over a single socket connection for sending messages between client and server.

- Websocket is based on TCP and allows streams of messages to be sent back and forth between the client and server while keeping TCP connection open.

- The client can be a browser, a mobile application or an IoT device.

iv. MQTT:-

- MQTT stands for Message Queue Telemetry Transport (MQTT).

- MQTT is a light weight messaging protocol based on the publish subscribe model.

- MQTT uses a client server architecture when the client (such as an IoT device) connects to the server (also called MQTT Broker) and publishes messages to topics on the server.
- The broker forwards the messages to the clients subscribed to topics.
- MQTT is well suited for constrained environments when the devices have limited processing and memory resources and network bandwidth is low.

V. ~~X~~XMPP:-

- Extensible Messaging and Presence Protocol (XMPP) is a protocol for real time communication and streaming XML data between network entities.
- XMPP powers wide range of applications including messaging, presence, data syndication, gaming, multiparty chat and voice/video calls.
- XMPP allows sending small chunks of XML data from one network entity to another in near real time.
- XMPP is a decentralized protocol and uses client server architecture.
- XMPP supports both client to server and server to server communication paths.
- In the context of IoT, XMPP allows near real time communication between IoT devices.

$\Rightarrow$ MQTT:-

- Message Queuing Telemetry Transport.
- It is an application layer protocol.
- It is used for M2M and IOT connectivity (Machine to Machine).
- It is a publish-subscribe-based messaging protocol that transports messages between devices.
- It is very light weight and thus suited for M2M (Mobile to Mobile), (WSN - Wireless Sensor Network), and IOT sensors, where communication nodes communicate with applications through the MQTT message broker.
- It usually over TCP/IP protocol.
- MQTT was developed by IBM and Eurotech.
- MQTT is designed for limited devices, high latency, and low BW communication.

• MQTT Components:-

(ii) Broker - Connect publisher and subscribers/Classify sensor data into topics.

ii. Topic

iii. Message

iv. Client → Publisher — Light weight sensors.
→ Subscriber — Applications int

② MQTT Client:- A client can be either the publisher or the subscriber.

- Subscribers can subscribe to topics and receive messages.
- Publishers can publish the topics.

b. MQTT Broker:-

- It is a central point of communication.
- It is responsible for dispatching all messages between clients.
- It receives subscribe requests from client on topics and based on subscriptions it forwards the message to clients.

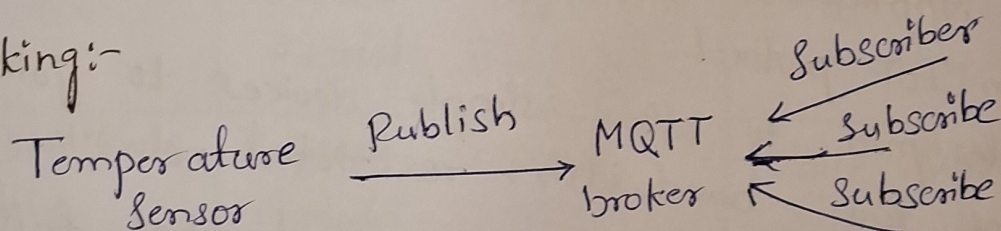
c. Topic:-

- It is an identifier for subscribers (Temperature, pressure, humidity, gases properties)

d. Message:-

- Subscriber receives messages based on topics of subscriber from broker.

Working:-



Step 1:- Publishers can publish topics.

Step 2:- Subscribers can subscribe to one or more topics.

Step 3:- Publisher gives messages on topics to MQTT broker.

Step 4:- MQTT broker publishes the message to subscribed users.

• There are 5 methods in MQTT:-

- i. Connect
- ii. Disconnect
- iii. Subscribe
- iv. Unsubscribe
- v. Publish.

→ Advantages:-

- It is a simple protocol for IoT
- Low network usage.
- Lower data rate.

→ Applications:-

- Home, factory, medical, healthcare.
- Facebook messages.
- Amazon webserver.
- Microsoft Azure.
- Every thing on IoT platform
- Adafruit.

⇒ CoAP:-

- Constrained Application Protocol.
- CoAP is an IoT protocol.
- CoAP is designed to allow simple devices to join IoT through the low bandwidth - restricted network.
- CoAP is an application layer protocol that follows the request-response model.
- CoAP runs over UDP protocol.
- CoAP is designed for M2M and IoT application.
- CoAP uses fewer resources than HTTP.
- CoAP client can use GET, PUT, and DELETE commands during.
- Web transfer protocol for use with constrained nodes and network.
- Based on Request Response model between end points.

- CoAP is a session layer protocol designed IETF.
- CoAP is designed to enable low power sensor to use REST (Representational State Transfer) services while meeting their power constraints.
- CoAP has four messaging modes.
 - 1) Confirmable.
 - 2) Non-Confirmable.
 - 3) Piggy back.
 - 4) Seperated.

⇒ CoAP vs MQTT

Parameters	CoAP	MQTT
i. Full Form	Constrained Application Protocol	Message Queuing Telemetry Transport.
ii. Architecture Model.	Request Response Model	Public Subscribe Model
iii. Messaging Mode.	Asynchronous and Synchronous	Asynchronously.
iv. Communication Mode	One to one	Many to many.
v. Transport Layer.	Mainly UDP	Mainly TCP
vi. Header size	4 Byte	2 Byte
vii. Message Labelling.	Yes (2 byte)	No
viii. Low effectiveness	Excellent	Low.

ix. Sorting of Data.	URI (Universe Resource Identifiers)	Broker topic.
x. Port used	61831	Secure 8883, Non secure 1883.
xi. Persistent Support	No support	It supports.

⇒ XMPP:-

- Extensible Messaging & presence protocol.
- A communication protocol for message oriented middle.
- For real time, chat, online or mobile, real time communication with satellite.
- XMPP is based on XML.
- XML stands for Extensible Markup Language.
- XML is like HTML,
- XML is used for data sharing and storing, transfer.
- XML is platform independent.
- XMPP uses client-server architecture.
- XMPP is open standard protocol.
- As the mode is decentralized, anyone can run their own XMPP server.
- It is well-suited for cloud computing.
- It is secure and flexible.

→ Applications:-

1. Video, file transfer, gaming.
2. IoT - Smart Grid, social networking services.
3. Publish subscribe systems.

points.

12
⇒ XMPP:-

* It is a protocol (standard) that is used to build chat systems.

→ Features:-

- Peer to peer sessions:- XMPP supports machine to machine (or) peer to peer across diverse set of networks.
- Multiuser chat:- XMPP supports group and conference chat.
- Encryption:- Point to point encryption is done by TLS (Transport Layer Security) and OTR (Off the Record Messaging) is an extension of XMPP that enables encryption of messages and data.
- Connection to other protocols:- Using XMPP Gateways other protocols like SMS and SMTP can also be connected with XMPP servers for different services.

→ Uses:-

- Real time webchat
- Instant messaging
- Real time group chat.
- IoT device control
- Online gaming.
- VoIP.
- Geo Location.
- System Control
- Push notifications.

⇒ AMQP

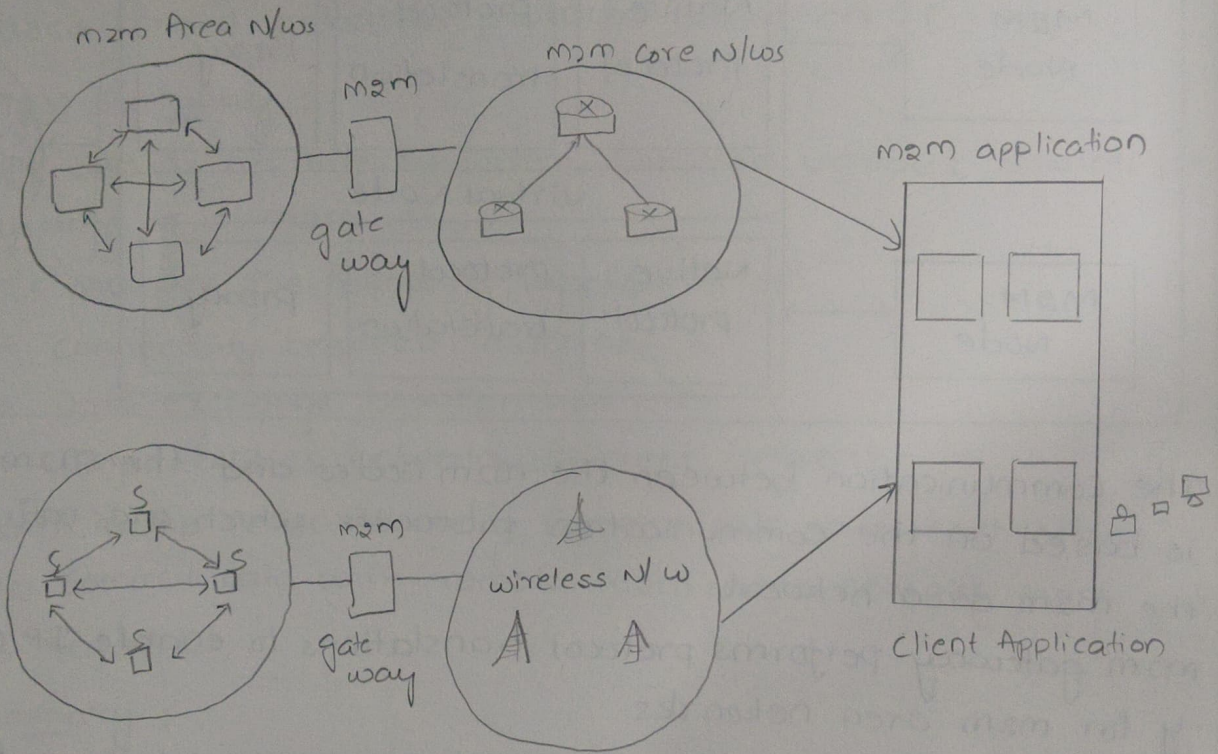
- AMQP stands for Advanced message Queuing Protocol.
- It is an open standard application layer protocol for message oriented middleware.
- AMQP is mainly used in transaction messages between servers and users.
- Mostly it is used in banking servers and banking apps.
- It uses TCP for reliable communication.
- End points acknowledge acceptance of each message.
- It follows public subscriber model for communication.
- It is an open application layer protocol for business messaging.
- AMQP supports both point to point and publisher/ subscriber models, routing and queuing.
- AMQP brokers receive messages from publishers and route them over connections to consumers.
- Publishers publish the messages to exchanges which then distribute message copies to queues.
- Messages are either delivered by the broker to the consumers which have subscribed to the queues or the consumers can pull the messages from the queues.

→ DDS:-

- DDS stands for Distribution Service (DDS). is
- It is data centric middleware standard for device to device or machine to machine communication.
- DDS uses a publish-subscribe model where publishers.
- Create topics to which subscribers can subscribe.
- Publisher is an object responsible for data distribution and the subscriber is responsible for receiving published data.
- DDS provides quality of service (QoS) control and configurable reliability.

Machine to Machine (M2M) refers to networking of machines (or) device for the purpose of remote monitoring and control and data exchange.

Fig below shows the end-to-end architecture for m2m Systems comprising of m2m area networks, Communication network and application domain

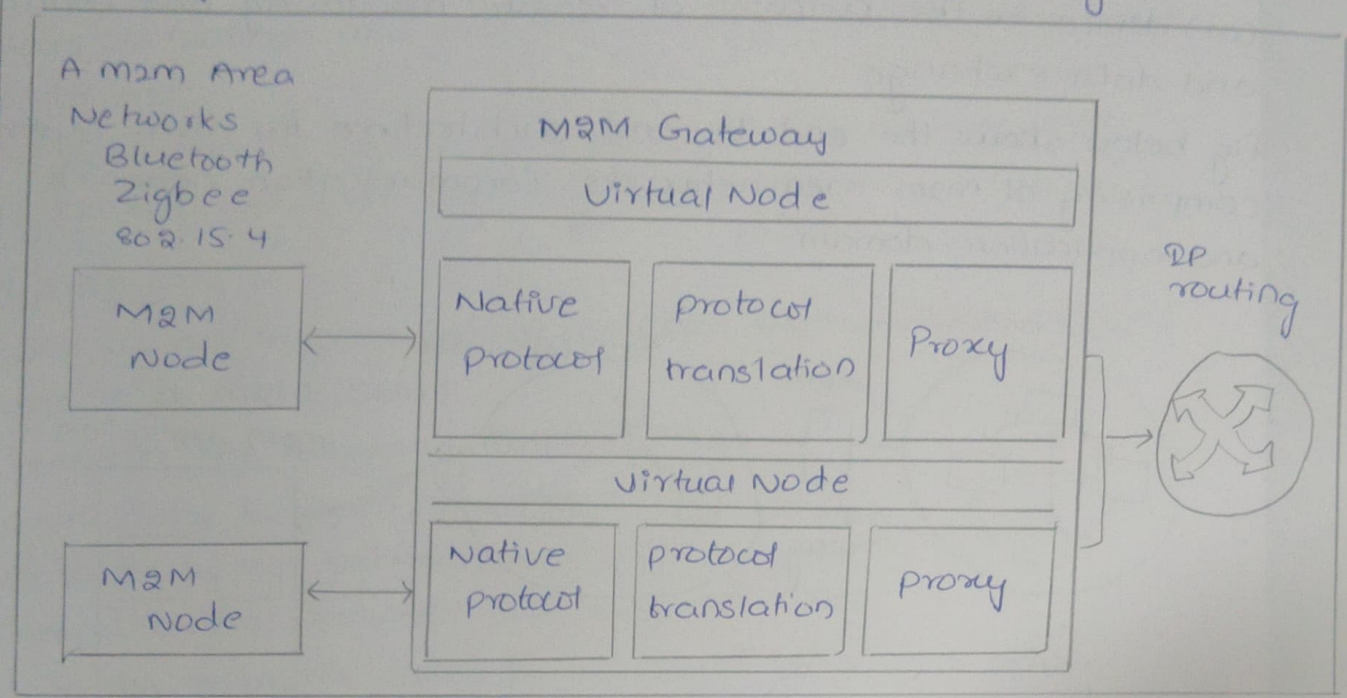


* * M2M System Architecture * *

An m2m are network comprises of machines which have embedded hardware modules for Sensing, actuation and Communication. Protocols used in m2m are ZigBee, Bluetooth, Mod Bus, M-Bus, wireless m Bus, power line Communication, 6LOWPAN, IEEE 802.15.4 etc.

The Communication N/w can be either wired or wireless network. To enable the Communication between remote m2m area networks, m2m gateways are used.

below fig shows block diagram of an m2m gateway.



The Communication between the m2m nodes and the m2m gateway is based on the Communication protocols which are native to the m2m area networks.

m2m gateway performs protocol translations to enable IP Connectivity for m2m area networks.

M2m gateway acts as a proxy performing translations from I to protocols to I from IP.

M2M has various application domains such as Smart metering, home automation, industrial automation, Smart grids etc.

Difference between IOT and M2M :-

Through both m2m and IOT involve networks of machines or devices, they differ in the underlying technologies, Systems architectures and types of application.

- * provide Remote Access: IOT and m2m provide access to information without human intervention
- * M2M : M2M provides direct communication between individual machine or devices. It is designed to communicate between

devices for a Specific purpose

* IOT : IOT is a broader concept for interest communication between devices. It involves a wide range of devices, sensors actuators, and applications that communicate with the internet

* M2M uses non IP based proprietary networks and IOT uses broad networks protocol based on IP

M2M Gateway :-

It sets as an intermediary between various devices and systems in a m2m setup

* It provides the following functionalities

- Connectivity between machines
- Data exchange between machines
- Compatibilities between different networks

* within the networks nodes communicate with each other

* To communicate with remote m2m area network, m2m gateway is required

Summary :-

* M2M is fundamental component of IOT

* M2M is the Backbone of IOT applications

Difference between M2M and IOT :-

1. communication protocols :-

M2M users either proprietary or non IP based communication protocols like Zigbee, Bluetooth, ModBns, PLC, 6LOWPAN.

M2M protocols usually on the below network layers IOT protocols are on above network layer such as HTTP, CoAP, websocket, MQTT, XMPP, DDS, AMQP

2. Machines in m2m vs things in IOT :

Things in IOT refers to physical objects that have unique identities and can sense and communicate with their

external environment

The unique identifiers for the things in IOT are the IP address (or) MAC address

M2M Systems in contrast to IOT, typically have homogeneous machine types within an M2M area network.

3. Hardware and Software Emphasis:

M2M is based hardware with embedded module, IOT is more on Software.

IOT devices run specialized Software for Sensor data collector, data analysis and interfacing with the cloud through IP based Communication.

Data Collection and Analysis: M2M data is collected in point solutions and often in on premises Software infrastructure. The data in IOT is collected in the cloud.

Application :

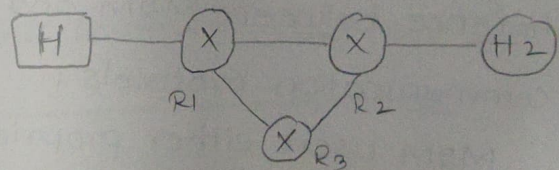
M2M: diagnosis applications, Servicemanagement, on premises enterprise application.

IOT: enterprise applications, remote diagnosis, management applications.

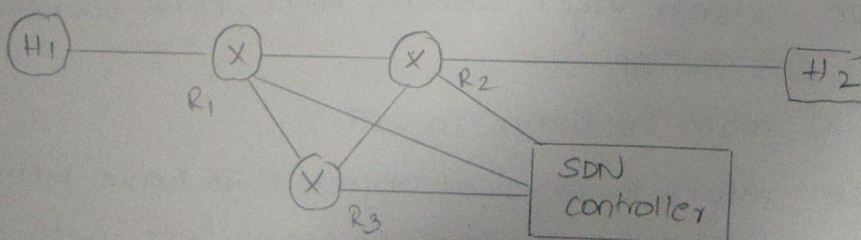
SDN - Software Defined Networking :

Traditional network :-

Routers will take the decision of routing data



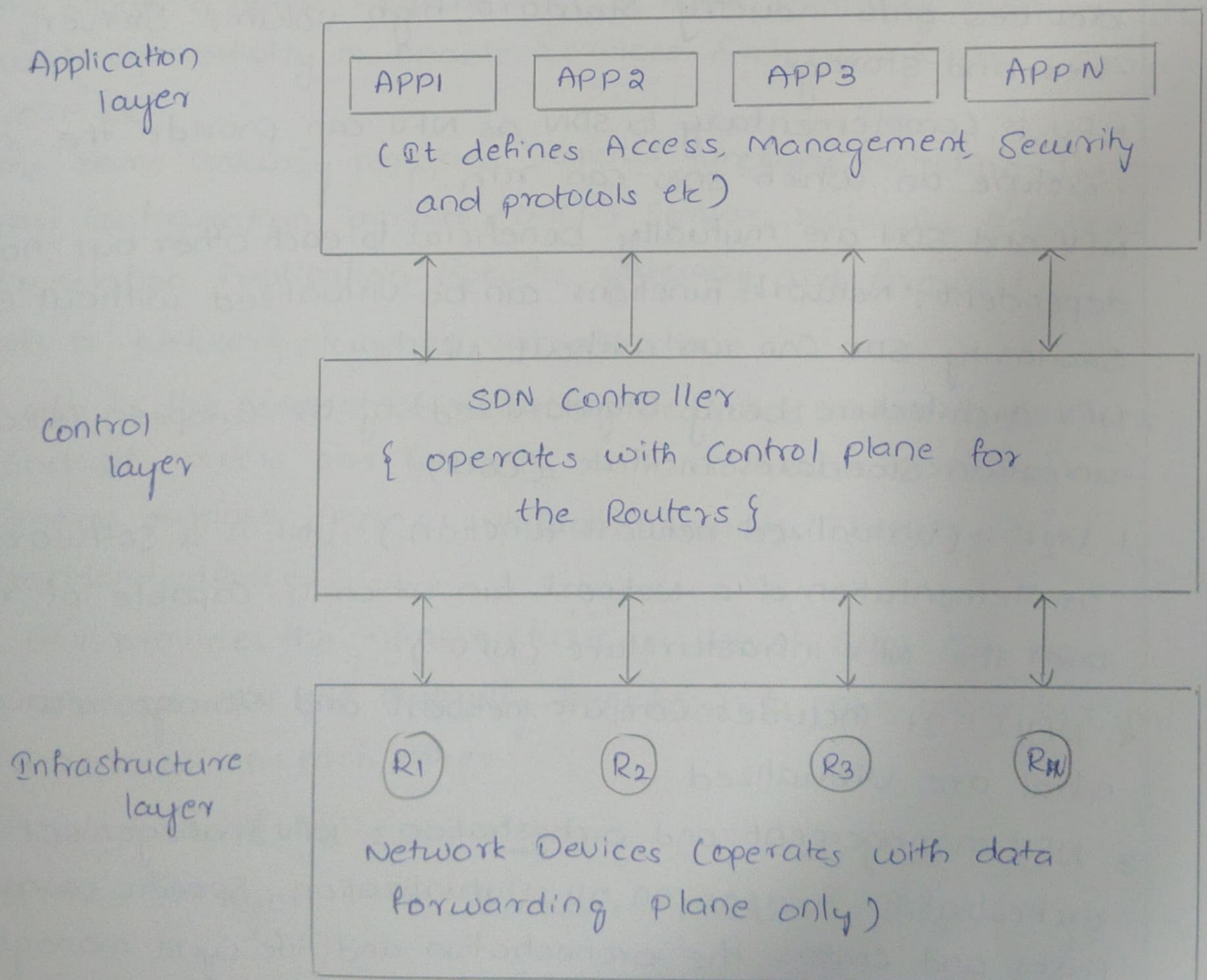
1. 2 planes
- Control plane - for Computation task
 - Data plane - Data forwarding



SDN Basics :-

- It Simplifies data Communication in the network (IoT, cloud, computer networks, NFV etc).
- SDN Simply removes the Control plane of task of the routers
- Routing decisions are now taken care by the SDN Controllers in the SDN
- SDN Controllers maintain a unified view of the network and make configuration, management and provisioning simpler

SDN Architecture :-



SDN Benefits:-

1. easier management of the network
2. Programmable network
3. Centralized management of the network
4. Agility
5. Improved visibility of the network
6. Cost effective network
7. Improved Security of the network

Network function virtualization is a technology that leverages virtualization to consolidate the heterogeneous network devices onto industry standard high volume Server, Switches and storage.

NFV is Complementary to SDN as NFV can provide the infrastructure on which SDN can run.

NFV and SDN are mutually beneficial to each other but not dependent, Network Functions can be virtualized without SDN. Similarly, SDN Can run without NFV

NFV architecture being Standardized by the European Telecommunication Standards Institute (ETSI)

1. VNF :- (virtualized Network function): VNF is a Software implementation of a Network fun which is Capable of running over the NFV infrastructure (NFVI).
2. NFVI: It includes Compute, network and Storage resources that are virtualized
3. NFV management and orchestration :- NFV management and orchestration focuses on all virtualization - Specific management tasks and covers the orchestration and life cycle manager of physical / Software resources that Support the infrastructure virtualization, and the life cycle management of VNFs

4

NFV Comprises of network functions implemented in software that run on virtualized resources in the cloud

NFV enables Separation of network functions which are implemented in software from the underlying hardware

Thus network functions can be easily tested and upgraded by installing new software while the network remains the same. Virtualizing network functions reduces the equipment costs and also reduces power consumption.

The multi-tenanted nature of the cloud allows virtualized network function to be shared for multiple network services.

NFV is applicable only to data plane and control plane functions in fixed and mobile networks.

Fig shows a home network with a Home Gateway that provides WAN connectivity to enable services such as internet, IPTV, VoIP etc.

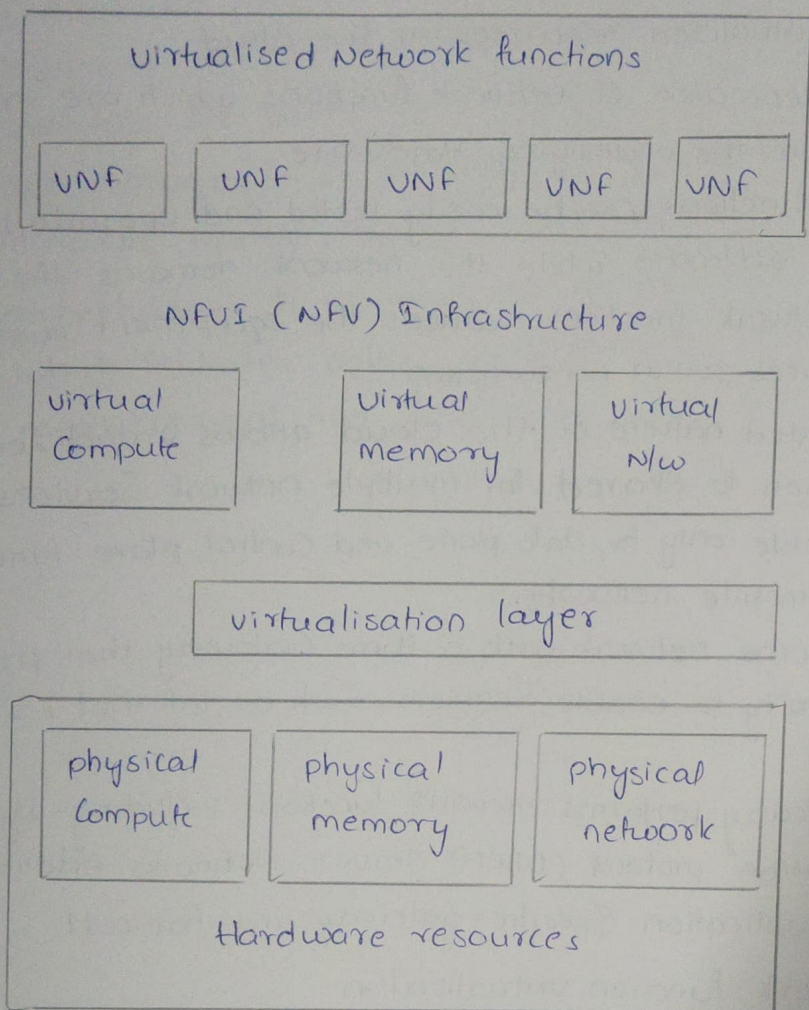
The Home Gateway performs various functions including - Dynamic Host Configuration protocol (DHCP) Server, Network Address Translation, Application Specific gateway and Firewall.

NFV is Network-Function Virtualization

1. NFV is the concept of replacing dedicated network devices such as routers and firewalls with software running on general purpose CPUs or virtual machines operating on standard devices.
2. NFV provides the infrastructure on which SDN can run.
3. NFV and SDN are mutually beneficial to each other but not dependent on each other.

Architecture of NFV :-

- It has 3 layers :
1. Hardware resources layer
 2. Virtualization layer
 3. virtual layer component.



NFVI :

First layer → hardware resources (CPU), Storage resources (Harddisk) and Network resources (Router, Switch and firewalls)

Second layer → it separates between and replaces it with software

Third layer → virtual Computers, virtual Networks and virtual storage

VNF :

- * It is a software implementation of a network function
- * It is capable of running over the NFV,
- * example: Vfirewall, VRouter

NFV management and orchestration :

- * virtualized infrastructure manager - It controls and manages network functions with NFVI resources and monitors the virtualization layer
- * VNF manager :- It manages the life cycle of VNF such as initialize, update, query, scale terminate etc.
- * orchestrator :- It manages the life cycle of network services which includes policy management, performance, measurement and monitoring.

Need for IOT systems management :-

• Internet of Things (IOT) systems can have complex software, hardware and deployment designs including sensors, actuators, software and network resources, data collection and analysis services and user interfaces. IOT systems can have distributed deployments comprising of a number of IOT devices which collect data from sensors or perform actuation. Managing multiple devices within a single system requires advanced management capabilities. The need for managing IOT systems is described as follows:

• **Automating configuration :** IOT system management interfaces provides predictable and easy to use management capabilities and the ability to automate system configuration. Automation becomes even more important when the system consists of multiple devices or nodes.

In such cases automating the system configuration ensures that all devices have same configurations and variations or errors due to manual configurations are avoided.

• **Monitoring Operational & Statistical Data :** Operational data is the data which is related to the system's operating parameters and is collected by the system at runtime. Statistical data is the data which describes the system performance (e.g. CPU and memory usage). Management systems can help in monitoring operational and statistical data of a system. This data can be used for fault diagnosis.

• **Improved Reliability :** A management system that allows validating the system configurations before they are put into effect can help in improving the system reliability.

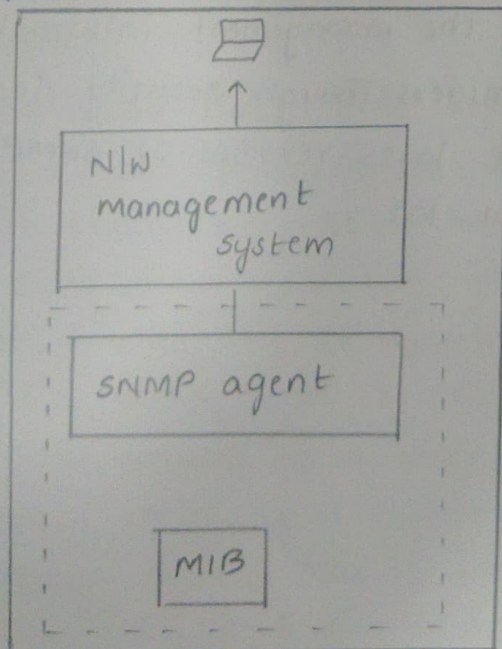
• **System wide configuration :** For IOT systems that consist of multiple devices or nodes, ensuring system wide configuration can be critical for the correct

-functioning of the system.

- management approaches in which each device is configured separately can result in system faults or undesirable outcomes. This happens when some devices are running on an old configuration while others start running on new configuration.
- To avoid this, system wide configuration is required where all devices are configured in a single atomic transaction. This ensures that the configuration changes are either applied to all devices or to none. In the event of a failure in applying the configuration to one or more devices, the configuration changes are rolled back. This "all or nothing" approach ensures that the system works as expected.
- Multiple system configurations: For some systems it may be desirable to have multiple valid configuration which are applied at different times or in certain conditions.
- Retrieving & Reusing configurations: management systems which have the capability of retrieving configurations from devices can help in reusing the configurations for other devices of the same type. For example, for an IoT system which has multiple devices and requires some configurations for all devices, it is important to ensure that when a new device is added, the same configuration is applied. For each case, the management system can retrieve the current configuration from a device and apply the same to the new devices.

Simple network management protocol :

- SNMP is a well-known and widely used network management protocol that allows monitoring and configuring network devices such as routers, switches, servers, hubs, bridges, Gateway, ~~no~~ printers etc.
- Fig below shows the component of the entities involved in managing a device with SNMP, including the Network management station (NMS), managed devices, management information Base (MIB) and SNMP agent that runs on the device.
- The managed device contains the MIB which has all the information of the device attributes to be managed. MIBs use the structure of management information (SMI) notation for defining the structure of the management data.
- The structure of the management data is defined in the form of variables which are identified by Object Identifiers (OIDs), which have a hierarchical structure. Management applications can either get or set the values of these variables. SNMP is an application layer protocol that uses UDP as the transport protocol.



Limitations:

1. SNMP is a stateless in nature and each SNMP request contains all the information to process the request. The application needs to be intelligent to manage the device. For a sequence of SNMP interactions, the applications needs to roll back the device into a consistent state in case of errors or failures in configuration.
2. SNMP is a connectionless protocol which uses UDP as the transport protocol, making it unreliable as there was no support for acknowledgement of requests.
3. MIBs often lack writable objects without which device configuration is not possible using SNMP with the absence of writable objects, SNMP can be used only for device monitoring and status polling.
4. It is difficult to differentiate between configuration and state data in MIBs.
5. Retrieving the current configuration from a device can be difficult with SNMP, SNMP does not support easy retrieval and playback of configurations.
6. Earlier versions of SNMP did not have strong security features making the management information vulnerable to network intruders. Though security features were added to in the later versions of SNMP, it increased the complexity a lot.

Network Operator Requirements:

To Address the limitations of the existing Network management protocols and plan the future work on n/w management, the Internet Architecture Board (IAB), which oversees the Internet Engineering Task Force led a workshop on n/w management in 2002 that brought together network operators and protocol developers. Based on the inputs from operators, a list of operator requirements was prepared. The following points provide a brief overview of the operator requirements:

1. Ease of use: From the operators point of view, ease of use is the key requirement for any network management technology.
2. Distinction b/w configuration and state data: Configuration data is the set of writable data that is required to transform the system from its initial state to its current state. state data is the data which is not configurable. state data includes operational data which is collected by the system at runtime, and statistical data which describes the system performance.
3. Fetch configuration and state data separately: In addition to making a clear distinction between configuration and state data, it should be possible to fetch the configuration and state data separately from the managed device. This is useful when the configuration and state data from different devices needs to be compared.
4. configuration of the network as a whole: It should be possible for operators to configure the network as a whole rather than individual devices. This is important for systems which have multiple devices and configuring them

14-0
Within one n/w wide transaction is required to ensure the correct operation of the system.

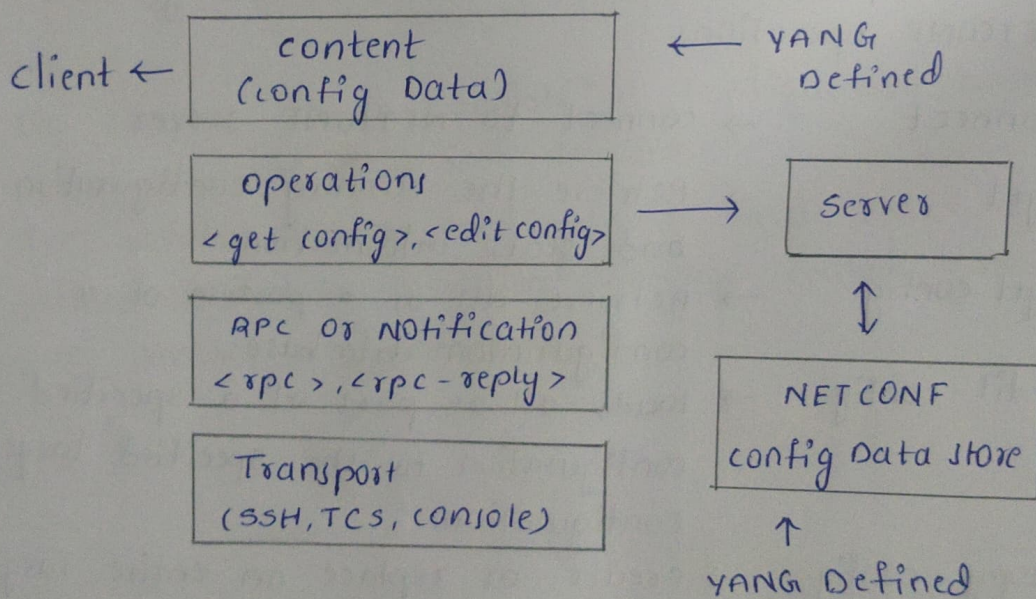
5. configuration transactions across devices: configuration transactions across multiple devices should be supported.
6. configuration deltas: It should be possible to generate the operations necessary for going from one configuration state to another. The devices should support configuration deltas with minimal state changes.
7. Dump and restore configurations: It should be possible to dump configurations from devices and restore configurations to devices.
8. configuration validation: It should be possible to validate configurations.
9. configuration database schemas: There is a need for standardized configuration database schemas or data models across operators.
10. comparing configurations: Devices should not arbitrarily reorder data, so that it is possible to use text processing tools such as diff to compare configurations.
11. Role-based access control: Devices should support role-based access control model, so that a user is given the minimum access necessary to perform a required task.
12. consistency of access control lists: It should be possible to do consistency checks of access control lists across devices.
13. multiple configuration sets: There should be support for multiple configurations sets on devices. This way a distinction can be provided between candidate and active configurations.

14. support for both data-oriented and task-oriented access control;

while SNMP access control is data-oriented, CLI access control is usually task oriented. There should be support for both types of access control.

NETCONF : Network configuration protocol

- A session based network management protocol
- Allows retrieving state or configuration data and manipulating configuration data on network devices.
- Below fig shows the layer architecture of NETCONF protocol



- For Nlw management architecture based on NET, the terms client and management system and the terms server and device are often used interchangeable.
- NETCONF works on SSH transport protocol.

- In addition to secure shell Transport layer protocol, NETCONF implementations can support other transport mappings such as Blocks Extensible Exchange protocol (BEEP)
- Transport layer provides end to end connectivity and ensure reliable delivery of messages.
- NETCONF uses XML-encoded Remote procedure for framing request and response messages.
- The RPC layer provides mechanism for encoding of RPC calls and notifications.
- NETCONF provides operations to retrieve and edit configuration data from n/w devices.
- Table 1 provides a list of some commonly used NETCONF operations.

- | | |
|------------------|---|
| 1) connect | → connect to NETCONF server |
| 2) get | → Retrieve the running configuration and state information. |
| 3) get config | → Retrieve all or a portion of a configuration data base. |
| 4) edit-config | → loads all or part of a specified configuration to the specified target configuration. |
| 5) copy-config | → create or replace an entire target config datastore with a complete source config. |
| 6) delete config | → Delete the contents of a configuration data store. |

- 7) lock → Lock a configuration data store for exclusive edits by a client.
- 8) unlock → Release the lock on a configuration database.
- 9) get-schema → This operation is used to retrieve a schema (plan) from the NETCONF server.
- 10) commit → commit the candidate configuration as the device's new current configuration.
- 11) close session → Gracefully terminate a NETCONF session.
- 12) kill-session → Forcefully terminate a NETCONF session.

- The content layer consists of configuration and state data which is XML encoded.
- The schema of the configuration and state is defined in a data modeling language called YANG (yet Another Next Generation).

Connectivity Technologies

Connectivity lies at the core of the Internet of Things due to this smart devices and sensors can share data with processing modules and perform their intended functions.

• There are 6 more popular technologies in IOT such as

1. wifi
2. Bluetooth
3. cellular network (2G, 3G, 4G, 5G)
4. zigbee
5. LPWAN
6. Ethernet.

1. Wifi: It stands for wireless fidelity, It is a convenient connectivity option to transmit data over the internet without physical cables.

• This wifi connectivity technology mainly used in smart home devices like lamps, LED relays & beyond.

• It is mainly used medical devices and also every hardware component like sensors.

• It consumes high power. It has limited range upto 35 meters.

2. Bluetooth: It is a short range wireless technology.

• It is mainly used for connecting different sensors in IOT.

• The main bluetooth IOT devices are smart watches, smart phones, wearable things.

• It is slower compared to wifi.

• The range of bluetooth is upto 35 meters. It is energy efficient technology compared to wifi.

3. Cellular Network: wifi and bluetooth are used to transfer data within short range but by using cellular networks we can transfer data to anywhere.

- It has high reliability.
- It has high speed and bandwidth.
- It has robust security feature.
- The drawbacks of cellular Networks are usually high cost, high power consumption.
- The cellular technologies may be 2G, 3G, 4G or 5G.

4. Zigbee: It is also one of the IOT connectivity network.

- It is low power mesh networking.
- It creates a network of IOT devices that help each other transmit data efficiently.
- This network comprises 3 key components
 1. zigbee coordinator
 2. zigbee repitor or router
 3. zigbee end device.

1. zigbee coordinator: which servers as the network central hub and stores essential information.

2. zigbee repitors or routers: It is responsible for transmitting messages between devices & applications.

3. zigbee end devices: These are actual IOT device.

- zigbee has low power consumption and high reliability.

5. LORAWAN: It stands for long range wide area network its name suggests, facilitates long range communication between IOT devices and base stations.

• It is one of low power wide area network.

• It has high data speeds and high latency.

6. Ethernet: It is a wired connectivity technology. It is a most reliable ways to link IOT devices.

• It is exactly how your computer internet using

(2)

ethernet cable.

- It offers high speed data transfer with no or little lag.
- It more reliable and complex in nature.
- It has high speed data transmission and real time communication.

Gateway prefix Allotment:

The core purpose of gateway prefix allotment is to conserve address space while ensuring that IoT devices can communicate across the roles of the gateways differentiating the addressing and the devices it manages.

• Gateways' Role: Gateways are networking devices that connect an IoT local Area Network (LAN) of low-power devices (like sensors or controllers) to the wide area Network (WAN) or the cloud. They are responsible for forwarding packets between these networks on the IP layer.

• The Allotment: Gateway is assigned a unique network prefix. This prefix is a globally unique identifier that allows the gateway itself to be recognized and routed on the internet.

• Address conservation: The individual IoT devices connected within that gateway's domain (the LAN) are assigned internal IP addresses that are only unique and valid within that domain.

• Efficiency: By giving only the gateway a globally unique prefix, it removes the need to allocate a unique, global IP address to every single device, thereby saving a vast amount of IPv6 address space.

- The IOT Gateway acts as a communication bridge between the local, low-power sensors and actuators and the wider Internet or cloud.

- Its primary function is to enable communication between these two distinct environments that often use different connectivity types, interfaces, and protocols

Features and functions

- Protocol Translation / Bridging: The gateway translates communication protocols. For example, it might convert protocols used by sensors, such as Bluetooth LE, Zigbee, and Z-wave, into standard Internet protocols like TCP/IP.

- Edge Computing: It provides software-driven capabilities known as Edge Computing, which are executed locally on the gateway devices.

- Data Filtering and Reduction: A crucial function is to filter out unnecessary data from the massive volume generated by sensors. This reduces the amount of data that needs to be forwarded to the backend system or the cloud, saving bandwidth and processing power.

- Security Implementation: The IOT gateway often includes features for implementing security measures to protect the local network and data as it moves to the Internet.

- Connecting Minimalistic Devices: It allows devices that are too small or resource-constrained to connect directly to the Internet to be connected via the

③ gateway, giving them access to remote IT components.

Impact for mobility on addressing

The impact of mobility on addressing by the use of specialized communication and addressing standards that support mobile devices and large-scale connectivity:

- Mobile communication standards - The need to support mobile IoT devices, such as those in vehicular or wearable applications, requires the use of Mobile communication standards like 2G/3G/4G/5G. These standards are explicitly designed for mobile communications and allow IoT devices to communicate over cellular networks.

- Unique Addressing for scale - Every IoT device, whether fixed or mobile, must have a unique identity which is typically an IP address or a Uniform Resource Identifier (URI). Addressing is handled by hierarchical IP addressing schemes like IPv4 or IPv6. The massive number of devices, which includes mobile endpoints, is the primary driver for the adoption of IPv6.

- IPv6 addresses the scale issue by using a 128-bit address scheme, providing 2^{128} or 3.4×10^{38} addresses. This vast address space is necessary to uniquely identify the trillions of devices expected to be connected, including all mobile and highly dynamic "things" in the IoT.

Application layer protocol | Data protocol

- Application layer protocols define how the applications interface with lower layer protocols to send the data over the network.

- The Application data typically in files or text is encoded by the application layer protocol and encapsulated in

the transport layer protocol which provides connection or connection-less communication over the network.

- A port number will be assigned for every application such as 80 for HTTP and 22 for SSH to transmit the data.

- Application layer protocol enables process to process connection using ports.

- There are different types of protocols available in application layer.

- 1) HTTP 2) COAP 3) websocket 4) MQTT 5) XMPP 6) DDS
- 7) AMQP

1) HTTP: Hypertext transfer protocol is the application layer protocol that forms the foundation of the world wide web.

- HTTP includes different commands such as GET, PUT, POST, DELETE etc.

- This protocol works based on request and response service, for example a client send request to server using HTTP commands.

- HTTP is a state-less protocol and each HTTP request is dependent of other request.

- HTTP mainly used for searching different web browser or applications for the client of the user.

2) COAP: Constrained application protocol

- COAP is an IOT protocol.

- It is an application layer protocol for machine to machine applications. mean for constrained environments with constrained devices and constrained networks.

- COAP is designed to allow simple devices to join

IOT through the low band width restricted network,

- It follows request response model.

- It runs over UDP protocol.

- It uses fewer resources than HTTP

- COAP client can use GET, PUT, DELETE commands.

- COAP is designed to enable low power sensor to use REST while meeting their power constraints.

- It has 4 messaging modes.

- confirmable
 - Non conformable
 - piggy back
 - separated.

3) MQTT: (Message Queuing Telemetry Transfer)

- It is an application layer protocol.

- It is used for machine to machine and IOT connectivity.

- It is a publish, subscribe based messaging protocol.

- It is light weight messaging protocol.

- MQTT uses a client-server architecture where the client (IOT devices) connects to the server (MQTT broker) and publishes messages to topics of server.

- It is suited for wireless sensor network and IOT sensors where communication nodes communicate with application through broker.

- It runs over TCP IP protocol

- It was developed by IBM and eurotech in 1999.

- MQTT is designed for limited devices and low bandwidth communication.

- It has 4 main components

- Broker

- Topic

- message

- client.

4) XMPP: (Extensible messaging & presence protocol)

- It is a real-time communication protocol and streaming XML data between network entities.
- XMPP has wide range of applications including messaging, presence, data syndication, gaming, multi-party chat and voice or video calls.
- It works based on XML (Extensible Markup language).
- It uses client-server architecture, i.e. It supports client to server and server to server.
- It is open standard protocol.
- It supports group chat and conference chat.
- It is well suited for cloud computing.
- It is secured and flexible.
- XMPP allows real-time communication between IoT devices.
- XMPP is a decentralized protocol.

5) AMQP: (Advance messaging queuing protocol)

- It is an Application layer protocol for business messaging.
- AMQP supports both point to point and publish-subscribe model.
- It is an open standard Application layer protocol for message oriented middle ware.
- Mostly, it is used in Banking applications.
- AMQP is mainly used in transaction message between servers and users.
- It uses TCP for reliable communication.
- It provides end point Acknowledge acceptance of each message.

5)

- It follows pub-sub model for communication.

6) DDS: (Data distribution service)

- It is data centric middleware standard for device to device or machine to machine communication.
- DDS uses a publish-subscribe model where publishers create topics to which subscribers can subscribe.
- Publisher is an object responsible for data distribution and subscriber is responsible for receiving published data.
- DDS provides quality of service control and configurable reliability.

7) web socket:

- It allows full duplex communication over a single socket connection for sending messages between client and server.
- web socket is based on TCP protocol.
- The client can be browser or a phone application or an IoT device.

1. Multihoming

Definition: Multihoming in the Internet of things (IoT) refers to IoT devices connecting to multiple networks simultaneously or switching between them, enhancing reliability, latency, and improving fault tolerance.

* What is Multihoming?

Multihoming allows IoT devices, such as sensors or wearables, to maintain active links to more than one network (e.g., WiFi, cellular, or Bluetooth) at once. This setup helps devices stay online even if one connection fails, which is crucial for real-time data transmission in IoT ecosystems.

* Why it matters in IoT?

In IoT, where billions of devices generate vast data streams, single-network reliance can lead to downtime. Multihoming addresses this by enabling load balancing and failover, supporting applications from industrial automation to connected vehicles. For instance, a smart home hub might use both Wi-Fi and 5G to ensure uninterrupted control.

* Common use cases:

- Smart cities: Traffic sensors multihomed to public Wi-Fi and LTE for continuous monitoring.
- Healthcare: Wearable devices switch between Bluetooth and cellular for vital signs tracking.

• Agriculture: Soil sensors using satellite and LoRaWAN for remote farm data.

* Comprehensive Overview of Multihoming in IOT.

Introduction to Multihoming concepts.

Multihoming, a networking strategy traditionally used in enterprise IT to connect to multiple internet service providers (ISPs) for redundancy and performance optimization, has evolved significantly in the context of IOT. In IOT environments, multihoming extends this idea to resource-constrained devices that must operate in heterogeneous, often unreliable networks. It involves devices maintaining simultaneous or alternate connections to diverse access technologies, such as Wi-Fi (IEEE 802.11), cellular networks (e.g., 4G/5G), Bluetooth Low energy (BLE), Zigbee, or low-power wide-area network (LPWAN) like LoRaWAN. This capability is particularly vital as IOT deployment scale to trillions of devices by 2030, demanding uninterrupted connectivity amid fluctuating signal quality, mobility, and interference.

The core principle of multihoming in IOT is to mitigate mitigate points of failure. Unlike traditional multihoming, which focusses on IP-level routing (e.g. Border gateway protocol or BGP for path selection), IOT variants

emphasize lightweight protocols suitable for battery-powered endpoints. For ex. protocols like multipath TCP (MPTCP) or stream control Transmission protocol (SCTP) are adapted to aggregate bandwidth from multiple paths, ensuring data flows even if one link degrades.

* Technical mechanisms and protocols.

Implementing multihoming in IoT requires careful orchestration at multiple layers of the protocol stack. At the transport layer, MPTCP stands out as a key enabler, allowing a single TCP connection to use multiple subflows over different interfaces. This is especially useful for mobile IoT nodes, where devices can dynamically add or remove path based on ^{signal} strength or cost.

At the network layer, IPv6 multihoming plays a pivotal role, leveraging features like identifier locator Network Protocol (ILNP) or shim6 for address portability without session disruption. For edge computing scenarios, Software Defined Networking (SDN) controllers can orchestrate multihoming by virtualizing network slices across 5G Non-standalone (NSA) architectures, enabling seamless handover during vertical handoffs (eg., from Wi-Fi to LTE). Security is integrated via protocols like IPsec for encrypted multihomed tunnels, addressing vulnerabilities such as

man-in-the-middle attacks during path switches. However, the evidence leans toward a trade-off: while multi-hopping enhances availability, it increases attack surfaces due to more exposed interfaces.

Protocol/Mechanism	Layer	Key feature in IOT multihoming	Example usecase.
MPTCP	Transport	Aggregate multiple paths for throughput	Video Streaming from drones.
SCTP	Transport	multi-hopping with multi-streaming	Industrial sensor data Aggregation.
ILNP	Network	Locator Identifier Split for mobility.	wearable health monitors.
SDN with 5G slicing	control	Dynamic path Orchestration	Smart grid-fault detection.
CoAP over UDP	Application	Lightweight multihoming for constrained devices	environmental monitoring sensors.

* Benefits and Advantages:

The adoption of multihoming in IOT yields substantial gains in reliability and efficiency. primarily, it provides fault tolerance, where devices can failover to secondary networks in under 50ms, critical for time-sensitive applications like autonomous vehicles or remote surgery. Research suggests that multihoming can reduce packet loss by upto 40% in urban deployments

3

with 'intermittent coverage'.

performance-wise, it enables load balancing and bandwidth aggregation, allowing low-bandwidth interfaces (eg., BLE at 1mbps) to complement high-speed ones (eg. 5G at 10Gbps). This is particularly empathetic to developing regions, where hybrid networks bridge digital divides. Energy efficiency, though debated, leans towards net positives when using duty-cycling techniques, conserving battery life by offloading traffic to optimal paths.

* Challenges and Limitations.

Despite its promise, multihoming in IoT faces hurdles that warrant diplomatic consideration from all perspectives. Seamless handover remains complex, with latency spikes during vertical handoffs potentially disrupting ultra-reliable low-latency communications (URLLC) in 5G-IoT.

Power consumption is a contentious issue; maintaining multiple radios can drain batteries 2-3 times faster, challenging for untethered sensors.

Security and privacy concerns amplify in multihomed setups, as device juggle authentication across providers, risking key management overhead.

Standardization lags, with interoperability issues, fragmenting support across IoT platforms.

2. Deviations from Regular web.

* Key points on deviations from Regular web:-

- Device-centric shift: Unlike the human-focused regular web, IoT emphasizes automated data generation and exchange among resource-constrained devices, leading to push-based interactions rather than user-initiated pulls.
- Lightweight protocols: IoT favors efficient, low-overhead protocols like MQTT and CoAP over HTTP's heavier request-response model, enabling real-time communication in bandwidth-limited environments.
- Layered Architecture: IoT uses multi-layered structures for sensors data handling and scalability, diverging from the traditional web's simpler client-server setup.
- Event-Driven Dynamics: IoT prioritizes proactive notifications and data fusion for actions, contrasting the web's hyperlink-based navigation and query responses.
- Challenges in Maturity and security: IoT's emerging standards introduce interoperability hurdles and heightened security needs due to diverse, often unsecured devices.

Protocols: Efficiency over web standards.

The regular web relies on HTTP over TCP/IP for reliable, human-readable exchanges, but IoT protocols adapt to constraints like low power and intermittent connectivity. For instance, MQTT employs a publish model for bidirectional, real-time data flow, reducing bandwidth use compared

to HTTP's polling.

Architecture: From servers to sensors network. Traditional web architecture centers on browsers and servers for content delivery, while IoT introduces layers like perception (sensors) and middleware (integration) to manage vast device ecosystems. This supports massive scalability but requires handling heterogeneous data streams, unlike the web's uniform hyperlink structure.

Architectural Deviations: Layered vs. client-server models. IoT architectures are typically multi-layered to bridge physical sensors with cloud analytics, diverging from the web's streamlined client-server duality. A common IoT framework includes five layers:

Layer	Description in IoT	Deviations from regular web.
perception	Sensors (e.g., RFID, temp. gauges) collect raw environmental data.	Absent in web; web lacks direct hardware interfacing, relying on user inputs via keyboards/screens.
Network / Transmission	Transmits data via wireless protocols (e.g., Wi-Fi, Zigbee) to gateways.	web uses TCP/IP for high-bandwidth reliability; IoT opts for low-power alternatives like LPWAN for extended range in remote setups.
Middleware	Integrates services, handles computation, and	web middleware is server-centric (e.g., APIs)

	ensures device interoperability.	IoT focuses on fusing heterogeneous data streams for event detection.
Application	Delivers domain-specific services (e.g., smart homes, traffic management).	web apps are browser-agnostic and generalized; IoT apps embed directly in devices for real-time control.
Business	Analyzes data for strategies, models, and monetization.	Not standard in web architectures; web emphasizes content delivery over operational insights.

* Protocol Innovations: Lightweight Alternatives to HTTP.

At the protocol level, IoT rejects the web's HTTP dominance due to its overhead - headers alone can exceed 200 bytes per request, unsuitable for battery-powered nodes. Instead, IoT protocols prioritize efficiency, security, and reliability in constrained environments. A comparative table illustrates

key contrasts:

protocol	model	Transport	key features in IoT	Deviations from HTTP (web standard).
HTTP	Request/response	TCP	stateless, polling for updates; widely used for APIs.	Baseline: High reliability but bandwidth-heavy; stateless sessions

MQTT	Publisher/Subscriber	TCP	QoS levels (0-2 for delivery guarantees); retained msgs; low overhead (w/2-byte headers).	require constant reconnections. push-based for real-time; stateful connections reduce latency; ideal for telemetry vs web's
CoAP	Request/Response	UDP	DLTS security; multicast support; translates HTTP semantics.	Lighter than HTTP (no TCP handshake); suits low-power networks like 6LoWPAN, enabling REST-like IoT APIs.
Zigbee	Mesh Network -ing.	Physical (IEEE 802.15.4)	Low data rate (250 Kbps); AES-128 encryption; short-range (30m).	Not IP-based; focuses on energy efficiency over webs high-speed ethernet / Wi-Fi.
BLE (Bluetooth low energy).	point-to-point	Physical	1 Mbps; 150m range, secure pairing.	Short-range, low-latency for wearables; contrasts web's long-range internet reliance.
LPWAN (eg., LoRaWAN)	wide-range	various	Long range (upto 15kms); ultra-low power (eg., 100 bps for Sigfox).	Geared for massive, rural deployments; minimal bandwidth vs web's high-throughput cellular.

Data Handling and Value Delivery: From Queries to Action.
In the regular web, data consumption follows a query-response cycle: users search, servers reply, and value emerges from retrieved information. IoT inverts this - devices push fused data for automated actions, such as alerting a farmer to soil moisture levels.

This event-driven fusion relies on context and patterns (e.g., anomaly threshold), demanding semantic processing beyond the web's simple links.

The evidence points to enhanced efficiency in IoT, but it introduces challenges like data silos, where proprietary formats hinder interoperability.

Security deviations amplify risks: IoT devices often ship with default credentials, vulnerabilities to botnets, unlike the web's mature TLS ecosystem.

Protocols like MQTT incorporate topic-based authorization, yet overall, IoT's scale heightens threats - research estimates billions of unsecured endpoints by 2025.

3. Idea IOT Identification.

key points.

- IOT identification refers to process of assigning and using unique identifiers to billions of connected devices in the Internet of things (IOT), enabling secure communication, tracking, and data exchange; research suggests this is foundational for IOT scalability, through challenges like address exhaustion persist.
- Traditional methods like RFID and IP addresses provide reliable but limited identification, while modern machine learning approaches offer higher accuracy in dynamic environments, with evidence leaning towards hybrid techniques for better security.
- standards such as EPC (Electronic product code) from GS1 ensure interoperability, but privacy concerns resources constraints remain debated across stakeho

- lders.

* What is IOT identification?

IOT identification is the mechanism that gives each device or object a unique "fingerprint" or ID, allowing it to be recognized, authenticated, and managed within vast networks. This is crucial for everything from smart homes to industrial sensors, where devices must interact without confusion. For ex. a smart thermostat might use an RFID tag for initial setup and an IP address for ongoing communication.

* Why it Matters.

Without robust identification, IoT system risk security breaches, inefficient resources use, and interoperability issues. It supports applications like supply chain tracking and healthcare monitoring, helping isolate vulnerable devices. Studies show machine learning methods can achieve up to 98% accuracy in identifying devices, enhancing network safety.

* Common Methods and Standards.

Methods range from hardware-based (e.g., RFID tags embedded unique serial numbers) to software-driven (e.g., behavioral fingerprinting via network traffic). Key standards include:

- EPC/GS1: A global syntax for unique object IDs, often paired with RFID.
- IPv6: Addresses the IPv4 shortage, supporting trillions of devices.
- UUID: Software-generated unique IDs for non-physical entities.

* The Role and Importance of Identification in IoT.
Identification serves as the "proof of identity" for IoT objects, linking the physical world to digital services. In essence, it allows devices to:

- Communicate effectively:

- Devices exchange data only after verifying each other's identity, preventing unauthorized access.

• **Enable Services:** From environmental monitoring in wireless sensor networks (WSNs) to supply chain logistics, IDs match objects to virtual profiles on servers.

• **Support Scalability:** With IoT devices outnumbering humans, unique IDs prevent collisions and ensure efficient routing.

The importance is underscored in real-world applications. For instance, in healthcare, RFID-enabled IPs track medical devices, reducing errors in delivery and improving efficiency. In smart cities, IP-based identification manages traffic sensors for real-time analytics.

Research highlights that poor identification manages traffic sensors for real-time analytics.

* Traditional Identification Methods.

Traditional methods, rooted in early IoT developments like sensors network, rely on hardware tags or network protocols for straightforward, cost-effective identification. These have been battle-tested in industries but face limitations in massive, dynamic deployments.

• **Radio Frequency Identification (RFID):**

RFID uses electromagnetic fields to automatically identify tags attached to objects. Tags (passive, active, or semi-passive) store unique serial numbers transmitted via radio waves to readers. Applications include inventory management

access control. The read range follows Friis' transmission equation, influenced by frequency, power, and antenna gains, typically spanning metres.

- Barcodes and 2D codes:-

These optical labels encode static data (e.g., product details). Scanned by lasers or cameras. While simple and inexpensive for retail, they require proximity and visibility, making them unsuitable for remote IoT scenarios.

- IP Addressing (IPv4 and IPv6): IP addresses serve as network-layer IDs, with IPv4 providing about 4.3 billion unique values and IPv6 expanding to roughly 3.4×10^{38} . IPv6 includes auto-configuration and enhanced security features.

Complementary protocols like 6LoWPAN adapt IPv6 for low power WSNs using IEEE 802.15.4, reducing overhead in battery-constrained environments.

- Electronic product codes (EPC):

Developed by EPCglobal (under GS1), EPC is a URI-based standard for static object details like manufacturer and serial number.

Its integral to RFID, where tags carry EPC data along-side Tag Identifier (TID) for the tag itself.

* Modern Identification Methods.

Advancements in computing have spurred modern techniques that leverage data patterns, AI, and hardware.

intrinsic for more robust, adaptive identification. These address traditional shortcomings by focusing on dynamic behaviours rather than static tags.

- **Device Fingerprinting:** This extracts unique "signature" from device traits, such as network packet inter-arrival times (IAT) or hardware variations via physical unclonable functions (PUFs). For example, IOT-ID combines clock counts, ADC signals, and differences into a composite ID using multi-stage ML classifiers like k-Nearest Neighbors (KNN).

- **Machine Learning (ML) and Deep Learning (DL):** ML models, such as Decision Trees (DT) or Random forests, classify devices from traffic features (e.g., packet size, protocol flags). DL-based methods reach 98% accuracy in some studies. DL variants like Convolutional Neural Networks (CNNs) handle complex data, such as YOLO for object detection in vision-integrated IOT.

- **Behavior-Based Approaches (e.g., IOTDevID):**

IOTDevID models packet-level behaviors using 52 generalized features (e.g., payload entropy, port classes) selected via genetic algorithms and voting. It classifies via DT, aggregating labels post-prediction to handle shared addresses in gateway scenarios.

- **Computer Vision Integration:**

Techniques like U-Net segmentation with Histogram of Oriented Gradients (HOG) identify objects from images, useful in surveillance IOT.

* Standards and Protocols:

Standardization ensures interoperability across vendors. Core ones include:

- **GS1 EPC and RFID Standards:** EPC defines syntax for unique IDs (e.g., serial shipping (version 1.13) specifies patterns like pure identity for fixed-width integers).
- **Unique Identifier (UID/TID):** embedded in chips during manufacturing, UIDs provide tamper-proof serial number; in EPC, TID identifies the tag separately from the product.

- **Unique Identifier (w/ IETF and IEEE protocols):**

IPv6 / 6LOWPAN for addressing; IEEE 802.15.4 for low-rate wireless.

- **Semantic standards:** RDF / OWL for knowledge representation, aiding ID-based querying.

* Challenges and Future Directions

IoT identification grapples with:

- **Scalability and mobility:** Billions of devices strain resources; dynamic IPs fail for roaming.
- **Security / privacy:** wireless method risk interception; ML models are vulnerable to adversarial tweaks.
- **Resource Limits:** Low-power devices can't handle DL's demands.
- **Interoperability:** heterogeneous encodings confuse systems.

Bluetooths:-

It is used for short range wireless voice and data

Communication.

* It is a wireless personal Area Network (WPAN) technology and is used for data communications over smaller distances.

* It operates within the unlicensed, business, scientific and clinical (ISM) bands from 2.4 GHz to 2.485 GHz.

* Bluetooth range is upto 10 meters with speed of 1 mbps or 3 mbps.

* A bluetooth network is called a piconet and a group of inter connected piconets is called a scatter net.

* Bluetooth simply follows the principle of transmitting and receiving data using radio waves (3 kHz to 300 GHz)

Key features:-

* The transmission capacity of Bluetooth is 720 kbps.

* Bluetooth is a wireless technology.

* Bluetooth is a low cost and short distance radio communications standard.

* Bluetooth is robust and flexible.

* The basic architecture unit of bluetooth is piconet.

Bluetooth Protocol stack:-

* Radio layer

* base band link layer

* link manager protocol layer

* logical link control and adaptation protocol layer

* Service Discovery Protocol layer

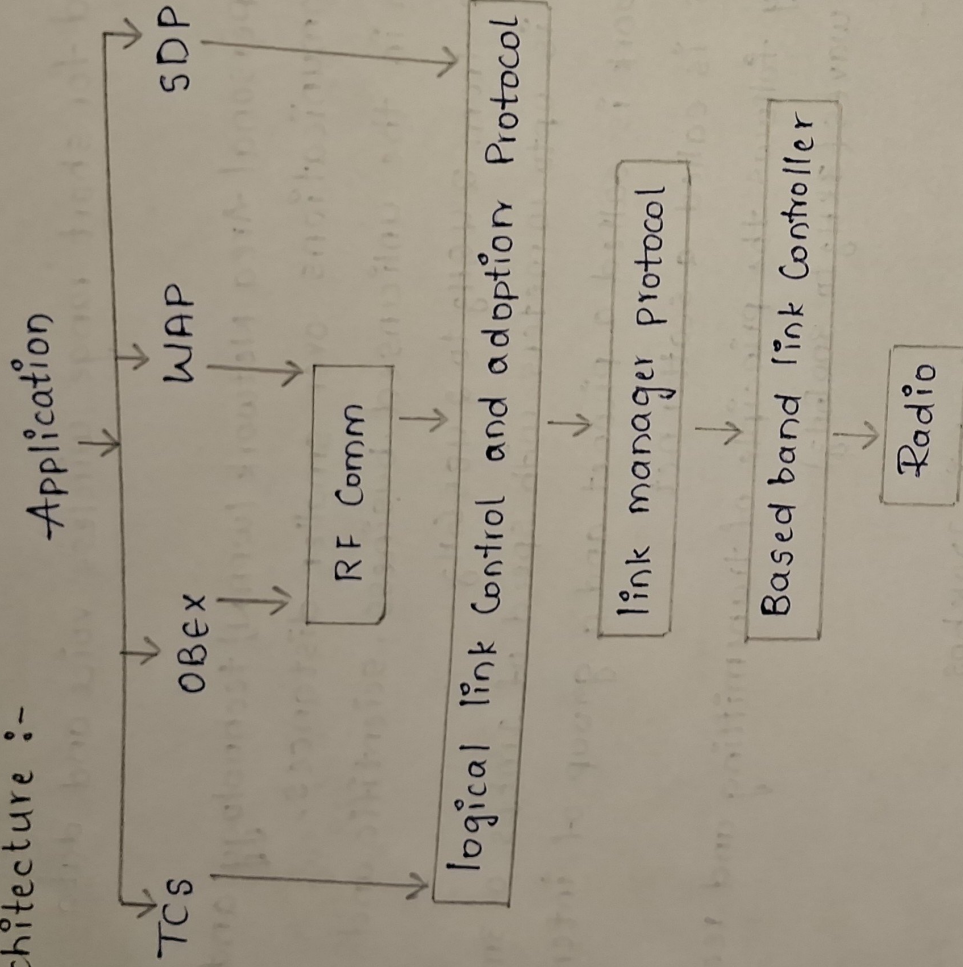
* RF Common layer

* OBEX:- It is short for object exchange

* WAP:- It is short for wireless Access Protocol.

TCS :- It is the short for telephony Control Protocol.

Architecture :-



Various types of Bluetooth :-

1. In Car Headset
2. Stereo headset
3. web Cam
4. Bluetooth Equipped printer
5. Bluetooth GPS

Applications :-

- * It can be used in wireless headsets, wireless PANS, JANS.
- * It can connect a digital camera wireless to a mobile phone.
- * It can transfer data in terms of videos, songs, photography files from device to other.
- * It is used in the sectors of medical healthcare, sports and fitness and military.

Advantages :-

- * It is a low cost and easy to use.
- * It can also penetrate through walls.
- * It creates a network without any wires.
- * It creates a network without any wires.

802.11 Protocol :-

802.11 Wave is a wireless communication protocol used by automatic or automatic applications for the purpose of connection and communication.

- * It is invented by Zensys a Danish American Company in 1999.

802.11 Wave Protocol Stack :-

802.11 Wave consists of five layers.

- 1. Physical layer
- 2. MAC layer
- 3. Transport layer
- 4. Network layer
- 5. Application layer

1. Physical layer :- This layer has many features but the important one is modulation and coding. In this layer data is transferred in

one is modulation and coding. In this layer data is transferred in one is modulation and coding. In this layer data is transferred in

8-bit blocks and the most significant bit is sent first.

2. MAC layer :- MAC layer as the name suggests take care of medium access control among slave nodes based on collision avoidance and back off algorithms.

3. Transport layer :- 802.11 wave transport layer is responsible for packet acknowledgement and packet origin

extramission, packet acknowledgement and packet origin authentication.

* It consists of four basic frame types.

- 1. Single cast frame
- 2. Ack frame
- 3. Multicast frame
- 4. Broadcast frame

4. Network layer :- It controls the frame routing from one node to another node.

5. Application layer :- It is responsible for recording and execution of commands in the 802.11 wave network.

→ The components of 802.11 wave include controllers, slave nodes, home ID.

node ID and routing tables.

Controllers:- A Controller is a unit that has the ability to compile a routing table of the network and can calculate routers to the different nodes.

There are two types of Controllers.

i. Primary controller which contains description of wave network and controls the outputs.

* It assigns network ID or home ID or Node ID to the wave during the enrollment process.

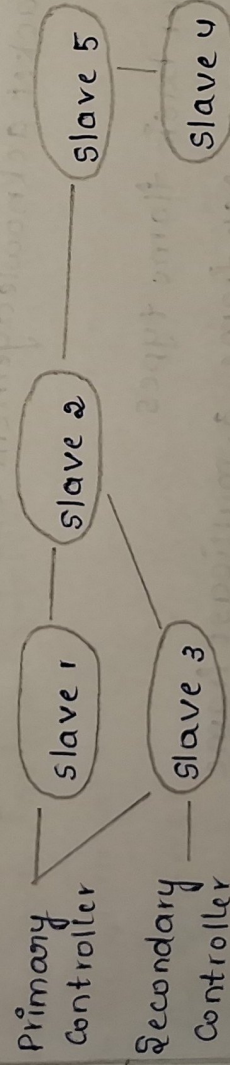
ii. Secondary Controller can also have a network ID and it remains constant to maintain routing tables.

Slave nodes:- Slave nodes are the nodes that do not contain routing tables that may contain a network and slave node have the ability to receive frames and respond to them if necessary.

Home ID:- The ID used by a wave for the separation of the network from each other is called Home ID. It is created by the primary controller and is 32 in size.

Node ID:- The identification number or an address that is given to every device during the process of inclusion is called Node ID.

Routing table:- It is used by controllers for calculating routers.



Characteristics:-

1. Uses RF for signalling control.
2. Data Rate: upto 100 kbps
3. frequency: 700 MHz (15m)
4. FSK modulation
5. Range: 30 meter

Applications:-

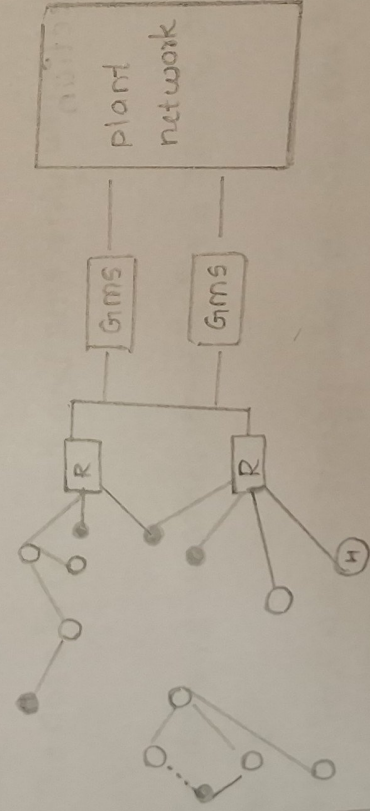
1. Home Automation
2. Water management using flood sensors.
3. finger print scanners.

ISA 100.11a :-

* ISA 100.11a an Industrial Internet of Things (IIoT) standard from the International Society of Automation (ISA) is a wireless connectivity technologies for industrial environments using a protocol stack based on IEEE 802.15.4 for the low power, secure and reliable communication in star or mesh network topologies.

* It supports IPv6, AES encryption, frequency hopping for interference avoidance.

* ISA 100.11a is the first standard of the family which describes mesh network designed to provide secure wireless communication to process control.



0 - Routing

● - non routing

Ⓜ - Handled

R - Back bone Router

G - Gateway

m - System manager

S - Security manager

ISA native and legacy protocols (tunnelling)
UDP CIETP RFC 768
6LOWPAN
upper data links
IEEE 802.15.4
IEEE 802.15.4 (2.4GHz)

In wireless sensor networks, nodes of detection refer to the strategy and mechanisms by which sensor nodes detect and respond to events. stimuli or conditions in their environments. these nodes play a crucial role in determining the efficiency, accuracy and energy consumption of the network.
→ main nodes of detection in wsn's are :-

1. Event Driven Detection
2. Periodic (Time-Driven) Detection
3. Query Driven Detection
4. Qa. hybrid detection
5. Threshold based detection
6. Continuous detection
1. Event Driven Detection:-

Sensors remain mostly in a sleep or low power state and only become active when a specific event occurs.

Ex:- Motion Detection (eg detecting instruction)



malicious activity, unauthorized access, policy violations

fire or gas leak detection.

Advantages:-

- * low power consumption
- * quick response to specific trigger

Disadvantages:-

- * May miss gradual or non sudden events.

2. Periodic (Time driven) Detection:-

- * Sensor, collect and transmit data at regular time intervals, regardless of changes in the environment.

Ex:- Temperature logging every 10 mins
soil moisture monitoring in agriculture

Adv:- Good for monitoring trends or long term changes.

Disad:- higher energy consumption

- * may send redundant data if the environment is stable.

3. Query Driven Detection:-

- * Sensor nodes respond to queries sent by the base station or sink node.

- * The data is collected and sent only when requested.

Ex:-

On demand weather that is in a specific area.

adv:-

Efficient use of energy when queries are infrequent

dis adv:-

Not suitable for critical or time sensitive app.

4. Hybrid detection:-

It combines two or more detection nodes (eg driven and periodic) to leverage the advantage of each.

Ex:- Sensors sleep but perform periodic health checks and wake fully on event detection.

Adv:- flexible and adaptive to different situation

disadv:- more complex design and logic.

5. Threshold based detection:-

Sensors monitor data continuously or periodically and triggers an action or report only if a certain threshold is crossed.

Ex:- Alert when temp exceeds 50°C

adv:- Reduces unnecessary data transmission

disadv:- may miss events that doesn't cross the set threshold

6. Continuous detection:-

* Sensors continuously monitors and report data without interruption.

Ex:- Real time video surveillance

Adv:- high granularity and accuracy

Disadv:- very high energy and bandwidth consumption.

Main Challenges in WSNs :-

1. Energy Constraints.

2. Limited Processing and storage

3. Scalability

4. Data Aggregation and fusion

5. Communication Reliability

6. Sensor security and privacy

7. Localization and Coverage

8. Fault Tolerance

9. Topology management

10. Environmental and physical challenges

11. Realtime data processing

12. Realtime data processing

12. Interoperability and standards.

* A WSN comprises of differe distributed devices with sensors that are used to monitor the Environmental and physical Conditions

* A WSN consists of a number of End nodes and router and a Coordinators.

End nodes have several sensors attached to them. End nodes can also act as routers.

- * Routers are responsible for routing the data packets from end nodes to the coordinators.
- * The coordinators collect the data from all the nodes.
- * The coordinators also act as a gateway that connects the WSN to the Internet.

Characteristics:-

- * wireless protocols (Zigbee, Bluetooth, WiFi etc)
- * Range (10m to 100m)
- * Speed (100bps to few mbps)
- * Operating frequency (15 bands, 2.4 GHz in majority devices)
- * low power operating nodes
- * WSNs are self-organizing networks
- * Robust structure

IoT Enabling technologies:-

- * wireless sensor networks
- * cloud computing
- * By data Analyst
- * Communication protocols
- * Embedded systems

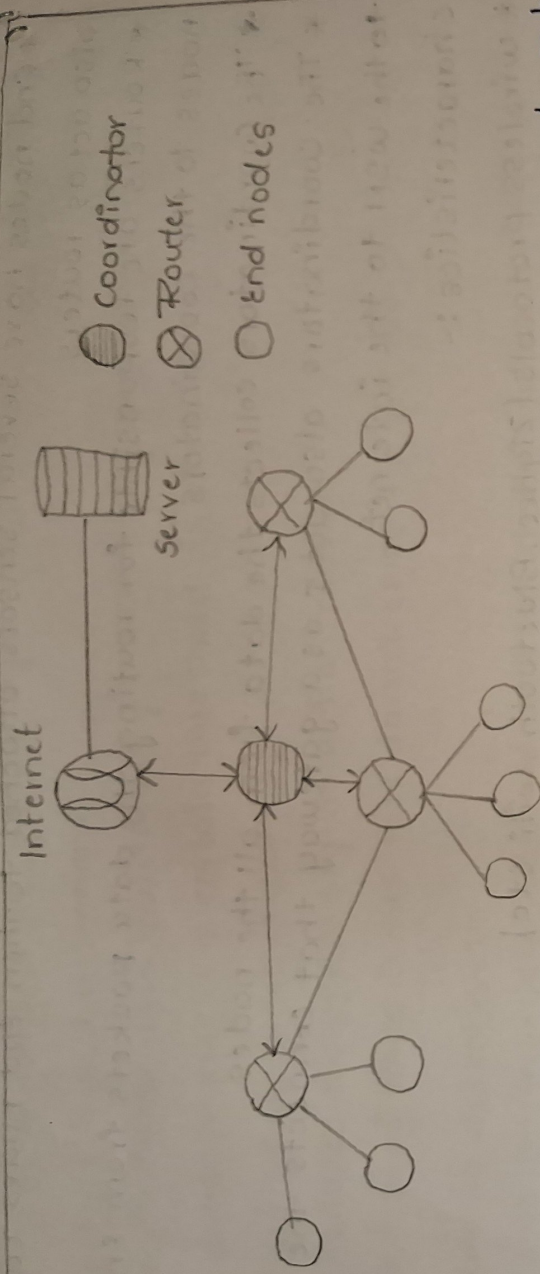
wireless sensor networks:-

End nodes:- These are used to capture physical data. End nodes are nothing but sensors.

Ex:- moisture sensor, gas sensor, temperature sensor. Router will be having physical data of multiple end nodes.

Coordinator/gateway:- these are used to connect to the Internet.

* End nodes have wireless protocol that can communicate with routers and routers are connected to gateways to forward packets from every other devices.



* Each device is assumed with an updated network graph network layer (HART) $\Rightarrow$ N/w + Transport + Session

Application layer:-

- * Handles communication between gateways and devices via a series of command and response messages.
- * Responsible for extracting commands from message. Executing it and generating responses.
- * This layer is seamless and does not differentiate between wireless and wired versions of HART.

Introduction of ZigBee

ZigBee is a personal area network task group with low rate task group 4. It is a technology of home networking. ZigBee is a technological standard and created for controlling and sensing the network.

As we know that ZigBee is the personal area network of task group 4 so it is based on IEEE 802.15.4 and is created by ZigBee Alliance.

ZigBee is an open, global, packet-based protocol designed to provide an easy-to-use architecture for secure, reliable, low power wireless networks, flow or process control equipment can be placed anywhere and still communicate with the rest of the system. It can also be moved, since the network doesn't care about the physical location of a sensor, pump or valve.

IEEE 802.15.4 developed the PHY and MAC layer whereas the ZigBee takes care of upper higher layers.

ZigBee is a standard that addresses the need for very low-cost implementation of low power devices with low data rates for short-range wireless communications.

IEEE 802.15.4 supports star and peer-to-peer topologies.

The ZigBee specifications supports star and two kinds of peer-to-peer topologies, mesh and cluster tree. ZigBee-compliant devices are sometimes specified as supporting point-to-point and point-to-multipoint topologies.

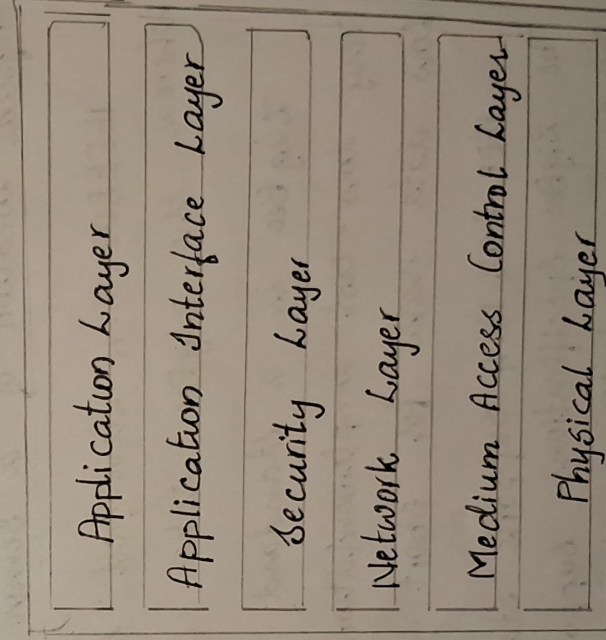
Zigbee Network Topologies

- Star Topology (ZigBee Smart Energy) : Consists of a coordinator and several end devices, end devices communicate only with the coordinator.
- Mesh Topology (Self Healing Process) : Mesh topology consists of one coordinator, several routers, and end devices.
- Tree Topology : In this topology, the network consists of a central node which is a coordinator, several routers, and end devices. The function of the router is to extend the network coverage.

Architecture of Zigbee

Zigbee architecture is a combination of 6 layers.

1. Application Layer
2. Application Interface Layer
3. Security Layer
4. Network Layer
5. Medium Access Control Layer
6. Physical Layer



Zigbee Applications

1. Home Automation
2. Medical Data Collection
3. Industrial Control Systems
4. Meter reading System
5. Light Control System
6. Commercial
7. Government Markets worldwide
8. Home Networking

Comment

More info

Advertise with us

General Characteristics of Zigbee Standard

- Low Power Consumption
- Low Data Rate (20 - 250 kbps)
- Short - Range (75 - 100 meters)
- Network Join Time (~30 msec)
- Support Small and Large Networks (upto 65000 devices (Theory); 240 devices (Practically)).
- Low Cost of products and cheap Implementation (Open Source Protocol).
- Extremely low-duty cycle.
- 3 frequency bands with 27 channels.

Operating Frequency Bands (Only one channel will be selected for use in a network):

1. Channel 0 : 868 MHz (Europe)
2. Channel 1 - 10 : 915 MHz (the US and Australia)
3. Channel 11 - 26 : 2.4 GHz (Across the world)

- Physical layer : The lowest two layer ~~is~~ i.e. the Physical and the MAC (Medium Access Control) layer are defined by the IEEE 802.15.4 Specifications. The Physical layer is closest to the hardware and directly controls and communicates with the Zigbee radio. The physical layer translates the data packets in the over-the-air bits for transmission and vice-versa during the reception.

- Medium Access Control layer (MAC Layer) : The layer is responsible for the interface between the physical and network layer. The MAC layer is also responsible for providing PAN ID and also network discovery through beacon requests.

- Network layer : This layer acts as an interface between the MAC layer and the application layer. It is responsible for mesh networking.

- Application layer : The application layer in the zigbee stack is the highest protocol layer and it consists of the application support sub layer and Zigbee device object. It contains manufacturer-defined applications.

Channel Access :

1. Contention Based Method (Carrier-sense Multiple Access with Collision Avoidance Mechanism)

2. Contention Free Method (Coordinator dedicates a specific time slot to each device (Guaranteed Time slot GTS)).

Features of Zigbee

1. Stochastic addressing : A device is assigned a random address and announced. Mechanism for address conflict resolution. Parents node don't need to maintain assigned address table.
2. Link Management : Each node maintains quality of links to neighbors. Link quality is used as link cost in routing.
3. Frequency Agility : Nodes experience interference report to Channel manager, which then selects another channel.
4. Asymmetric link : Each node has different transmit power and sensitivity. Paths may be asymmetric
5. Power Management: Routers and Coordinators use main power. End devices use batteries.

Advantages of Zigbee

1. Designed for low power consumption
2. Provides network security and application support services operating on the top of IEEE
3. Zigbee makes possible completely networks homes where all devices are able to communicate and be
4. Use in smart home
5. Easy implementation
6. Adequate security features
7. Low cost: Zigbee chips and modules are relatively inexpensive, which makes it a cost-effective solution for IoT applications.

8. Mesh networking: zigbee uses a mesh network topology, which allows for devices to communicate with each other without the need for a central hub or router. This makes it ideal for use in smart home applications where devices need to communicate with each other and with a central control hub.

9. Reliability: zigbee protocol is designed to be highly reliable, with robust mechanisms in place to ensure that data is delivered reliably even in adverse conditions.

Disadvantages of Zigbee.

1. Limited range: zigbee has a relatively short range compared to other wireless communication protocols, which can make it less suitable for certain types of applications or for use in large buildings.

2. Limited data rate: zigbee is designed for low-data-rate applications, which can make it less suitable for applications that require high-speed data transfers.

3. Interoperability: zigbee is not as widely adopted as other IoT protocols, which can make it difficult to find devices that are compatible with each other.

4. Security: zigbee's security features are not as robust as other IoT protocols, making it more vulnerable to hacking and other security threats.

Applications of GLOWPAN

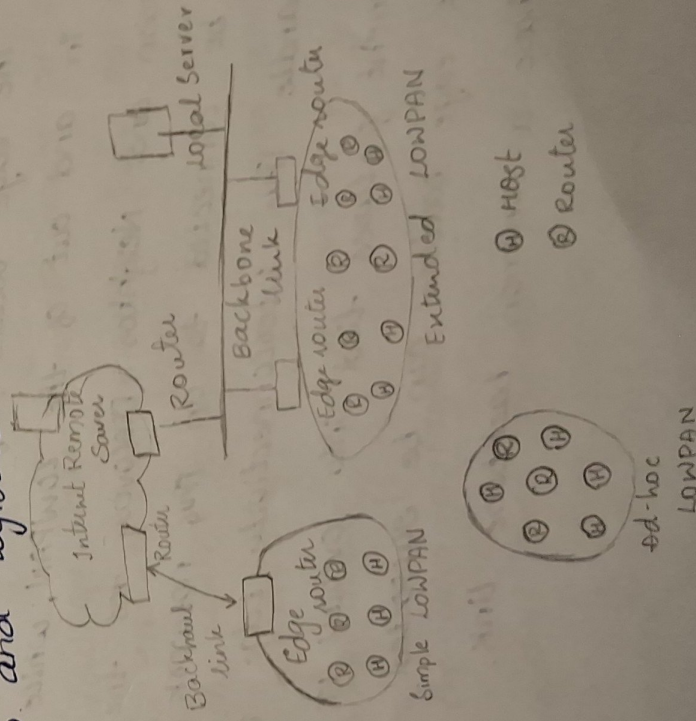
The reason why there are such a large number of technical solutions in the wireless embedded networking market is that the requirements, scale and market of embedded applications vary wildly. Applications can range from personal health sensors monitoring to large scale facility monitoring, which differs greatly. This is in contrast to PC information technology, which is fairly homogeneous and mainly aimed at home and office environments. The ideal use of GLOWPAN is in application where:

- embedded devices need to communicate with Internet-based services,
- low-Power heterogeneous networks need to be tied together,
- the network needs to be open, reusable and evolvable for uses and services
- scalability is needed across large network infrastructure with mobility.

Connecting the Internet to the physical world enables a wide range of interesting applications where GLOWPAN technology may be applicable. For example:

1. home and building automation and logistics.
2. healthcare automation and logistics.

The GLOWPAN Architecture.



• The 6LOWPAN architecture is made up of low-power wireless area networks (LOWPANs)², which are IPv6 stub networks. The overall 6LOWPAN architecture is presented in Figure Three different kinds of LOWPANs have been defined:

- Simple LOWPANs
- Extended LOWPANs
- Ad hoc LOWPANs.

• A LOWPAN is the collection of 6LOWPAN nodes which share a common IPv6 address prefix (the first 64 bits of an IPv6 address), meaning that regardless of where a node is in a LOWPAN its IPv6 address remains the same.

• An ad hoc LOWPAN is not connected to the Internet, but instead operates without an infrastructure. A simple LOWPAN is connected through one LOWPAN edge router to another IP network. A backbone link (point-to-point, e.g. GPRS) is shown in the figure, but this could also be a backbone link. An extended LOWPAN encompasses the LOWPANs of multiple edge routers along with a backbone link (e.g. Ethernet) interconnecting them.

• LOWPANs are connected to other IP networks through edge routers. The edge router plays an important role as it routes

traffic in and out of the LOWPAN, while handling 6LOWPAN compression and Neighbor Discovery for the LOWPAN. If the LOWPAN

is to be connected to an IPv6 network, the edge router will also handle IPv6 interconnectivity. Edge routers typically have

management features tied into overall IT management solutions.

Multiple edge routers can be supported in the same LOWPAN if they share a common backbone link.

5

- A LowPAN consists of nodes, which may play the role of host or router, along with one or more edge routers. The network interfaces of the nodes in a LowPAN share the same IPv6 prefix which is distributed by the edge router and routers throughout the LowPAN. In order to facilitate efficient network operation, nodes register with an edge router. These operations are part of Neighbor Discovery (ND), which is an important basic mechanism.

- of IPv6. Neighbor Discovery defines how hosts and routers interact with each other on the same link. LowPAN nodes may participate in more than one LowPAN at the same time (called multi-homing), and fault tolerance can be achieved between edge routers, LowPAN nodes are free to move throughout the LowPAN, between edge routers, and even between LowPANs. Topology change may also be caused by wireless channel conditions, without physical movement. A multihop mesh topology within the LowPAN is achieved either through link-layer forwarding (called Mesh-Under) or using IP routing (called Route-Over). Both techniques are supported by 6LoWPAN.

- Communication between LowPAN nodes and IP nodes in other networks happens in an end-to-end manner, just as between any normal IP nodes. Each LowPAN node is identified by a unique IPv6 address, and is capable of sending and receiving IPv6 packets. Typically LowPAN nodes support ICMPv6 traffic such as "ping", and use the user datagram protocol (UDP) as a transport. The Simple LowPAN and Extended LowPAN nodes can communicate with either of the servers through their

edge router. As the payload and processing capabilities of LowPAN nodes are extremely limited, application protocols are usually designed using a simple binary format in a UDP payload.

- The main difference between a simple LowPAN and an Extended LowPAN is the existence of multiple edge routers in the LowPAN, which share the same IPv6 prefix and a common backbone link. Multiple LowPANs can overlap each other even on the same channel. When moving from one LowPAN to another, a node's IPv6 addressing will change. A LowPAN Edge Router is typically connected to the Internet over a backhaul link such as cellular or DSL [10-6lowpan-nd]. A network deployment may also choose to use multiple simple LowPANs rather than an extended LowPAN on a shared backbone link, e.g. for management reasons. This is not a problem if there is low mobility between LowPANs in the network, or the application does not assume stable IPv6 addresses for nodes.

- In an extended LowPAN configuration, multiple edge routers share a common backbone link and collaborate by sharing the same IPv6 prefix, offloading most Neighbor Discovery messaging to the backbone link [10-6lowpan-nd]. This greatly simplifies LowPAN node operation as IPv6 addresses are stable throughout the extended LowPAN and movement between edge routers is very simple. Edge routers also handle IPv6 forwarding on behalf of the nodes. To IP nodes outside the LowPAN, the LowPAN nodes are always reachable regardless of their attachment point in the LowPAN extended LowPAN. This enables large

enterprise 6LOWPAN infrastructures to be built, functioning similar to a WLAN (WiFi) access point infrastructure (but at layer 3 instead of layer 2).

- 6LOWPAN does not require an infrastructure to operate, but may also operate as an Ad Hoc 6LOWPAN [10-6lowpan-nd].

In this topology, one router must be configured to act as a simplified edge router, implementing two basic functionalities:

- unique local unicast address (ULA) generation [RFC4193] and handling 6LOWPAN Neighbor Discovery registration functionality.

From the 6LOWPAN Node's point of view, the network operates just like a simple 6LOWPAN, except the prefix advertised is an IPv6 local prefix rather than a global one, and there are no routes outside the 6LOWPAN.

The protocol stack with 6LOWPAN (also called a

- A simple IPv6 protocol stack with 6LOWPAN (also called a 6LOWPAN Protocol Stack) is almost identical to a normal IP stack with the following differences.

- 6LOWPAN only supports IPv6, for which a small adaptation layer (called the 6LOWPAN adaptation layer) has been defined to optimize IPv6 over IEEE 802.15-4.

- 6LOWPAN stack implementations in embedded devices often implement the 6LOWPAN adaptation layer together with IPv6, thus they can alternatively be shown together as part of the network layer.

- The most common transport protocol used with 6LOWPAN is the user datagram protocol (UDP), which can also be compressed using the 6LOWPAN format.

• The transmission control protocol (TCP) is not commonly used with 6LOWPAN for performance, efficiency and complexity reasons. The Internet control message protocol v6 (ICMPv6) is used for control messaging, for example ICMP echo, ICMP destination unreachable and Neighbor Discovery messages.

• Application protocols are often application specific and in binary format, although more standard application protocols are becoming available.

• Adaptation between full IPv6 and the 6LOWPAN format is performed by routers at the edges of 6LOWPAN islands, referred to as edge routers. This transformation is transparent, efficient and stateless in both directions. 6LOWPAN adaptation in an edge router typically is performed as part of the 6LOWPAN network interface driver and is usually transparent to the IPv6 protocol stack itself.

IP Protocol stack

HTTP	RTP	
TCP	UDP	ICMP
IP		
Ethernet MAC		
Ethernet PHY		

6LOWPAN Protocol Stack

Application	Application Protocols	
Transport	UDP	ICMP
Network	IPv6	
Data Link	6LOWPAN	
	IEEE 802.15.4 MAC	
Physical	IEEE 802.15.4 PHY	

IP and 6LOWPAN protocol stacks

IPv6	
Ethernet MAC	6LOWPAN adaptation IEEE 802.15.4 MAC
Ethernet PHY	IEEE 802.15.4 PHY

(15)
2
Figure illustrates one realization of an edge router with 6LOWPAN support, see Section 6.4 for edge router implementation considerations. Inside the 6LOWPAN, hosts and routers do not actually need to work with full IPv6 or UDP header formats at any point as all compressed fields are implicitly known by each node.

Link layers for 6LOWPAN

- The most basic requirements for a link layer to support 6LOWPAN are framing, unicast transmission and addressing.
- Addressing is required to differentiate between nodes on a link, and to form IPv6 addresses which are then elided by 6LOWPAN compression
- It is highly recommended that a link supports unique addresses by default (e.g. a 64-bit extended unique identifier [EUI-64]), to allow for stateless autoconfiguration.
- Multi-access links should provide a broadcast service. Multicast services is required by standard IPv6, but not by 6LOWPAN (broadcast is sufficient). IPv6 requires a maximum transmission unit (MTU) of 1280 bytes from a link, which 6LOWPAN fulfills by supporting fragmentation at the 6LOWPAN adaptation layer.
- A link should provide payload sizes at least 30 bytes in length to be useful (and preferably larger than 60 bytes). Although UDP and ICMP include a simple 16-bit checksum, it is recommended that the link layer also provides strong error checking.

→ Finally, as IPsec may not always be practical for 6LoWPAN, it is highly recommended that links include strong encryption and authentication.

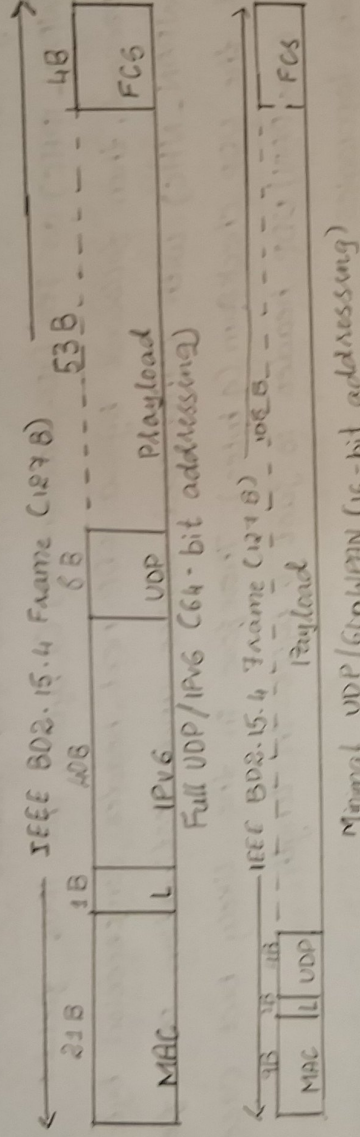
Addressing

→ IPv6 addressing are typically formed automatically from the prefix of the 6LoWPAN and the link-layer address of the wireless interfaces. The difference in a 6LoWPAN is with the way low-power wireless technologies support link-layer addressing; a direct mapping between the link-layer address and the IPv6 address is used for achieving compression.

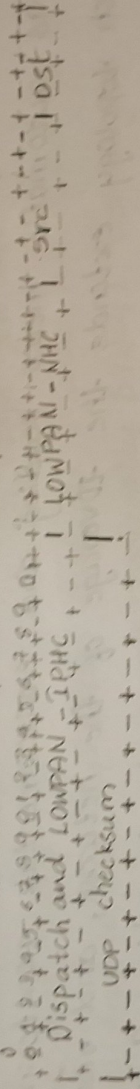
IPv6 addresses consist of a 64-bit prefix part and a 64-bit relevant header) consist of a 64-bit prefix part and a 64-bit interface identifier (IID)

→ 6LoWPAN networks assume that the IID has a direct mapping to the link-layer address, therefore avoiding the need for address resolution. The IPv6 Prefix is acquired through Neighbor Discovery Router Advertisement (RA) messages [ID-6lowpan-nd] as on a normal IPv6 link. The construction of IPv6 addresses in 6LoWPAN from known prefix information and known link-layer addresses, is what allows a higher header compression ratio.

Header format:



GLOWPAN header compression example (L = LOWPAN header)



LOWPAN/UDP compressed headers (6 bytes)

→ The LOWPAN header consists of a dispatch value identifying the type of header, followed by an IPv6 header compression byte indicating which fields are compressed, and then any in-line IPv6 fields. If, for example, UDP or IPv6 extension headers follow IPv6, then these headers may also be compressed using what is called next-header compression [10-6lowpan-hc].

→ The LOWPAN header consists of a dispatch value identifying the type of header, followed by an IPv6 header compression byte indicating which fields are compressed, and then any in-line IPv6 fields. If, for example, UDP or IPv6 extension headers follow IPv6, then these headers may also be compressed using what is called next-header compression [10-6lowpan-hc].

→ An example of GLOWPAN compression is given in Figure. In the upper packet a one-byte LOWPAN dispatch value is included full IPv6 over IEEE 802.15.4. Figure gives an example of

6LOWPAN/UDP in its simplest form (equivalent to the lower packet in Figure), with a dispatch value and IPv6 header compression (LOWPAN_IPHC) as per [10-6lowpan-hc] (2 bytes), all IPv6 fields compressed, then followed by a UDP next-header compression byte (LOWPAN_NHC) with compressed source and destination port fields and the UDP checksum (4 bytes). Therefore in the likely best case the 6LOWPAN/UDP header is just 6 bytes in length.

Mesh topologies

Mesh topologies are common in applications of 6LOWPAN such as automatic meter reading and environmental monitoring. A mesh topology extends the ID usage of the network, and reduces the cost of needed infrastructure. In order to achieve a mesh topology, multihop forwarding is required from one node to another. In 6LOWPAN this can be done in three different ways: link-layer mesh, 6LOWPAN mesh or IP routing. Link-layer mesh and 6LOWPAN mesh are referred to as Mesh-Under as the mesh forwarding is transparent to the Internet Protocol. IP routing is referred to as Route-Over.

Internet integration

When connecting a 6LOWPAN to another IP network or to the Internet, there are several issues to be considered. 6LOWPAN enables IPv6 for simple embedded devices over low-power wireless networks by efficiently compressing headers and simplifying IPv6 requirements. Issues to be considered when integrating 6LOWPANS with other IP networks include:

1. Minimum transmission unit: In order to comply with the 1500 byte MTU size requirement of IPv6, 6LOWPAN performs fragmentation and reassembly. Applications designed for the wireless Embedded Internet should however try to minimize packet sizes if possible. This is to avoid forcing a 6LOWPAN to fragment IPv6 packets, as this incurs a performance penalty.

2. Application protocols: Application protocols on the web today depend on payloads of HTML, XML or SOAP carried over HTTP and TCP.

This results in payloads varying in size from hundreds of bytes to several kilobytes. This is far too large for use with 6LOWPAN to several kilobytes.

Nodes. End-to-end application protocols should make use of UDP and compact payload formats (preferably binary) technologies which are capable of the transparent compression of web services into a format suitable for 6LOWPAN nodes are especially interesting.

3. Firewalls and NATs: In real network deployments firewalls and network address translators (NATs) are a reality. When connecting 6LOWPAN through these there may be several problems that need to be dealt with, for example the blocking of compressed UDP ports and non-standard application protocols used for 6LOWPAN applications, along with the unavailability of static IP addresses.

4. IPv4 interconnectivity: 6LOWPAN natively supports only IPv6, however often it will be necessary for 6LOWPAN nodes to interact with IPv4 networks. These are several ways to deal with IPv4 interconnectivity, including IPv6-in-IPv4 tunneling and address translation.

These nodes configured for that purpose on the Internet.

5. Security: When connecting embedded devices to the public Internet, security should always be a major concern as embedded devices are limited in resources and are autonomous. This is very much so with GLOWPAN as node and network limitations prevent the use of the full IPsec suite, transport layer ("socket") security or the use of sophisticated firewalls on each node. Although link-layer security inside a LOWPAN Router is still vulnerable - This increases the need for end-to-end security at the application layer.

Wireless

HART network

Wireless HART Communication Protocol Overview

As the need for additional process measurement increases, users seek a simple, reliable secure and cost-effective method

to deliver new measurement values to control systems without the need to run more wires. With process improvements, plant expansions, regulatory requirements and safety levels demands for additional measurements, users are looking to wireless technology for that solution.

With approximately 30 million HART devices installed and in service worldwide, HART technology is the most widely used field communication protocol for intelligent process instrumentation. With the additional capability of wireless communication, the legacy of benefits this powerful technology provides continues to deliver the operational insight users need to remain competitive.

Wireless HART

Backed by the power of HART

• Built on proven industry standards

• Created by industry and technology experts

• Multi-vendor support and interoperable devices

• Uses existing devices, tools and knowledge.

Flexible Applications

- Reduced installation costs - on wires!
- Process monitoring, control and asset management
- Health, safety and environmental compliance monitoring.

Supports All phases of the Plant Life Cycle.

- Fast engineering, deployment and commissioning
- Cost-effective move from scheduled to predictive maintenance
- Easy diagnosing and trouble shooting

Simple. Reliable. Secure.

Even though millions of HART devices are installed worldwide, in most cases the valuable information they can provide is stranded in the devices. An estimated 85% of all installed HART devices are not being accessed to deliver device diagnostics information with only the process variable data communicated via the 4-20mA analog signal. This is often due to the cost and the difficulty of accessing the HART information.

Wireless HART technology allows users to access the vast amount of unused information stranded in these installed HART smart devices - 85% of the installed HART devices. It also provides a simple, reliable and secure way to deploy new point of measurement and control without the wiring costs.

1. Simple

Wireless HART is a robust technology that is simple to implement. It enables users to quickly and easily gain the benefits of wireless technology while maintaining compatibility

Compatibility with existing HART devices, tools and systems.

Easy installation and commissioning

- Familiar tools, work flow and procedures.
- Multiple power options.
- Reduced installation and cabling costs.
- Coexistence with other wireless networks.
- Supports both star and mesh topologies.
- Add devices one at a time.

Automatic Network Features.

- Self-organizing and Self-healing
- Always-on security
- Adjust as new instruments are added
- Adjust to changes in plant infrastructure.

2. Reliable

Industrial facilities with dense infrastructures, frequent movement of large equipment, changing conditions, or numerous sources of radio-frequency and electromagnetic interference may have communication challenges. Wireless HART includes several features to provide built-in 99.9% end-to-end reliability in all industrial environments.

Standard Radio with Channel Hopping

- Radios comply with IEEE 802.15.4.4-2006
- 2.4 GHz license free frequency band
- "Hops" across channels to avoid interference
- Delivers high reliability in challenging radio environments.

Coexistence with other wireless networks.

- Clear Channel Assessments tests for available channels
- Blacklisting avoids frequently used channels
- Optimizes bandwidth and radio time
- Time Synchronization for on-time messaging.

Self-Healing Network

- Adjusts communication paths for optimal performance
- Monitors paths for degradation and repairs itself
- Finds alternate paths around obstructions
- Mesh network and multiple access points.

3. Secure

Wireless HART employs robust security measures to protect the network and secure the data at all times. These measures include the latest security techniques to provide the highest levels of protection available.

Protects valuable information

- Robust, multi-tiered always-on security
- Industry standard 128-bit AES encryption
- Unique encryption key for each message
- Data integrity and device authentication
- Rotate encryption keys used to join the network.

Protects Wireless Network

- Channel hopping
- Adjustable transmit power
- Levels
- Multiple levels of security keys for access
- Indication message integrity failures
- Reports message integrity failures
- Reports authentication failures
- Safe from Wi-Fi type internet attacks.

WIRELESS HART - HOW IT WORKS.

WIRELESS HART is a wireless mesh network Communications Protocols for process automation applications. It adds wireless capabilities to the HART Protocol while maintaining compatibility with existing HART devices, commands, and tools.

Each WIRELESS HART network includes three main elements:

- Wireless field devices connected to process or plant equipment.

This device could be a device with WIRELESS HART built in or an existing installed HART-enabled device with a WIRELESS HART adapter attached to it.

- Gateways enable communication between these devices and host applications connected to a high-speed backbone or other existing plant communications network.

- A Network Manager is responsible for configuring the network, scheduling communications between devices, managing message routes, and monitoring network health. The Network Manager can be integrated into the gateway, host application, or process automation controller.

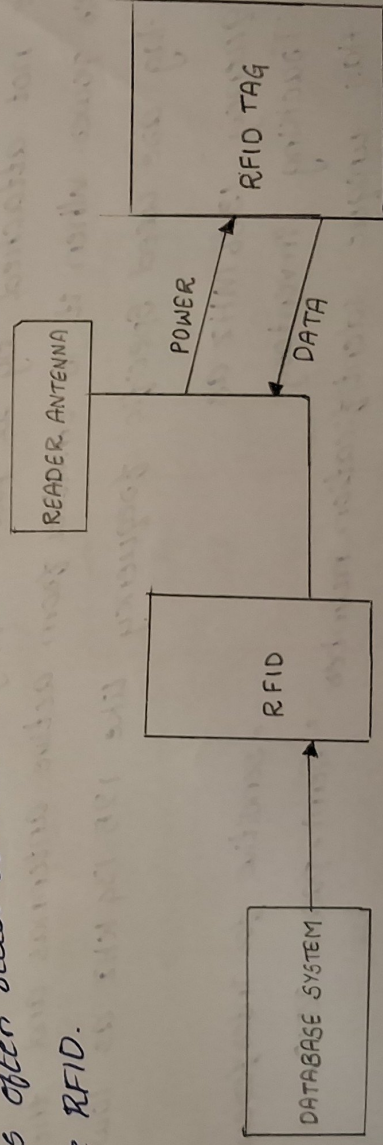
The network uses IEEE 802.15.4 compatible radios operating in the 2.4 GHz Industrial, Scientific, and Medical radio band. The radios employ direct-sequence spread spectrum technology and channel hopping for communication security and reliability, as well as TDMA synchronized, latency-controlled communications between devices on the network. This technology has been proven in field trials and real plant installations across a broad range of process control industries.

Each device in the mesh network can serve as a router for messages from other devices. In other words, a device doesn't have to communicate directly to a gateway, but just forward its message to the next closest device. This extends the range of the network and provides redundant communication routes to increase reliability.

RFID

Radio Frequency Identification (RFID) is a form of wireless communication that incorporates the use of electromagnetic or electrostatic coupling in the radio frequency portion of the electromagnetic spectrum to uniquely identify an object or person. It uses radio frequency to search, identify, track, and communicate with items and people.

RFID is a technology that uses electromagnetic fields to automatically identify and track tags attached to objects. These tags contain electronically stored information that can be read from several meters away, without requiring direct line-of-sight. RFID is commonly used in inventory management, asset tracking, access control, and supply chain logistics due to its efficiency and accuracy in tracking and managing items. It is a method that is used to track or identify an object by radio transmission over the web. Data is digitally encoded in an RFID tag which might be read by the readers. This device works as a tag or label using which data is read from tags that are stored in the database through the readers as compared to traditional barcodes and QR codes. It is often read outside the read of sight either passive or active RFID.



Types of RFID

There are many kinds of RFID, each with different properties, but perhaps the most fascinating aspect of RFID technology is that most RFID tags have neither an electric plug nor a battery. Instead, all of the energy needed to operate them is supplied in the form of radio waves by RFID readers. This technology is called passive RFID to distinguish it from the (less common) active RFID in which there is a power source on the tag.

- UHF RFID (Ultra-High Frequency RFID). It is used on shipping pallets and some driver's licenses. Readers send signals in the 902-928 MHz band. Tags communicate at distances of several meters by changing the way they reflect the reader's signals; the reader is able to pick up these reflections. This way of operating is called backscatter.

- HF RFID (High-Frequency RFID). It operates at 13.56 MHz and is likely to be in your passport, credit cards, books, and non-contact payment systems. HF RFID has a short-range, typically a meter or less because the physical mechanism is based on induction rather than backscatter.

- Passive RFID: Passive RFID tags do not have their own power source. It uses power from the reader. In this device, RF tags are not attached by a power supply and passive RF tag stores their power. When it is emitted from active antennas and the RF tag are used specific frequency like 125-134 KHz as low

frequency 1356 MHz as

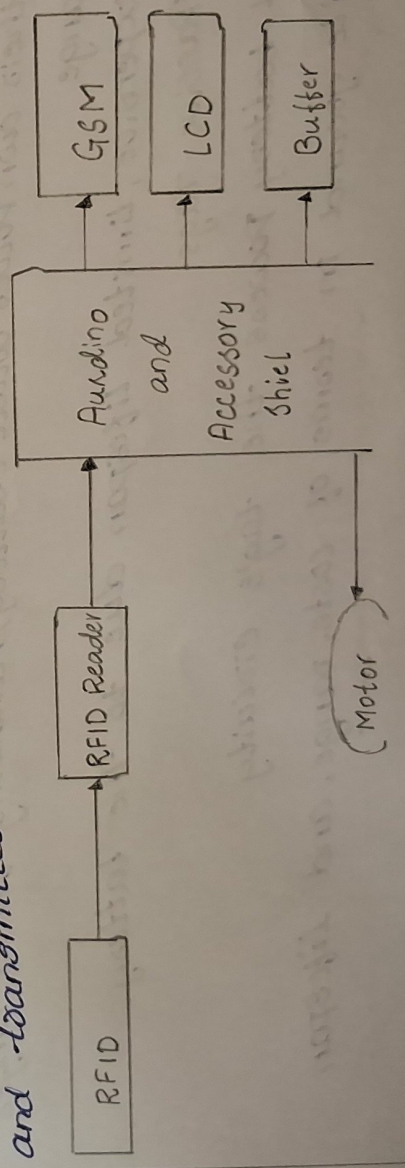
- Tracking inventory
- Has unique identification numbers
- sensitive for interference
- Semi-passive RFID.

Active RFID: In this device, RF tags are attached by a power supply that emits a signal and there is an antenna which receives the data. means, active tag uses a power source like battery. It has its own power source, does not require power from source/reader.

- Embedded Power: Communication over large distance
- Has unique identifiers/identification numbers
- Use other devices like sensors.
- Better than passive tags in the presence of metal.

These are also other forms of RFID using other frequencies, such as LF RFID (Low-Frequency RFID), which was developed before HF RFID and used for tracking.

Working Principle of RFID
Generally, RFID uses radio waves to perform AIDC function. AIDC stands for Automatic Identification and Data capture technology which performs object identification and collection and mapping of the data. An antenna is a device which converts power into radio waves which are used for communication between reader and tag. RFID readers retrieve the information from RFID tag which detects the tag and reads or writes the data into the tag. It may include one processor, package, storage and transmitter and receiver unit.



Every RFID System consists of three components: a scanning antenna, a transceiver and a transponder. When the scanning antenna and transceiver are combined, they are referred to as an RFID reader or interrogator. There are two types of RFID readers: fixed readers and mobile readers. The RFID reader is a network-connected device that can be portable or permanently attached. It uses radio waves to transmit signals that activate the tag. Once activated, the tag sends a wave back to the antenna, where it is translated into data.

The transponder is in the RFID tag itself. The read range for RFID tags varies based on factors including the type of tag, type of reader, RFID frequency and interference in the surrounding environment or from other RFID tags and readers. Tags that have a stronger power source also have a longer read range.

Types of RFID Tags

1. Passive Tags

- No built-in power source, relying on the RFID reader.
- Less expensive, longer lifespan, shortest read range (up to a few meters).

2. Active Tags.

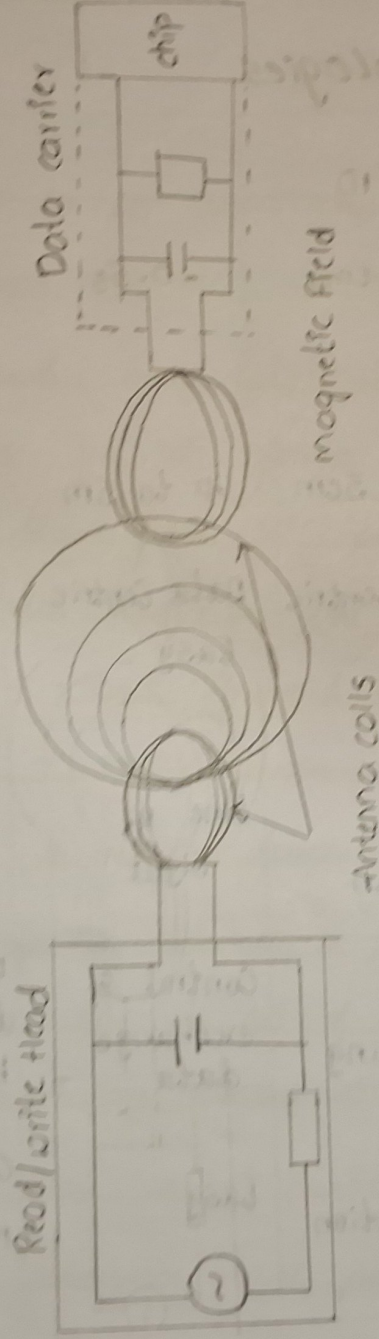
- Have their own power source (battery), allowing for a longer read range.
- More expensive, limited lifespan due to the battery.

3. Semi-Passive Tags.

- Small battery powers the tag's circuitry
- Middle ground in terms of costs, range, and lifespan.

Operation of NFC

- ▷ NFC devices communicate via magnetic field induction, where two loop antennas located within each other's near field, effectively forming an ad-hoc transformer.
- ▷ The reader continuously generates an RF carrier sine wave (at 13.56 MHz), which is always for modulation to occur. Detected modulation as the field would indicate the presence of a tag.



- ▷ A tag enters the RF field generated by the reader. Once the tag has received sufficient energy to operate covertly, it divides down the carrier and begins clocking its data to an output transistor, which is normally connected across the coil inputs.
- ▷ The tag's output transistor shunts the coil, sequentially corresponding to the data which is being clocked out of the memory array.

- ▷ Shunting the coil causes a momentary fluctuation (dampening) of the carrier wave, which is seen as a slight change in amplitude of the carrier.
- ▷ The reader peak-detects the amplitude modulated data and processes the resulting bit stream according to the encoding and data modulation methods used.

Comparison with Existing Technologies

Technology	Set-up time		Range		Usability		Selectivity		Use Cases		Consumer experience	
	< 0.1ms		Up to 10cm		Human Centric Easy, Intuitive, Fast		High, given, Security		Pay, get access, share, initiate service, easy set up.		Touch, wave, simply connect.	
	RFID		Up to 3cm		System centric Easy		Partly given		Item tracking		Get Information	
Bluetooth	- 0.5s		Up to 5m		Data Centric Easy		Line of sight		Control & exchange data		Easy	
	- 6 sec		Up to 30m		Data Centric Medium		Who are you?		Network for data exchange, headset.		Configuration needed.	

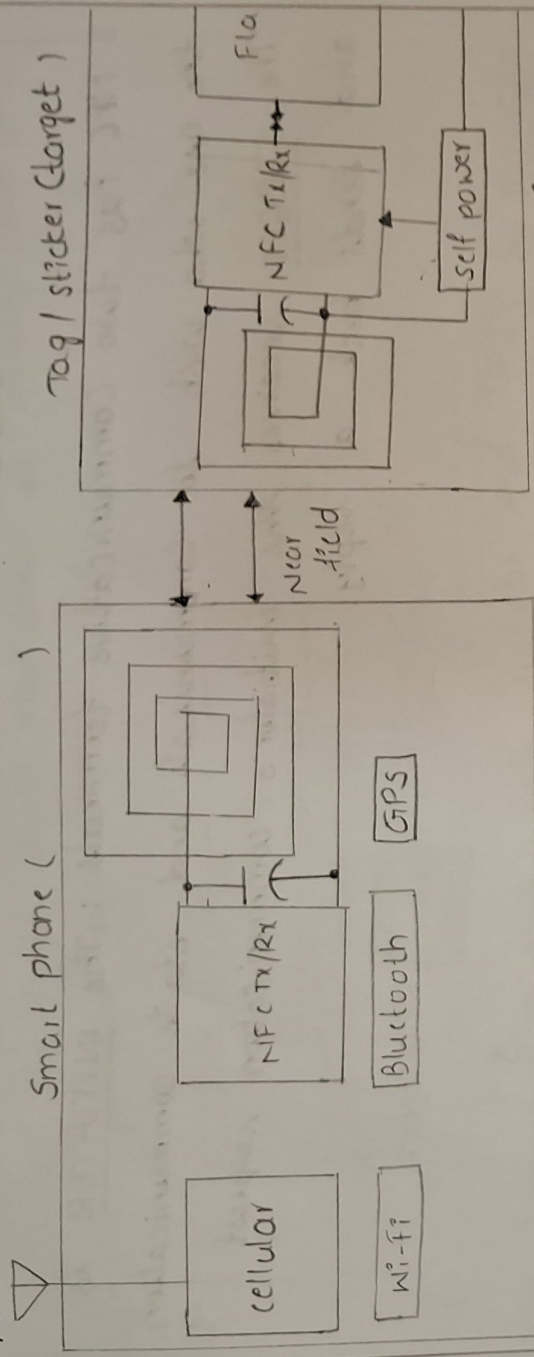
NFC Time Line

(16)

Year	Event
1983	The first patent to be associated with RFID was granted to Charles Walton.
2002	NXP Semiconductors and Sony co-invent NFC
2004	Nokia, Phillips and Sony established the NFC forum
2006	Initial Specifications for NFC forum Tags and "Smart Poster" records
2006	Nokia launches the first NFC phone (Nokia 6131)
2009	NFC Forum releases Peer-to-Peer standards
2010	Samsung, Nexus 5: First Android NFC phone shown
2010	AT & T, Verizon, T-Mobile announces softcard mobile payment joint venture
2011	Google I/O "How to NFC" demonstrates NFC to initiate a game and to share a contact, ^{URL, app, ...}
2011	RIM is 1st company for its devices is certified by Mastercard worldwide, the functionality of ^{Pay...}
2012	Samsung introduces TecTile, a set of NFC stickers and a Companion App for Android ^{write} to read
2012	Sony introduces "Smart Tags" using NFC technology to change modes & profiles on Sony
2012	Wired US is the 1st mass market publication to feature NFC enable advertisements
2013	Samsung and Visa announces major partnership to develop mobile payments.
2014	Apple introduces iPhone 6 with Apple Pay using NFC Technology.

(17)

Active and Passive NFC Devices.



Active NFC device usually a microcontroller based like a NFC enable Smartphone, would not only be able to collect information from NFC tags, but it would also be able to exchange information from NFC with other compatible phones or devices and could even alter the information on the NFC tag if authorized to make such changes.

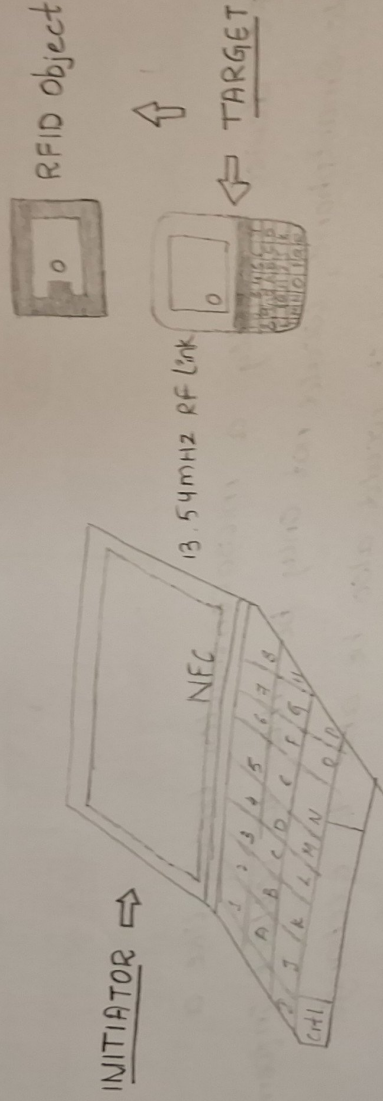
NFC reader continuously emits RF carrier signals and keeps observing the received RF signals for data.

A passive device, such as an NFC tag, contains information that other devices can read but read ~~any~~ any information itself. NFC tag does not have power source. It absorbs energy emitted by reader (phone) and starts sending modulated information when sufficient energy is acquired. The RF field generated by the reader. Data (0s and 1s) is accomplished by either direct modulation or FSK or phase modulation.

Operation: Active device generates the RF field and the passive devices use the field to power itself and co.

Operation of NFC

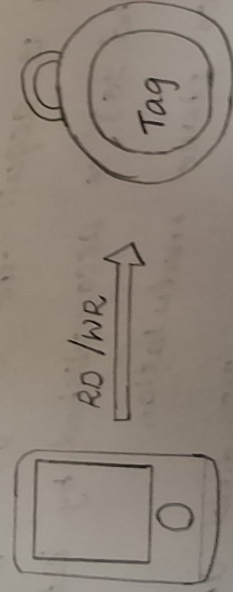
1. NFC has two Communicative terminals : The INITIATOR is the one who wish to Communicate and starts communication. The TARGET receives the initiator's communication request and Sends back a reply.



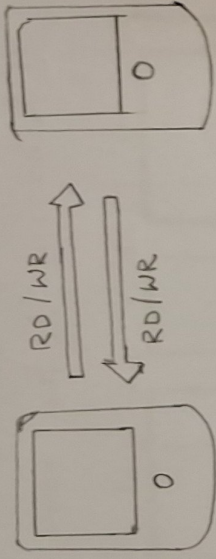
2. NFC employs two different coding to transfer data. If an active device transfers data at 106 kbit/s, a Modified Miller coding with 100% modulation is used. In all other cases Manchester Coding is used with a modulation ratio of 10%.

10% ASK modulation means ASK modulation in which modulation index will range between 8% and 14%. In 100% ASK modulation, unmodulated signal amplitude and modulated signal amplitude is almost equal.

There are two mode of communication



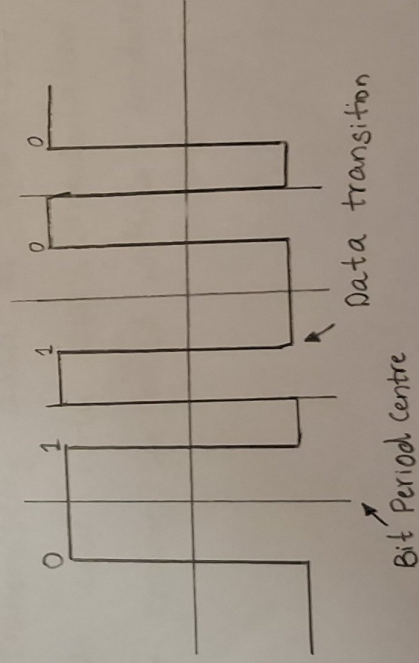
Passive Communication Mode: The initiator device provides a carrier field and the target device answers by modulating existing field. In this mode, the Target device may draw its operating power from the initiator provided electromagnetic field.



Active Communication Mode: Both initiator and Target device communicate by alternately generating their own field. A device deactivates its RF field while it is waiting for data. In this mode, both devices typically need to have a power supply.

NFC and Manchester Coding

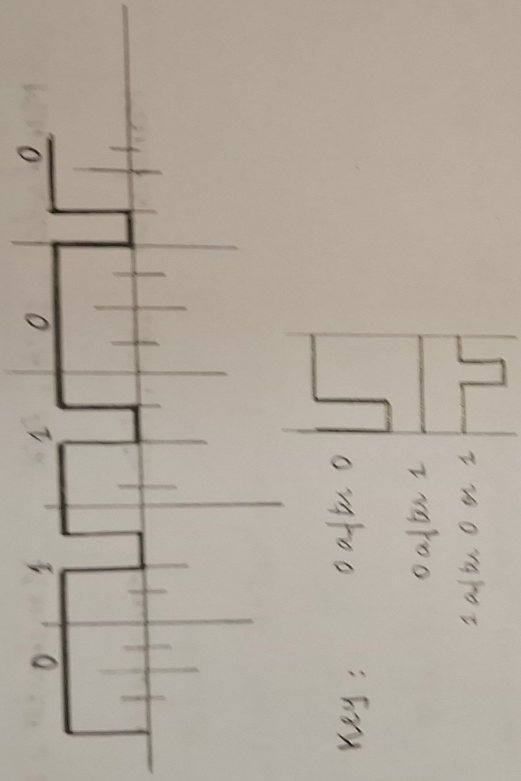
Manchester Coding is used for the majority of cases for the NFC communications. The Manchester coding utilises the two different transitions that may occur at the midpoint of a period. A low-to-high transition expresses a 0 bit, whereas, a high-to-low transition stands for a 1 bit.



Manchester Coding used for NFC data transfer.

NFC and Modified Miller Coding

Depending on the information to be transmitted, bits are coded as shown below. A high or "1" is always encoded in the same way, but a low or "0" is encoded differently dependent upon what preceded it.



Modified Miller Coding used for NFC data transfer used for 106 kbps active device transfers.

NFC RF Signal Coding

NFC employs two different coding systems on the RF signal to transfer data. In most cases a level of 10% modulation is used, with a Manchester coding format. However for an active device transmitting data at 106 kbps, a modified Miller Coding Scheme is used with 100% modulation. In all other cases Manchester coding is used with a modulation ratio of 10%.

DATA RATE KBPS	ACTIVE DEVICE	PASSIVE DEVICE
106	Modified Miller, 100%, ASK	Manchester, 10%, ASK
212	Manchester, 10%, ASK	Manchester, 10%, ASK
424	Manchester, 10%, ASK	Manchester, 10%, ASK

Logical Design of IOT:-

Logical design of a IOT system refers to an abstract representation of the entities and Process without going into the low-level specifics of the implementation.

Logical Design of IOT can be classified into 3 Blocks.

- 1) IOT Functional Blocks
- 2) IOT communication Models
- 3) IOT communication APP's

1) IOT Functional Blocks:-

An IOT system comprises of a number of functional blocks that provide the system like capabilities for identification, sensing, actuation, communication and management.

The functional blocks are described as follows:

- 1) Device: An IOT system comprises of devices that provide sensing, actuation, monitoring and control functions
- 2) Communication:- The communication block handles the communication for the IOT system
- 3) Services: An IOT system uses various types of IOT services such as services for device monitoring, device control services, data publishing.

4) Management:

- This block provides various functions to govern the IoT system.

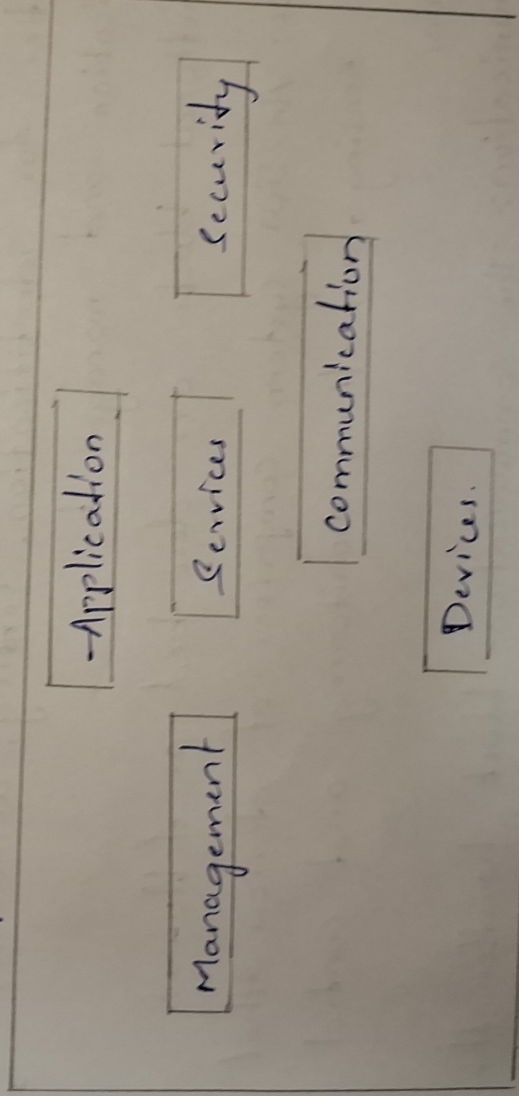
5) Security:

- It secures IoT system by providing functions like:

- * Authentication
- * Authorization
- * Message
- * and Data security

6) Applications:

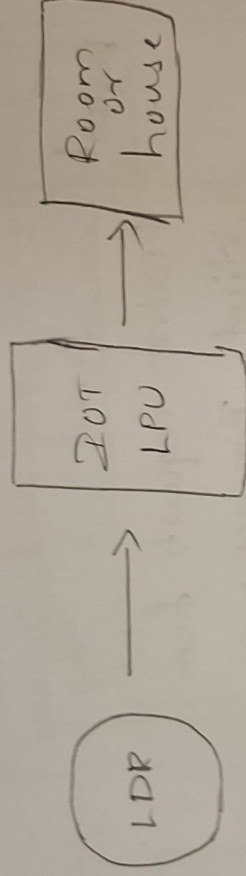
- It provides and interface that the user can use to control and monitor various aspects of the IoT system.



IoT communication model:-

* communication:- Transferring the data from one device to another device.

LDR:- Light Dependent Resistor



* There are four types of communication models

In IoT

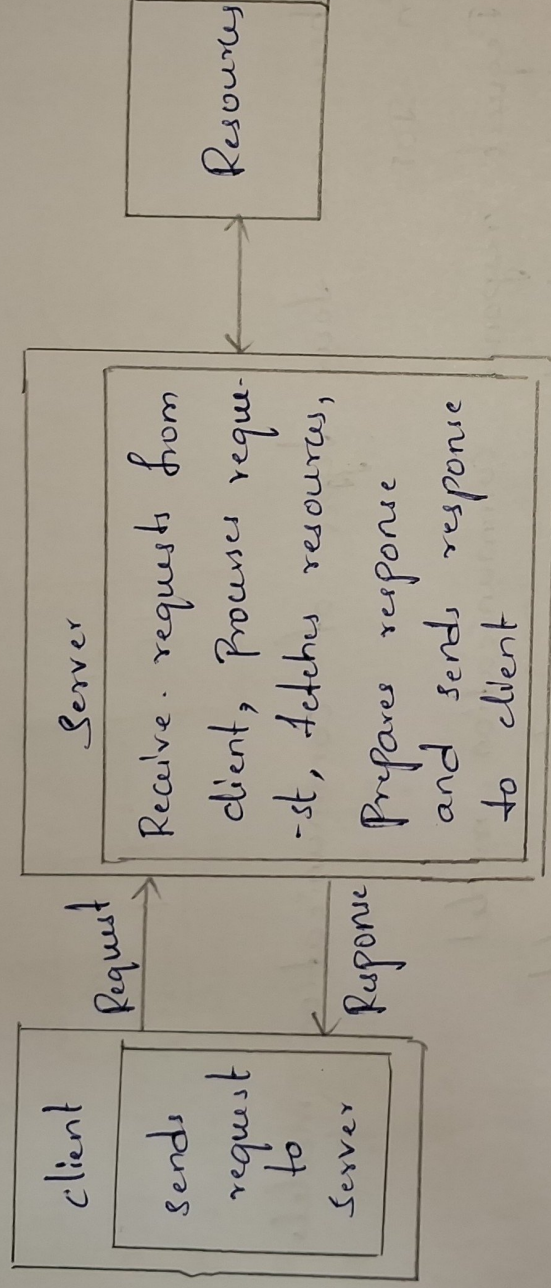
1. Request response communication model
2. Publish subscribe communication model
3. Push Pull communication model
4. Exclusive Pair communication model.

1. Request Response Communication model:-

• It is a communication model in which the client sends the request to the server and the server response to the request.

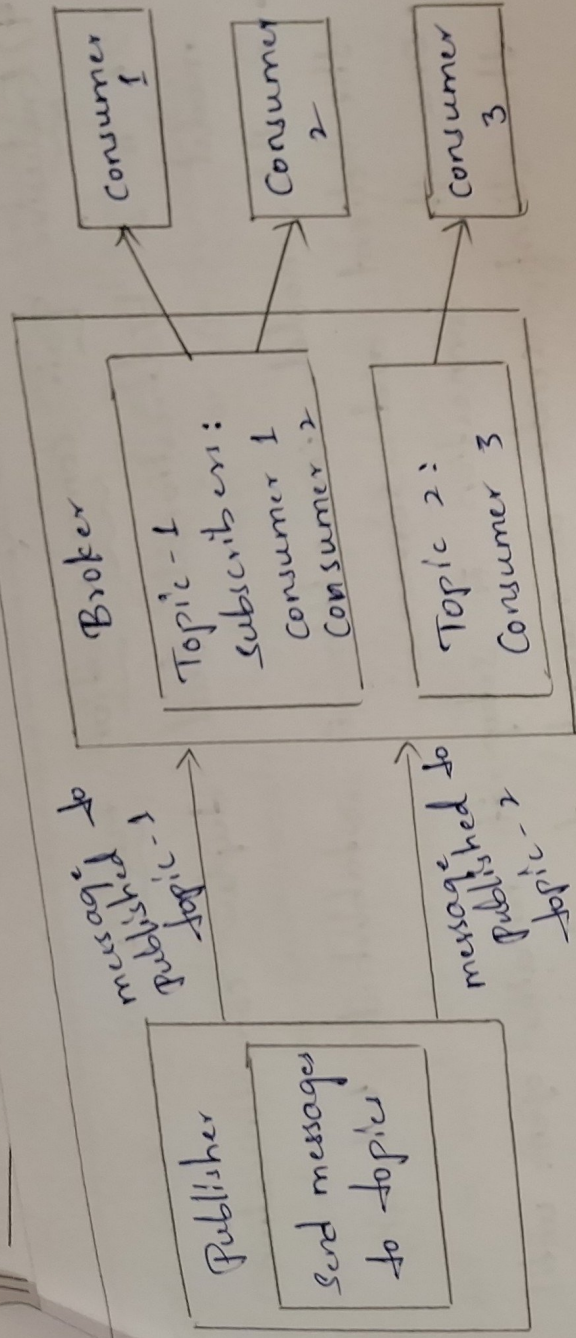
- When the server receives a request it decides how to respond, fetches the data, retrieves the resource representation, prepares the response and then response sends to the client.

- It is a stateless communication model and response pair is independent of others
- The request response communication model shown in below figure.



2. Publish subscribe communication model:-

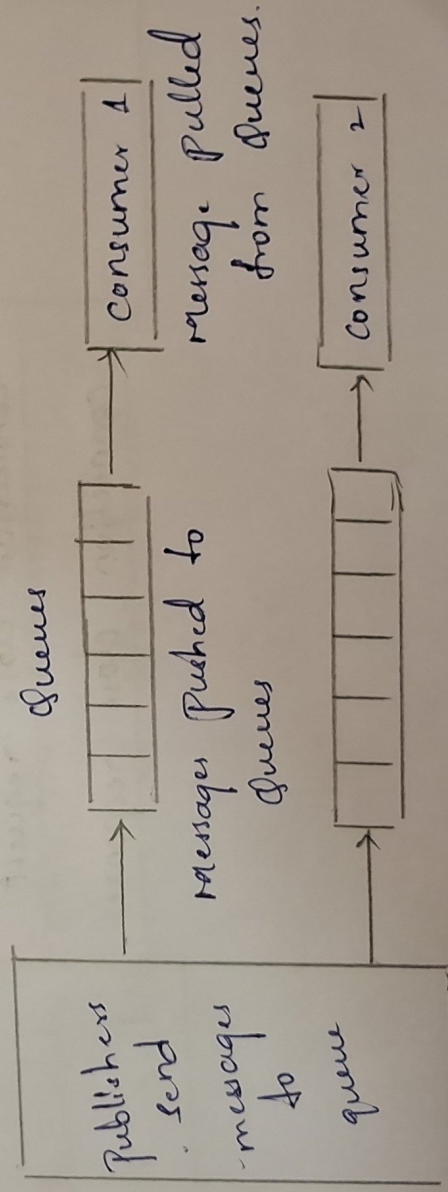
- It is a communication model that involves Publishers, brokers and Consumers
- Publishers are sources of data
- Publishers send the data to the topics which are managed by the broker
- Publishers are not aware of Consumers
- Consumer subscribe to the topics which are managed by broker.
- When the broker receives data for a topic from the Publisher sends the data to all subscribed Consumers.



3. Push-Pull communication model:-

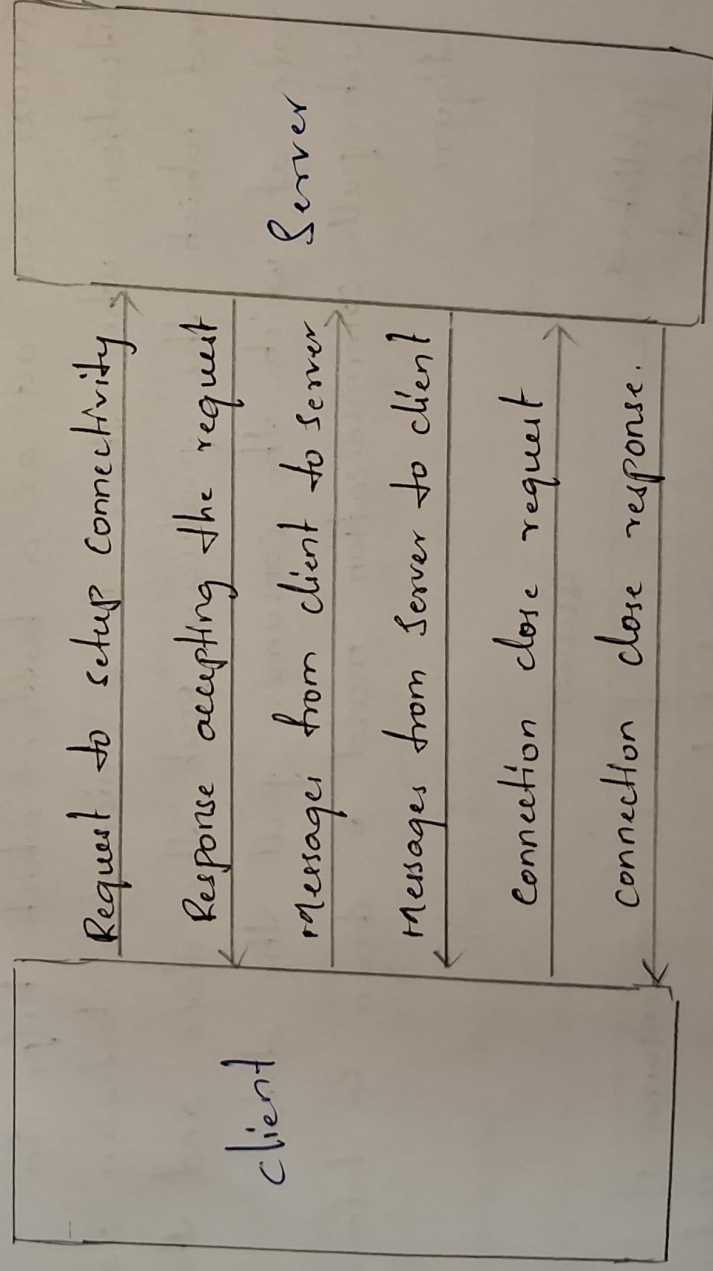
- It is a model in which the data producers push the data.
- Producers don't need to be aware of the consumers.
- Queues also act as a buffer which helps in situation when there is a mismatch between rate and which the producers push the data and the rate at which the consumers pull the data.
- Push-pull communication model is shown in below figure.

Figure.



4) Exclusive Pair Communication Model

- It is a bidirectional full duplex communication model.
- In this model connection is established between the client and server.
- Once the connection is setup it remains open until the client sends a request to close the connection.
- Exclusive pair is a stateful communication model and the server is aware of all the open connections.



DOT Communication APIs := (Applications Program Interface)

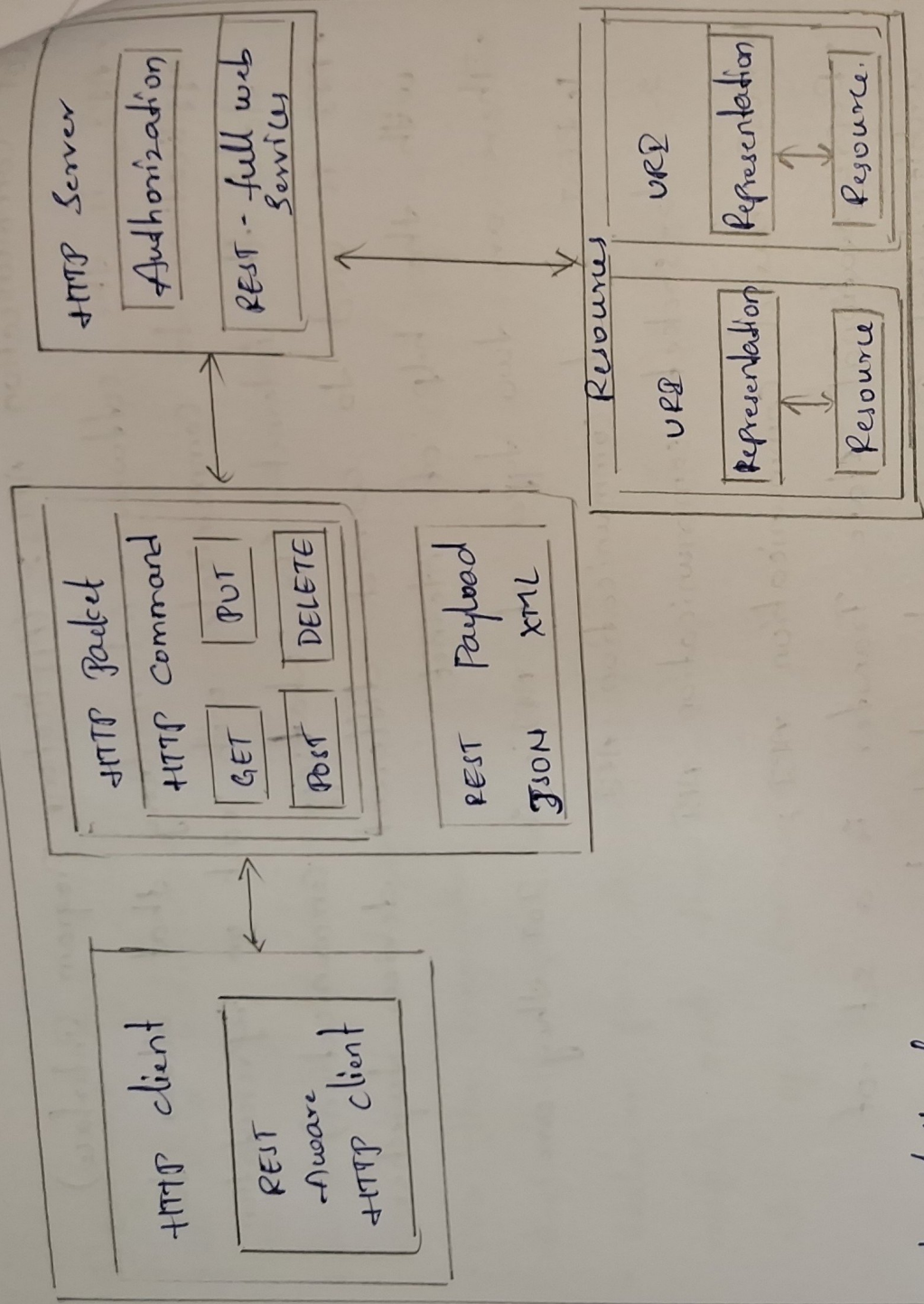
- API is the software intermediary that allows two applications to communicate with each other (representation state) the internet (Ex: whatsapp communication)
- API is used to connect different devices together with the help of internet

• There are two types of APIs in DOT, they are:

1. REST based communication API
2. web-socket communication API.

1. REST based communication API:

- Representational State Transfer is a set of architectural principles by which how resource states are addressed and transferred with the help of web services and APIs.
- It follows the request response communication model.
- It works based on HTTP protocol to exchange the data over the internet.
- The architecture of REST based API



The full form

- JSON - Javascript object notation
- XML - Extensible Markup Language
- URI - Uniform Resource Identifier

Features of REST based API:

- Constraints of REST based API.
- 1. client server separation:
- The principle behind the client-server constraint is the separation of concerns.

For example:

- client should not be concerned with the storage of the data of the server.
- Similarly client should not be concerned with the storage of data.

• Server should not be concerned user interface.

• Stateless -

Each request from client to server must contain all the information based on request and cannot take advantage of any previous or stored content.

• Layered system:

It contains the behaviour of components such that each component cannot see behind the intermediate layer with which they interfaced.

• Catchable

catch a constraint request the data within a response to a request be explicitly or explicitly labelled as catchable or non-catchable.

• Uniform Interface:

• It is the method of communication a client b/w a server to maintain uniform interface

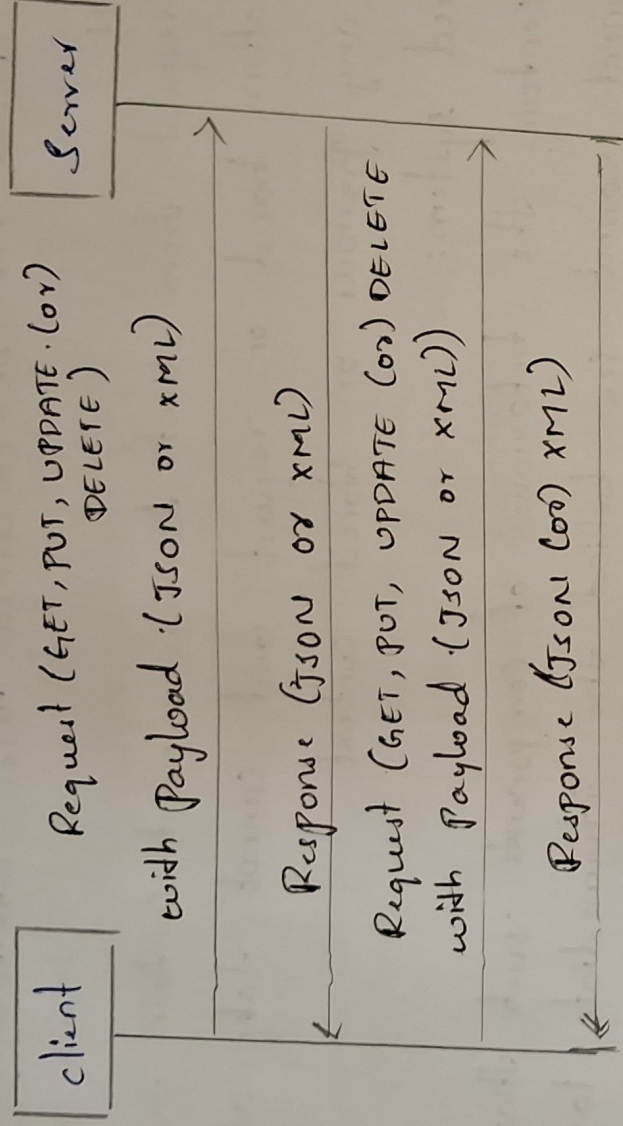
• Resources are identified if the requests and are themselves separate from the representations of the resources that written to client.

• code on demand:-

• Servers can provide executable code (or) scripts for clients to execute their content

• The request response model by REST API which is

shown in below.



2. Web-socket communication based API:-

- It allows bidirectional full-duplex communication between clients and servers.

- It follows the exclusive pair communication model.

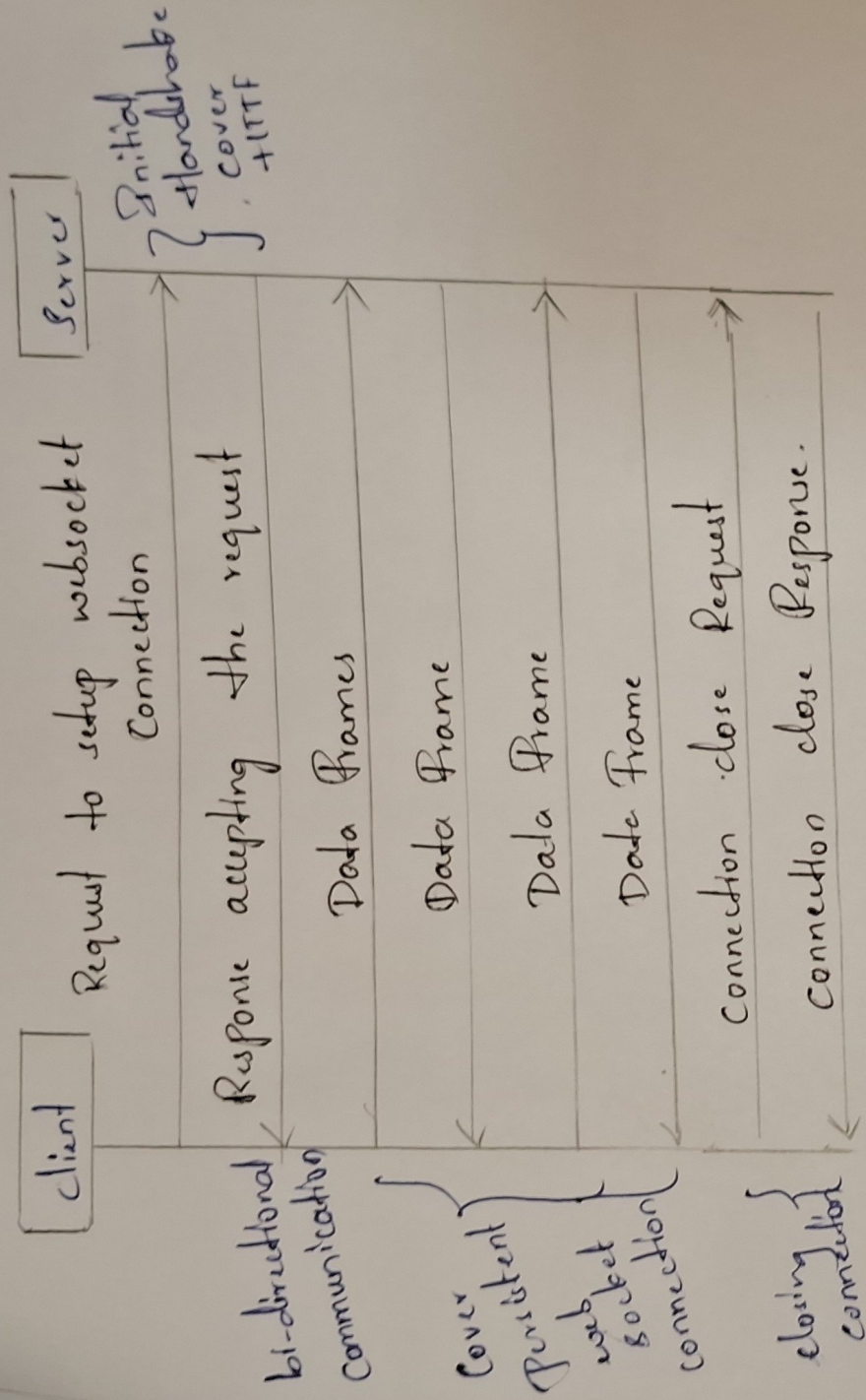
- Unlike request response API such as REST, web socket API allow full duplex communication don't require a new connection to be setup for each message to be sent.

- Web-socket communication with a connection setup request sent by the client to the server.

- Web socket APIs reduces the network traffic and latency.

- Web-socket is suitable for IoT application that have low latency and high throughput requirements.

Exclusive pair model used by web-socket API
is shown in below.



Introduction to the YANG Modelling Language :-

* It is a language that is designed to be readable by both humans and machines in order to model configuration and state information.

* YANG is defined in the following RFCs.

- RFC 6020 - YANG - A data modeling language for the Network Configuration Protocol
- RFC 6021 - Common YANG Data Types
- RFC 7950 - The YANG 1.1 Data Types Modeling Language

→ A YANG model defines a tree structure and data is mapped into this tree.

→ A model is defined in a text file and comprises a Module and optionally Submodules, which when compiled together form the tree.

Create Your own YANG Model :-

* In Addition to the standard type definitions detailed in RFCs, personalized type definitions can be created with YANG modules.

* Create a files called engineer - type .yang.

The contents of the file should be as follows.

IOT

Lock → Lock a configuration data store for exclusive edits by a client.

unlock → Release the lock on a configuration database.

get - schema → This operation is used to retrieve a schema (plan) from the NET CONF Server.

Commit → Commit the candidate configuration as the device's new current configuration.

close - session → Gracefully terminate a NET CONF Session.

kill - session - Forcefully terminate a NETCONF Session.

The Context layer consists of configuration and state data which is XML Encoded.

The Schema of the configuration and state is defined in a data modeling language called YANG.

(Let Another next Generation).

1. A module name - This name is defined in the module - Engineer types section with Engineer types. being the name of the new YANG module.

2. A prefix - This is short name that can be used with in YANG module to quickly reference the module.

3. A revision number - This is in the format of a date and should be adjusted when a change is made to the module.

The revision number is returned to the operator when some specific RPCs are made.

4. A YANG version - This is not the version number of the module but the version number of the YANG definition that the author has written the module to. Currently this would be 1.0 or 1.1.

This module also contains two elements called types def. These are as the name would suggest type definitions.

* They define custom types through the use of standardized YANG elements.

* Standardized types as defined in the RFCs could have been used but for the purpose of this tutorial we will define our own. Yang is a data modeling language.

The YANG model defines a hierarchical data structure, which can be used for operations based on


```

module engineer-types {
  yang-version 1.1;
  namespace "urn:nokia-com"; sr-expects-types;
  prefix "types";
  revision 2019-05-06;
  type-def age-type {
    description "Engineers start work at
    18 and should be retired by 110";
    type int0 {
      range "18...110";
    }
    units years;
  }
}

```

```

types of transportation-type {
  description "method of transportation";
  type enumeration {
    enum foot;
  }
}

```

```

enum cycle;
enum car;
enum bus;
enum boat;
enum aeroplane;
}
}
}

```


Elements of the YANG Model :-

1. module $\rightarrow$ YANG constructs a data model as a module.
-e. The module name is the same as the YANG file name. A module can import data from other modules and reference data from submodules..
2. namespace $\rightarrow$ Name Space of the module, which is a Globally Unique URI. Namespace are used during XML encoding of data.
3. prefix $\rightarrow$ Abbreviation of a namespace, which must be unique.
4. organisation $\rightarrow$ Name of the organization to which YANG belongs.
5. Contact $\rightarrow$ Contact information of the YANG module.
6. description $\rightarrow$ Functions of the YANG module.
9. Revision $\rightarrow$ Version information of the YANG module, providing the version editing history of module.
8. Container $\rightarrow$ Container node used to group related nodes into a subtree.
9. list $\rightarrow$ List node, used to define a sequence of list entities.
10. leaf $\rightarrow$ leaf nodes, which contains simple data such as an integer or character string.

on network configuration. management protocols.

The operations include configuration, status data, remote procedure calls (RPCs) and notifications.

Compared with the SNMP model MIB, YANG is more hierarchical can distinguish between configuration and status and provides high extensibility.

YANG is used to build data models.

YANG is defined in the following RFC standards.

1. RFC 6020 : In 2010 : IETF defined YANG for the.

First time YANG is a data modeling for NETCONF.
2. RFC 6024 : In 2010, The IETF defined various data types commonly used in network communication techniques.

This allows us to import and used predefined network data types without redefining them when building YANG models.

3. RFC 6991 : In 2013, the IETF added data type to the YANG Model on the basis of RFC 6021.

4. RFC 7950 : In 2016, The IETF released YANG 1.1 to correct ambiguity and defects in the initial sessions (RFC 6020).

YANG version of the Toaster - MIB (management information).

module toaster {

yang-version 1;

name-space "https://example.com/ns/toaster";

prefix toast;

revision "2009-11-20" {

description

"Toaster module";

}

identity toast-type {

description

"Berc for all bread types";

}

identity white bread {

based-toast : toast-type;

} identity wheat-bread {

base toast-type;

}

identity wonder-bread {

base toast-type;

}

identity frozen-waffle {

base toast-type;

}

identity frozen-bagel {

base toast-type;


```

    }
    identify hash - brown {
    }
    base toast type;
    type def display string {
    type string {
    length "0 .... -55-";
    }
    }
    container toaster {
    presence
    " indicates the toaster service is available ";
    description
    "Top level container for all toaster database
    objects";
    leaf toaster manufacturer {
    type display string;
    config false;
    mandatory true;
    }
    leaf toaster model number {
    type display string;
    config false;
    mandatory true;
    }
    leaf toast status {
    type enumeration {

```



```

enum "up" {
    value 1;
}
enum "down" {
    value 2;
}
config false;
mandatory true;
}
rpc make-toast {
    input {
        leaf toaster_pureness
            type uint-32 {
                range '1...10';
                default '5';
            }
        leaf toaster_type {
            type identity ref {
                base toast : toast-type
            }
            default "wheat bread";
        }
    }
}
rpc cancel-toast {

```


description

"Stop making toast, if any is being made";

}

notification toast box {

leaf toast status {

type enumeration {

enum "done" {

value 0;

description "The toast is done";

}

enum "cancelled"; {

value 1

description

"The toast was cancelled";

}

enum

"error" {

value 2;

description

"The toaster service was disabled
the toaster is broken";

}

}

IoT System Management with NETCONF - YANG :-

Below fig the Generic Approach of IoT device management with NETCONF - YANG.

Roles of various Components :-

1. Management System :- The operator uses a management system to send NETCONF message to configure the IoT device and receive state information & notifications from the device as NETCONF messages.
2. Management API :- Management API allows management applications to store NETCONF sessions, read & write configuration data, read state data, retrieve configuration and invoke RPCs, programmatically in the same way as an operator can.

3. Transaction Manager :- Transaction manager executes all the NETCONF transactions and ensures that the ACIDC Atomicity, consistency, Isolation, durability, properties hold true for the transactions.
Atomicity properly ensures that a transaction is executed either completed or not at all.

Consistency properly ensures that a transaction brings the device configuration from one valid state to another. Isolation property ensures that concurrent execution of transaction results in the same.

device configuration as its transactions were executed serially in order.

Durability property ensures that transaction once committed will persist.

4. Roll back - manager :- Roll back manager is responsible for generating all the transactions necessary to roll back a current configuration to its original state.

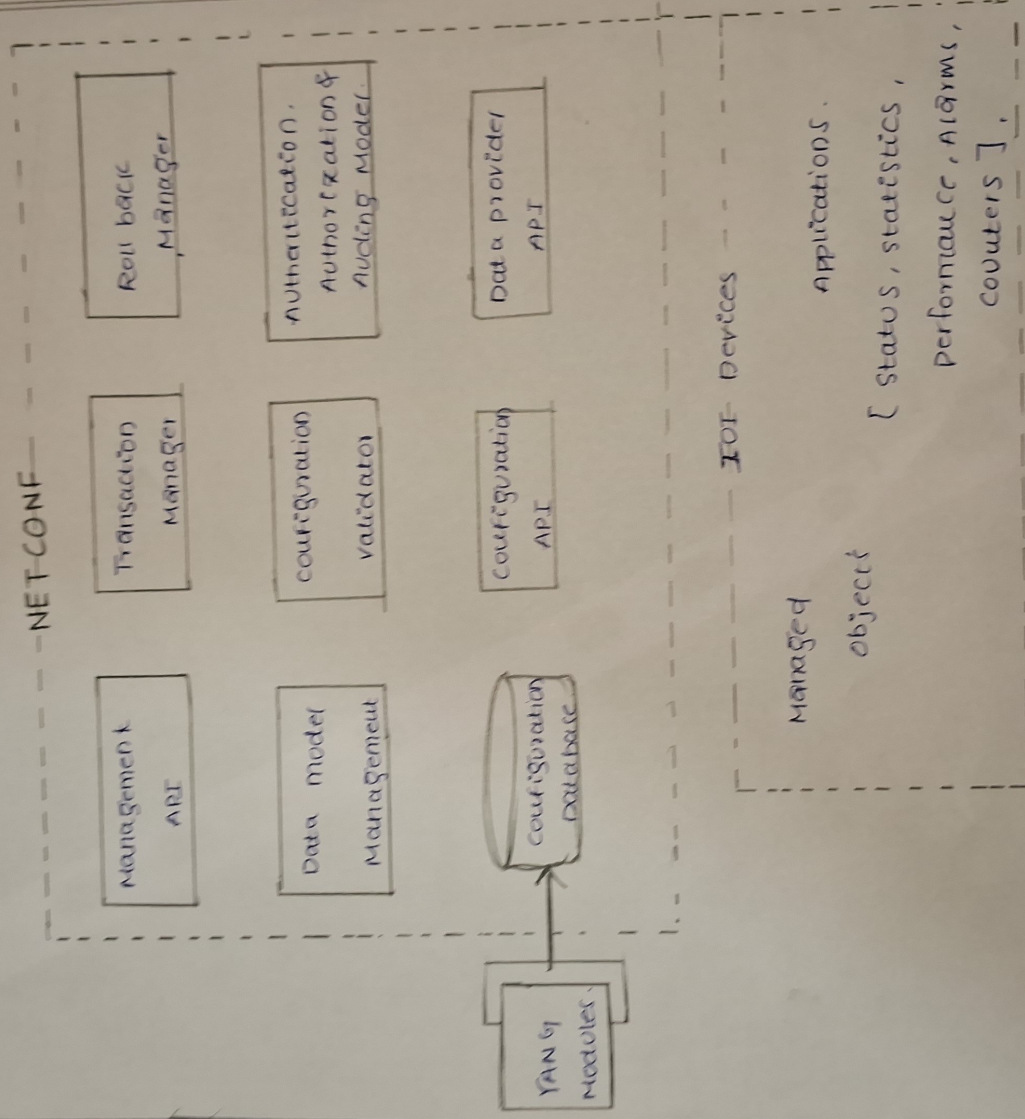
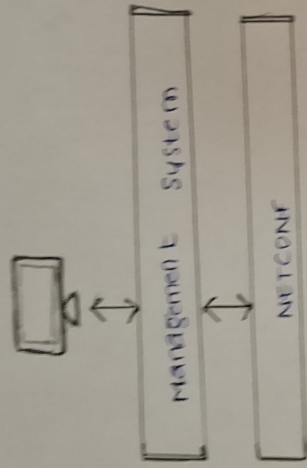
5. Data model manager :- The data model manager keeps track of all the YANG data models and the corresponding managed objects. The data model manager also keeps track of the applications which manager also keep data for each.

6. Configuration validate :- Configuration validator checks if the resulting configuration after applying a transaction would be avoid configuration.

7. Configuration Database :- This database contains both the configuration and operational data.

8. Configuration API :- Using the configuration API the application on the IoT device can read configuration data from configuration data store and write operational data to the operational data base.

9. Data provider API :- Applications on the IoT device can register for call backs for various events using the data provider API. through the data provider API,



Components of the Arduino UNO:

1) USB port :- All arduino boards feature a USB. There is a need to connect these boards to a power source. Therefore the USB cable from a computer can power the Arduino UNO PCB. Also this board can be powered by a power supply in a barrel jack.

(Also this board can).

Arduino UNO PCB comes with barrel jack and a USB connection cable.

Another function of the USB connection is that it enables users to load code on the board.

a) Reset button:-

The Arduino UNO PCB features a reset button. If you intended to connect the reset pin to ground, you just need to push the button and then restart code loaded on the board.

main IC:-

The main integrated circuit of the Arduino UNO board looks like metal legs. The component is the brain of the Arduino UNO board.

The main IC of the Arduino UNO PCB is different from another Arduino board.

However the main IC of the Arduino boards always comes from the ATmega line

Voltage Regulator:- Another important component on the UNO is the voltage regulator. This component regulates the amount of the voltage that passes into the board. The Arduino UNO PCB has its $\pm$ voltage limit, i.e. 20V. Avoid connecting this board to a device with more than 20V.

Tx Rx LEDs:-

Rx Simply means receive while Tx means Transmit. With these LEDs, users can know when the board is transmitting or receiving data.

Pins:- Every Arduino board comes with pins. However, the number of pins may differ according to the types of board.

The pins provide space for connecting wires to develop a circuit.

There are different types of pins on the Arduino UNO PCB. Each of these pins is labeled.

Crystal Oscillator or Ceramic Resonator:-

A microcontroller like the Arduino board requires a clock source. The operation of the microcontroller depends on the clock circuit.

The fundamental program elements of an Arduino sketch are structured around two main functions

and other supporting elements

1. Structure:-

(i) Setup():- This function runs only once when the Arduino board powers on or resets. It is used for initial configurations, such as setting pin modes (input or output); initializing serial communication or setting up libraries.

C++

```
void setup()
```

```
{
```

```
  // initialization code here
```

```
  pinMode(13, OUTPUT);
```

```
}
```

(ii) loop():- This function runs repeatedly after the setup() function has completed. It contains the main logic of the program, continuously reading inputs, processing data and controlling outputs.

Ex:-

C++

```
void loop() {
```

```
  digitalWrite(13, HIGH); // Turn on LED connected
```

```
  delay(1000); // wait for 1 second i.e. 1000ms
```

```
  digitalWrite(13, LOW); // Turn off LED
```

```
  delay(1000); // wait for 1 second
```

```
}
```


2. variables and constants:-

variables are used to store data that can change during program execution (Ex: int, float, boolean, char)

C++ `int ledPin = 13; // Declaring an integer variable`
constants are used to store values that remain fixed throughout the program

C++ `const int Button pin = 2; // Declaring a constant integer`

3. Built in functions:-

Arduino provides a rich set of built in functions for various tasks, such as digital I/O (digitalRead(), digitalWrite()), analog I/O (analogRead(), analogWrite()), Serial communication (Serial.begin(), Serial.print()), time management (delay(), millis()), and more

4. Control Structures:-

* conditional statements :- if, else if, else for making decision based on conditions.

* looping statements :- for, while, do while for repeating blocks of code.

5. operators:-

* Arithmetic operators:- $+$, $-$, $*$, $/$, $\%$

* comparison operators:- $=$, $!=$, $<$, $>$, $<=$, $>=$

* logical operators: $\&\&$, $(\&\&)$, $||$ (OR), $!$ NOT

Libraries in Arduino:-

The Arduino environment can be extended through the use of libraries.

libraries provide extra functionality for use in Sketches. 8157 libraries are available in Arduino

Display

Lvgl:- full featured Graphics Library for embedded Sys.

fast LED:- multi platform library for controlling dozens of different types of LEDs along with optimized math, effect and noise functions.

Arduino - Interrupts

Interrupts stop the current work of Arduino such that some other work can be done.

EX:- Suppose you are sitting at home, chatting with someone, suddenly the telephone rings, you stop chatting & pick up the telephone to speak

to the caller, when you have finished your telephonic conversation, you go back to chatting with the person before the telephone rang.

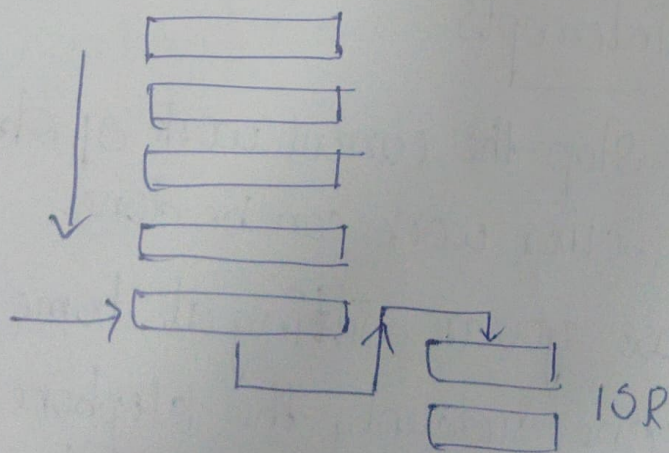
Similarly you can think of the main routine as chatting to someone, the telephone ringing causes you to stop chatting. The Interrupt Service routine is the process of talking on the telephone, when the telephone conversation ends, you then go back to your main routine of chatting.

This example explains exactly how an interrupt causes a processor to react.

The main programming is running and performing some function in a circuit.

However when an interrupt occurs the main program halts while another routine is carried out.

When this routine finishes, the processor goes back to the main routine again.



Features

- * Interrupts can come from various Sources. In this case we are using a hardware interrupt that is triggered by a state change on one of the digital pins.
- * Most Arduino designs have two hardware interrupts referred to as "interrupt 0" and "interrupt 1" hard wired to digital I/O pins 2 and 3 respectively.
- * Arduino mega has six h/w interrupts including the additional interrupts ("Interrupts 2" through "interrupt 5") on pins 21, 20, 19 and 18.
- * You can define a routine using a special function called as interrupt service routine (ISR)
- * It is possible to have that function executed automatically, each time an event happens on an input pin

Display

lvgl

— full featured Graphics library for embedded systems fast LED

multiplatform library for controlling dozens of different types of LEDs with optimized mark, effect and noise functions.

Arduino Tson:-

A single and efficient TSON Library for embedded C++

Wifimanager:-

Wifi configuration manager with web config portal for espressif boards, by EZape.

IR remote:-

Send and receive infrared signals with multiple protocols

Blinker:-

Build a easy way for your IoT project
Blinker project for embedded n/w, support
Arduino, E/

TFT-eopl:-

TFT graphics library for Arduino processors
With performance optimization for R2040, STM32,
ESP8266

pubsubclient:-

A client library for MQTT messaging

Blynk:-

(A client lib) Build a smartphone app for
your project in minutes

Adafruit Neo pixel:-

Arduino library for controlling single wirebased LED pixels and strip

1P1ON:-

It is an opensource, multi-master, multimedia bus network protocol

Tiny Gsm:-

A small Arduino library for GPRS modules that just works

DHT Sensor library:-

Arduino library for DHT11, DHT22, etc

RadioLib:-

universal wireless communication library.

LORA:-

An Arduino library for sending & receiving data using LORA radio.

There are two types of Interrupts

1) H/w:- They occur in response to an external event, such as an external interrupt, pin going to high or low

2) S/w:- They occur in response to an instruction

Sent in S/w - the only type of interrupt that the Arduino language supports is the attachInterrupt() function.

program:-

```
attachInterrupt (digital pin . To interrupt (pin),  
                ISR, mode);
```

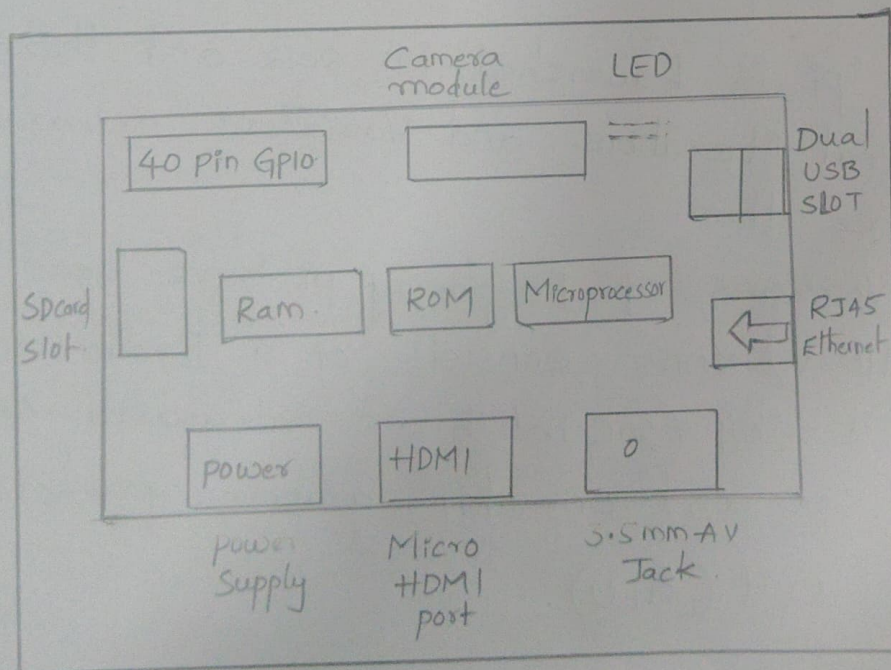
```
attachInterrupt (pin, ISR, mode).
```


Raspberry-pi :-

Raspberry-pi is the name of the "credit card sized computer board" developed by the Dept in the U.K.

- It gets plugged in a TV or monitor and provides a fully functional computer capability. It is aimed at imparting knowledge about computing to even younger students at the cheapest possible price.
- Although it is aimed at teaching computing to kids, but can be used by everyone willing to learn programming, the basics of computing, and building different projects by utilizing its versatility.
- Raspberry pi is developed by Raspberry pi foundation in the united kingdom.
- The Raspberry pi is a series of powerful, small single-board computers.
- Raspberry pi is launched in 2012 and there have been several iterations and variations released since then.
- Various versions of Raspberry pi have been out till date. All versions consists of a Broadcom System on a chip (SOC) with an integrated ARM-compatible CPU and on-chip graphics processing unit (GPU).

- The original device had a single-core processor. Speed of device ranges from 700MHz to 1.2GHz. and Memory range from 256MB to 1GB RAM.
- To store the operating system and program memory, Secure Digital (SD) cards are used.
- Raspbian OS, which is a Linux operating system, is recommended OS by Raspberry Pi Foundation.
- Some other third-party operating systems like RISC OS Pi, Diet Pi, Kali, Linux can also be run on Raspberry Pi.
- It also provides a set of general purpose input/output pins allowing you to control electronic components for physical computing and explore the IoT.



∴ Raspberry Pi

Raspberry pi Model -

There have been Many generations of raspberry pi From pi 1 to pi 4.

There is generally a model A and model B.

Model A is a less Expensive variant and it trends to have reduce RAM and dual cores such as USB and Ethernet.

List of Raspberry pi models and releases years:-

- 1) pi 1 model B - 2012.
- 2) pi 1 model A - 2013.
- 3) pi 1 model B⁺ - 2014.
- 4) pi 1 model A⁺ - 2014.
- 5) pi 2 model B - 2015.
- 6) pi 3 model B - 2016.
- 7) pi 3 model B⁺ - 2018.
- 8) pi 3 model A⁺ - 2019.
- 9) pi 4 model A - 2019.
- 10) pi Model B - 2020.
- 11) pi 400 - 2021.

Specs of the computer :- The computer has a quad core ARM processor that doesn't support the same instruction as an x86 desktop cpu.

→ It has 1GB of RAM, one HDMI port, four USB ports, one Ethernet connection, Micro SD slot for storage, one combined 3.5mm audio/video port,

and a Bluetooth connection.

→ It has got a series of input and output pins that are used for making projects like - home security cameras, Encrypted Door lock etc..

* Versatility of Raspberry pi :-

It is Indeed a Versatile computer and can be utilized by people from all age groups, it can be used for watching videos on youtube, Watching movies, and programming in Languages like python, Scratch, and Many more.

→ As Mentioned above it has a series of i/o pins that give this board the ability to interact with its environment and hence can be utilized to build really cool and interactive projects.

* Examples of projects :-

It can be turned into a Weather station by connecting some instruments to it for check the temperature, wind speed, humidity etc..

→ It can be turned into a home surveillance system due to its small size; by adding some cameras to it the security network will be ready.

→ If you love reading books it can also become a storage device for storing thousands of

Devices through the internet by using this device.

Conclusion:-

concluding the article it is convenient to ~~assert~~ that it is a small and powerful computer at a dirt-cheap rate and can handle most of the task as a desktop computer.

IoT Physical Servers & Cloud Offerings

5.1 Introduction to Cloud Storage Models & Communication APIs

—

5.2 WAMP - AutoBahn for IoT

5.3 Xively Cloud for IoT

5.4 Django

5.5 Designing a RESTful Web API

5.6 Amazon Web Services for IoT

5.7 SkyNetIoT Messaging Platform.

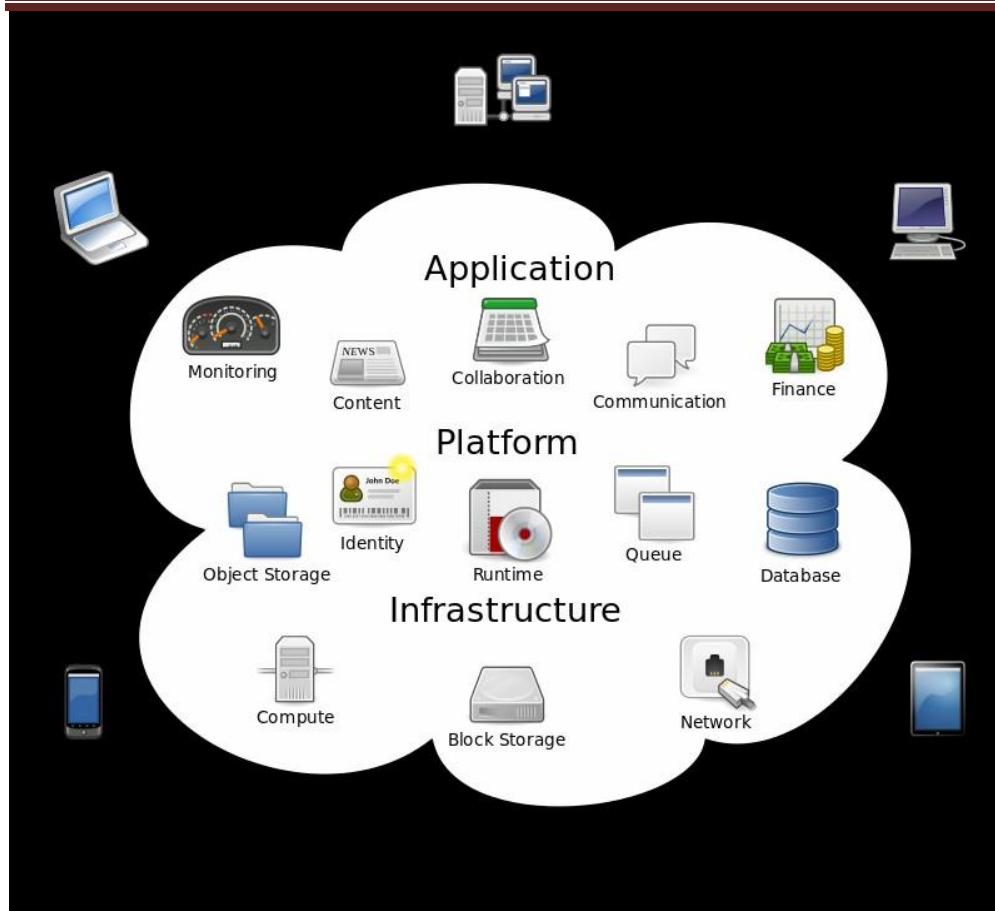


5.1 Introduction to Cloud Storage Models & Communication APIs

In truth, cloud computing and IoT are tightly coupled. The growth of IoT and the rapid development of associated technologies create a widespread connection of —things. This has led to the production of large amounts of data, which needs to be stored, processed and accessed. Cloud computing as a paradigm for big data storage and analytics. While IoT is exciting on its own, the real innovation will come from combining it with cloud computing. The combination of cloud computing and IoT will enable new monitoring services and powerful processing of sensory data streams. For example, sensory data can be uploaded and stored with cloud computing, later to be used intelligently for smart monitoring and actuation with other smart devices. Ultimately, the goal is to be able to transform data to insight and drive productive, cost-effective action from those insights. The cloud effectively serves as the brain to improved decision-making and optimized internet-based interactions. However, when IoT meets cloud, new challenges arise. There is an urgent need for novel network architectures that seamlessly integrate them. The critical concerns during integration are quality of service (QoS) and quality of experience (QoE), as well as data security, privacy and reliability. The virtual infrastructure for practical mobile computing and interfacing includes integrating applications, storage devices, monitoring devices, visualization platforms, analytics tools and client delivery.

Cloud computing offers a practical utility-based model that will enable businesses and users to access applications on demand anytime and from anywhere.





Deployment models

Deployment in cloud computing comprises four deployment models: private cloud, public cloud, community cloud and hybrid cloud.

A cloud storage API is an application program interface that connects a locally-based application to a cloud-based storage system, so that a user can send data to it and access and work with data stored in it. To the application, the cloud storage system is just another target device, like tape or disk-based storage. An application program interface (API) is code that allows two software programs to communicate with each other. The API defines the correct way for a developer to write a program that requests services from an operating system (OS) or other application. APIs are implemented by function calls composed of verbs and nouns. The required syntax is described in the documentation of the application being called

Three basic types of APIs

APIs take three basic forms: local, web-like and program-like.

1. **Local APIs** are the original form, from which the name came. They offer OS or middleware services to application programs. Microsoft's .NET APIs, the TAPI (Telephony API) for voice applications, and database access APIs are examples of the local API form.

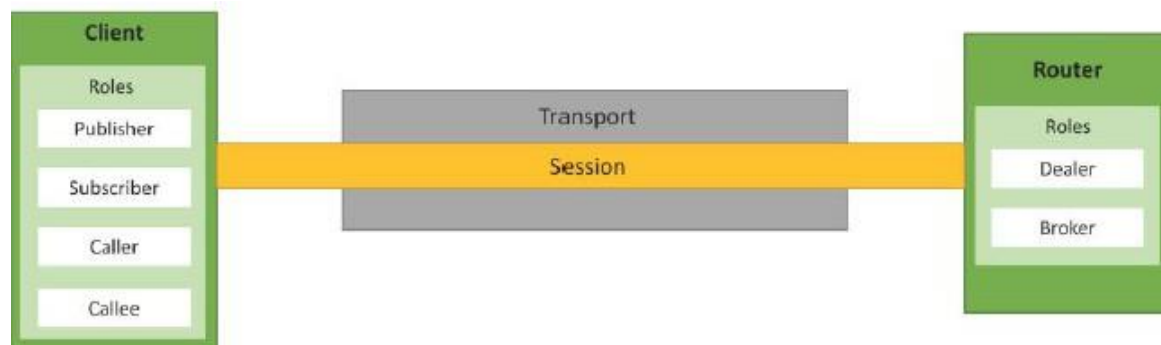


2. **Web APIs** are designed to represent widely used resources like HTML pages and are accessed using a simple HTTP protocol. Any web URL activates a web API. Web APIs are often called REST (representational state transfer) or RESTful because the publisher of REST interfaces doesn't save any data internally between requests. As such, requests from many users can be intermingled as they would be on the internet.

3. **Program APIs** are based on remote procedure call (RPC) technology that makes a remote program component appear to be local to the rest of the software. Service oriented architecture (SOA) APIs, such as Microsoft's WS-series of APIs, are program APIs.

5.2 WAMP - AutoBahn for IoT

Web Application Messaging Protocol (WAMP) is a sub-protocol of WebSocket which provides publish-subscribe and remote procedure call (RPC) messaging patterns.



Transport: Transport is channel that connects two peers.

- Session: Session is a conversation between two peers that runs over a transport.

- Client: Clients are peers that can have one or more roles. In publish-subscribe model client can have following roles:

- Publisher: Publisher publishes events (including payload) to the topic maintained by the Broker.

- Subscriber: Subscriber subscribes to the topics and receives the events including the payload. In RPC model client can have following roles:

- Caller: Caller issues calls to the remote procedures along with call arguments.
- Callee: Callee executes the procedures to which the calls are issued by the caller and returns the results back to the caller.
- Router: Routers are peers that perform generic call and event routing. In publish-subscribe model Router has the role of a Broker:
 - Broker: Broker acts as a router and routes messages published to a topic to all subscribers subscribed to the topic.

In RPC model Router has the role of a Broker:



– Dealer: Dealer acts a router and routes RPC calls from the Caller to the Callee and routes results from Callee to Caller.

• Application Code: Application code runs on the Clients (Publisher, Subscriber, Callee or Caller).

In RPC model client can have following roles: –

1. Caller: Caller issues calls to the remote procedures along with call arguments. –

Callee: Callee executes the procedures to which the calls are issued by the caller

and returns the results back to the caller. • Router: Routers are peers that perform generic call and event routing. In publish-subscribe model Router has the role of

a Broker: – Broker: Broker acts as a router and routes messages published to a topic to all subscribers subscribed to the topic.

In RPC model Router has the role of a Broker: –

1. Dealer: Dealer acts a router and routes RPC calls from the Caller to the Callee and routes results from Callee to Caller.

2. Application Code: Application code runs on the Clients (Publisher, Subscriber, Callee or Caller).

5.3 Xively Cloud for IoT

Use of Cloud IoT cloud-based service • The service provides for the data collection, data points, messages and calculation objects. • The service also provisions for the generation and communication of alerts, triggers and feeds to the user. • A user is an application or service. The user obtains responses or feeds from the cloud service.

Pachube platform: for data capture in real-time over the Internet • Cosm: a changed domain name, where using a concept of console, one can monitor the feeds • Xively is the latest domain name.

A commercial PaaS for the IoT/M2M • A data aggregator and data mining website often integrated into the Web of Things • An IoT PaaS for services and business services.

Xively PaaS services:

- Data visualisation for data of connected sensors to IoT devices.
- Graphical plots of collected data.
- Generates alerts.
- Access to historical data
- Generates feeds which can be real-world objects of own or others

Xively HTTP based APIs

- Easy to implement on device hardware acting as clients to Xively web services
- APIs connect to the web service and send data.



-
- APIs provides services for logging, sharing and displaying sensor data of all

Xively Support

- The platform supports the REST, WebSockets and MQTT protocols and connects the devices to Xively Cloud Services
- Native SDKs for Android, Arduino, ARM mbed, Java, PHP, Ruby, and Python languages
- Developers can use the workflow of prototyping, deployment and management through the tools provided at Xively

Xively APIs

- Enable interface with Python, HTML5, HTML5 server, tornado
- Interface with WebSocket Server and WebSockets
- Interface with an RPC (Remote Procedure Call).

Xively PaaS services

- Enables services
- Business services platform which connects the products, including collaboration products
- Rescue, Boldchat, join.me, and operations to Internet
- Data collection in real-time over Internet

Xively Methods for IoT Devices Data

- Concept of users, feeds, data streams, data points and triggers
- Data feed typically a single location (e.g. a device or devices network),
- Data streams are of individual sensors associated with that location (for example, ambient lights, temperatures, power consumption).
- Pull or Push (Automatic or Manual

Feed) Xively Data formats and

Structures

- Number of data formats and structures enable the interaction, data collection and services
 - Support exists for JSON , XML and CSV
-

- Structures: Tabular, spreadsheet, Excel, Data numbers and Text with a comma-separated values in file

Xively Uses in IoT/M2M

- Private and Public Data Access
- Data streams, Data points and Triggers
- Creating and Managing Feeds
- Visualising Data



5.4 Django

Django is an open source web application framework for developing web applications in Python.

- A web application framework in general is a collection of solutions, packages and best practices that allows development of web applications and dynamic websites.
- Django is based on the Model-Template-View architecture and provides a separation of the data model from the business rules and the user interface.
- Django provides a unified API to a database backend.
- Thus web applications built with Django can work with different databases without requiring any code changes.
- With this flexibility in web application design combined with the powerful capabilities of the Python language and the Python ecosystem, Django is best suited for cloud applications.
- Django consists of an object-relational mapper, a web templating system and a regular-expression based URL dispatcher. Django is Model-Template-View (MTV) framework.

Model

- The model acts as a definition of some stored data and handles the interactions with the database. In a web application, the data can be stored in a relational database, non-relational database, an XML file, etc. A Django model is a Python class that outlines the variables and methods for a particular type of data.

Template

- In a typical Django web application, the template is simply an HTML page with a few extra placeholders. Django's template language can be used to create various forms of text files (XML, email, CSS, Javascript, CSV, etc.)

View

- The view ties the model to the template. The view is where you write the
-

code that actually generates the web pages.

View determines what data is to be displayed, retrieves the data from the database and passes the data to the template.

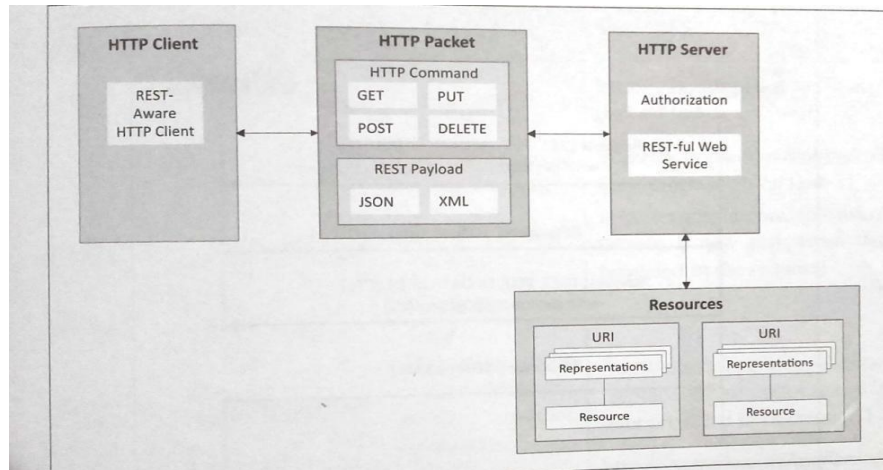
5.5 **Designing a RESTful Web API**

- i) REST based communication APIs(Request-Response Based Model)
- ii) WebSocket based Communication APIs(Exclusive Pair Based Model)

i) REST based communication APIs: Representational State Transfer(REST) is a set of architectural principles by which we can design web services and web APIs that focus on a system's resources and have resource states are addressed and transferred.



The REST architectural constraints are as follows: The below figure shows the communication between client server with REST APIs



Client-Server: The principle behind client-server constraint is the separation of concerns. Separation allows client and server to be independently developed and updated.

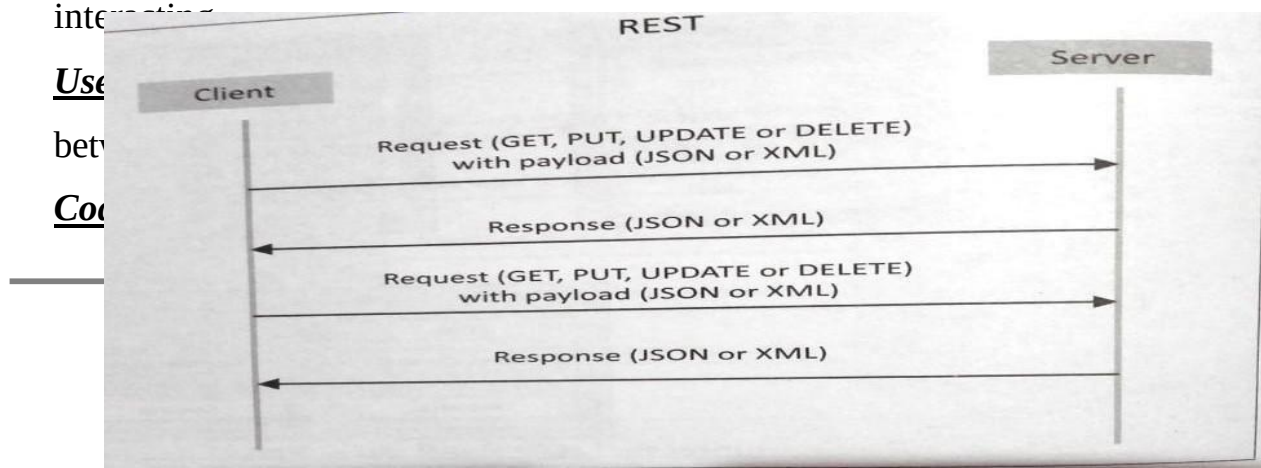
Stateless: Each request from client to server must contain all the info. Necessary to understand the request, and cannot take advantage of any stored context on the server. **Cache-able:** Cache constraint requires that the data within a response to a request be implicitly or explicitly labeled as cache-able or non-cacheable. If a response is cache-able, then a client cache is given the right to reuse that response data for later, equivalent requests.

Layered System: constraints the behavior of components such that each component cannot see beyond the immediate layer with which they are interacting.

Use

bet

Co



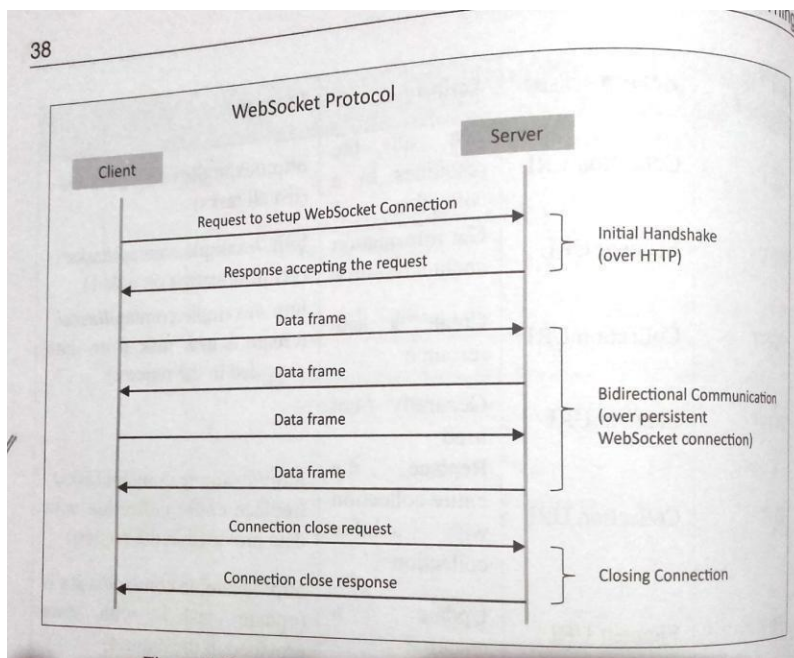
clients to execute in their context. This constraint is the only one that is optional.

The Request-Response model used by REST:

RESTful web service is a collection of resources which are represented by URIs. RESTful web API has a base URI(e.g: <http://example.com/api/tasks/>). The clients and requests to these URIs using the methods defined by the HTTP protocol(e.g: GET, PUT, POST or DELETE). A RESTful web service can support various internet media types.

ii) WebSocket Based Communication APIs

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



5.6 Amazon Web Services for IoT

i) Amazon EC2

In this example, a connection to EC2 service is first established by calling `boto.ec2.connect_to_region`.

- The EC2 region, AWS access key and AWS secret key are passed to this function. After connecting to EC2, a new instance is launched using the `conn.run_instances`

function.

- The AMI-ID, instance type, EC2 key handle and security group are passed to this function.

ii) **Amazon**

AutoScaling

AutoScaling

Service

- A connection to AutoScaling service is first established by calling `boto.ec2.autoscale.connect_to_region` function.

• **Launch Configuration**

- After connecting to AutoScaling service, a new launch configuration is created by calling `conn.create_launch_configuration`. Launch configuration contains instructions on how to launch new instances including the AMI-ID, instance type, security groups, etc.

AutoScaling Group

- After creating a launch configuration, it is then associated with a new AutoScaling group.

AutoScaling group is created by calling `conn.create_auto_scaling_group`. The settings for AutoScaling group such as the maximum and minimum number of instances in the group, the launch configuration, availability zones, optional load balancer to use with the group, etc. **AutoScaling Policies**

- After creating an AutoScaling group, the policies for scaling up and scaling down are defined.
- In this example, a scale up policy with adjustment type Change In Capacity and `scaling_adjustment = 1` is defined.
- Similarly a scale down policy with adjustment type ChangeInCapacity and `scaling_adjustment = -1` is defined.

CloudWatch Alarms

- With the scaling policies defined, the next step is to create Amazon CloudWatch alarms that trigger these policies.
- The scale up alarm is defined using the CPUUtilization metric with the Average statistic and threshold greater 70% for a period of 60 sec. The scale up policy created previously is associated with this alarm. This alarm is triggered when the average CPU utilization of the instances in the group becomes greater than 70% for more than 60 seconds.
- The scale down alarm is defined in a similar manner with a threshold less than 50%.

iii) **Amazon S3:**

- In this example, a connection to S3 service is first established by calling

boto.connect_s3 function.

- The upload_to_s3_bucket_path function uploads the file to the S3 bucket specified at the specified path.

iv) **Amazon RDS**

In this example, a connection to RDS service is first established by calling boto.rds.connect_to_region function.

- The RDS region, AWS access key and AWS secret key are passed to this function.
- After connecting to RDS service, the conn.create_dbinstance function is called to launch a new RDS instance.

-
- The input parameters to this function include the instance ID, database size, instance type, database username, database password, database port, database engine (e.g. MySQL5.1), database name, security groups, etc.

v) **Amazon Dynamo DB**

In this example, a connection to DynamoDB service is first established by calling `boto.dynamodb.connect_to_region`.

- After connecting to DynamoDB service, a schema for the new table is created by calling `conn.create_schema`.
- The schema includes the hash key and range key names and types.
- A DynamoDB table is then created by calling `conn.create_table` function with the table schema, read units and write units as input parameters.

5.7 SkyNetIoT Messaging Platform.

SkyNet is running on a dozen Amazon EC2 servers and has nearly 50,000 registered smart devices including: Arduinos, Sparks, Raspberry Pis, Intel Galileos, and BeagleBoards, Matthieu said. SkyNet runs as an IoT platform-as-a-service (PaaS) as well as a private cloud through Docker, the new lightweight container technology. The platform is written in Node.js and released under an MIT open source license on GitHub.

The single SkyNet API supports the following IoT protocols: HTTP, REST, WebSockets, MQTT (Message Queue Telemetry Transport), and CoAP (Constrained Application Protocol) for guaranteed message delivery and low-bandwidth satellite communications, Matthieu said. Every connected device is assigned a 36 character UUID and secret token that act as the device's strong credentials. Security permissions can be assigned to allow device discoverability, configuration, and messaging.

