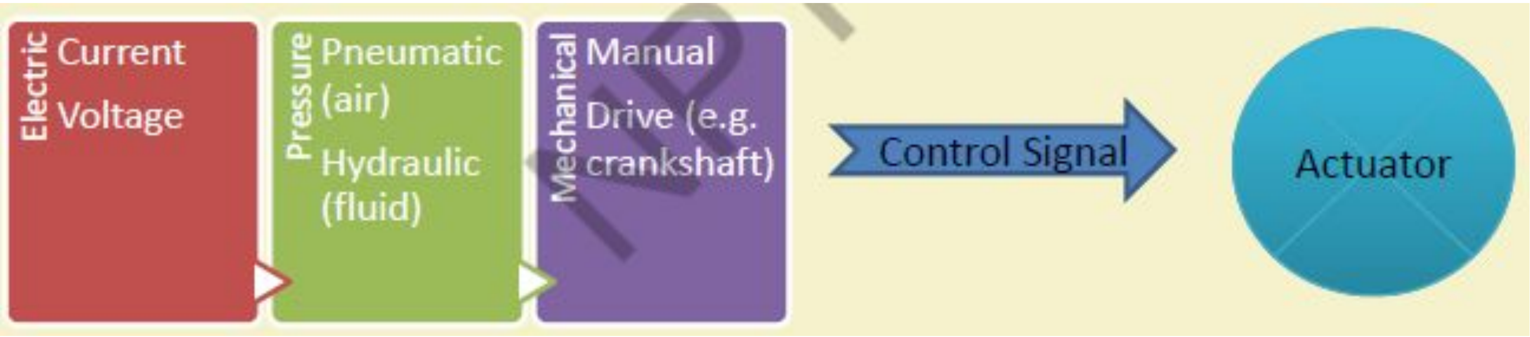
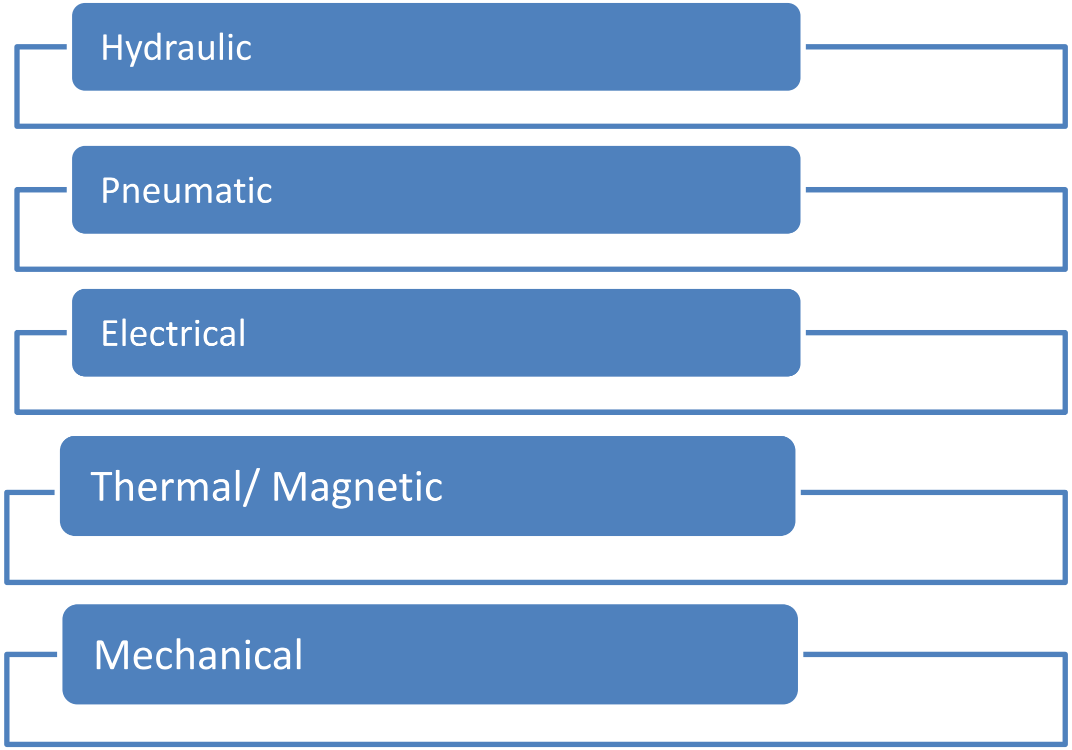
Actuator

Actuator

- An actuator is a component of a machine or system that moves or controls the mechanism or the system.
- An actuator is the mechanism by which a control system acts upon an environment
- An actuator requires a control signal and a source of energy.
- Upon receiving a control signal is received, the actuator responds by converting the energy into mechanical motion.
- The control system can be simple (a fixed mechanical or electronic system), software-based (e.g. a printer driver, robot control system), a human, or any other input.



Actuator types:



Hydraulic

Pneumatic

Electrical

Thermal/ Magnetic

Mechanical

Hydraulic Actuators

- A hydraulic actuator consists of a cylinder or fluid motor that uses hydraulic power to facilitate mechanical operation.
- The mechanical motion is converted to linear, rotary or oscillatory motion.
- Since liquids are nearly impossible to compress, a hydraulic actuator exerts considerable force.
- The actuator’s limited acceleration restricts its usage.

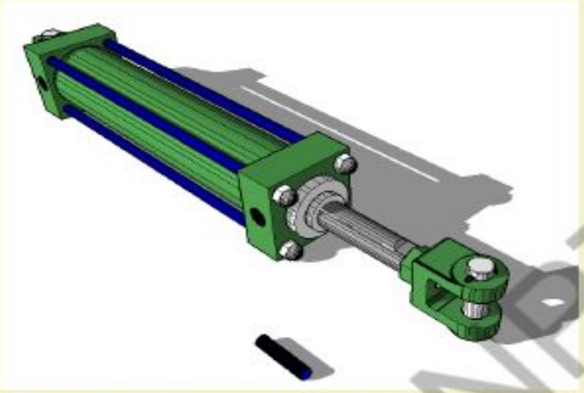


Fig: An oil based hydraulic actuator

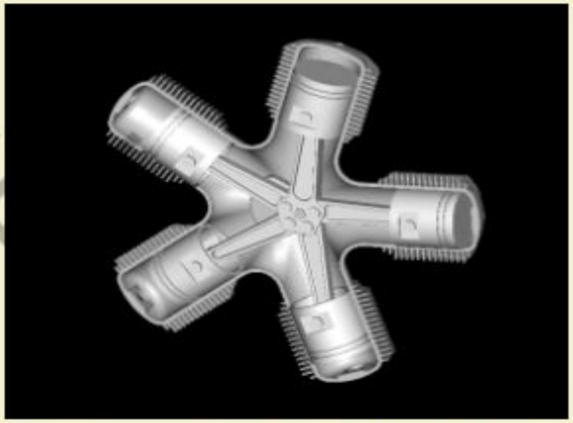


Fig: A radial engine acts as a hydraulic actuator
Source: Wikimedia Commons

Pneumatic Actuators

- A pneumatic actuator converts energy formed by vacuum or compressed air at high pressure into either linear or rotary motion.
- Pneumatic rack and pinion actuators are used for valve controls of water pipes.
- Pneumatic energy quickly responds to starting and stopping signals.
- The power source does not need to be stored in reserve for operation.
- Pneumatic actuators enable large forces to be produced from relatively small pressure changes (e.g., Pneumatic brakes can are very responsive to small changes in pressure applied by the driver).
- It is responsible for converting pressure into force.

Pneumatic Actuators

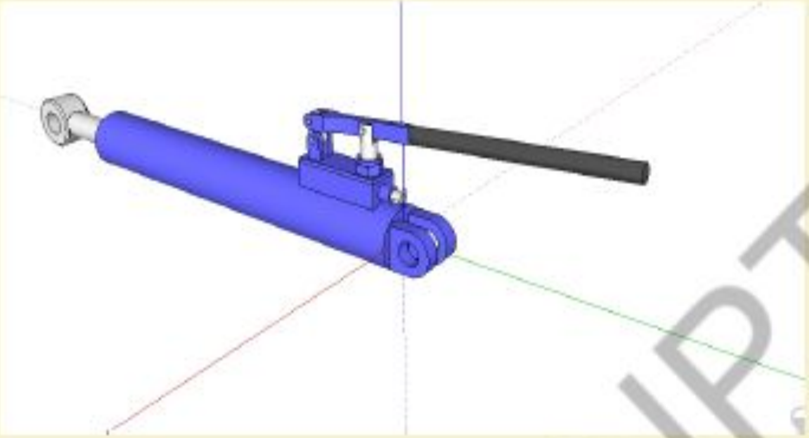


Fig: A manual linear pneumatic actuator

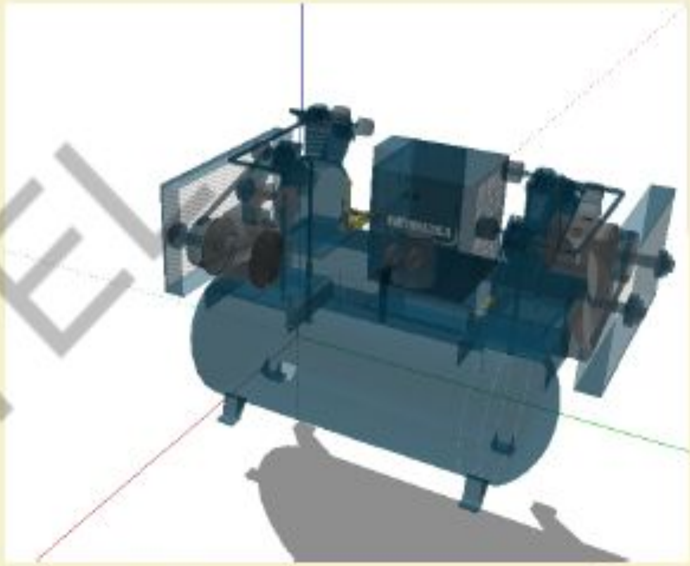


Fig: An air pump acts as a pneumatic actuator

Electric Actuators

- An electric actuator is generally powered by a motor that converts electrical energy into mechanical torque.
- The electrical energy is used to actuate equipment such as solenoid valves which control the flow of water in pipes in response to electrical signals.
- Considered as one of the cheapest, cleanest and speedy actuator types available.

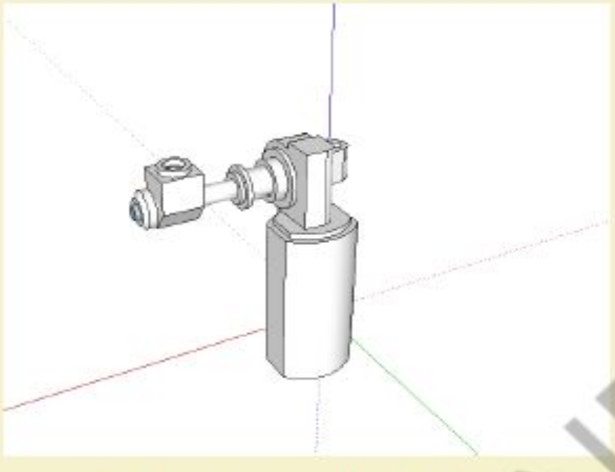


Fig: A motor drive-based rotary actuator

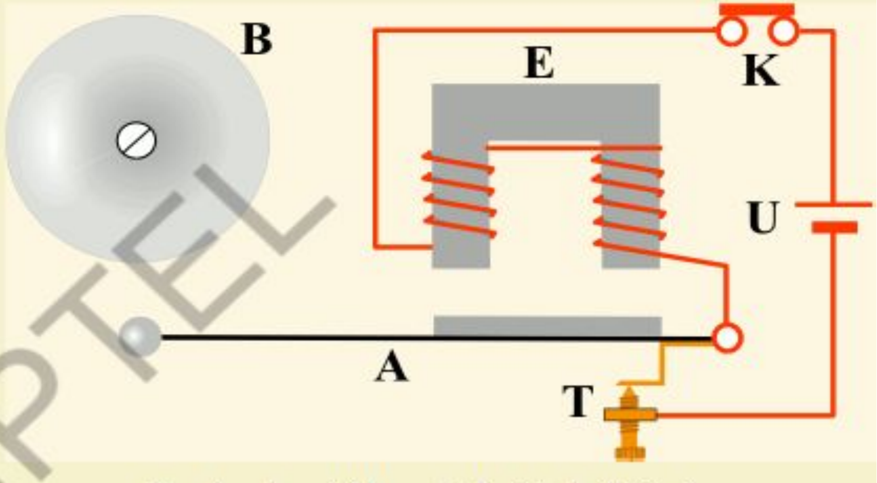


Fig: A solenoid based electric bell ringing mechanism

Source: Wikimedia Commons

Thermal or Magnetic Actuators

- These can be actuated by applying thermal or magnetic energy.
- They tend to be compact, lightweight, economical and with high power density.
- These actuators use shape memory materials (SMMs), such as shape memory alloys (SMAs) or magnetic shape-memory alloys (MSMAs).
- Some popular manufacturers of these devices are *Finnish Modti Inc.* and *American Dynalloy*.

Thermal or Magnetic Actuators

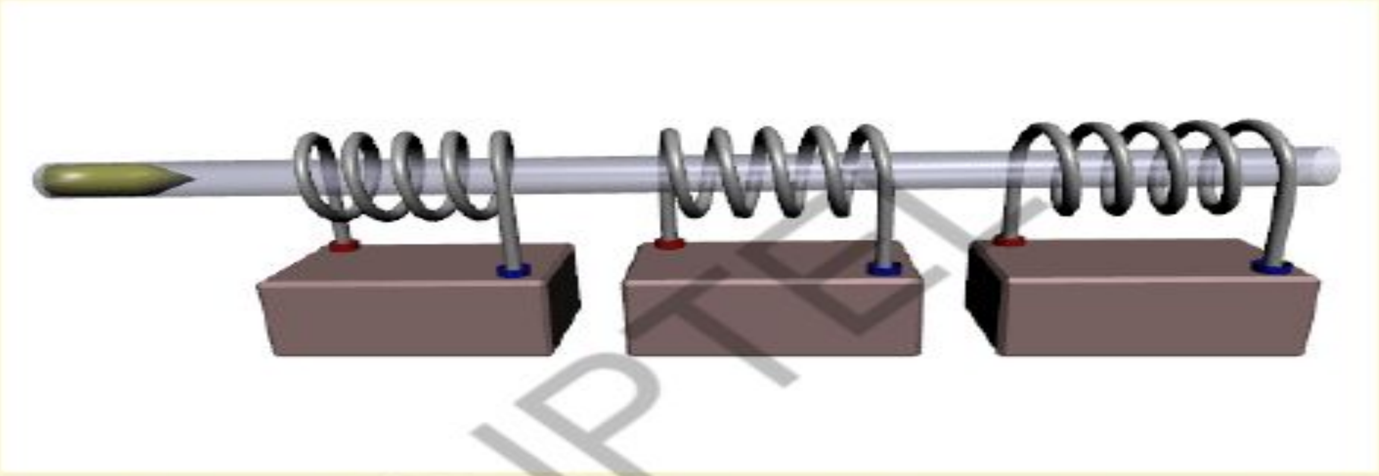


Fig: A coil gun works on the principle of magnetic actuation

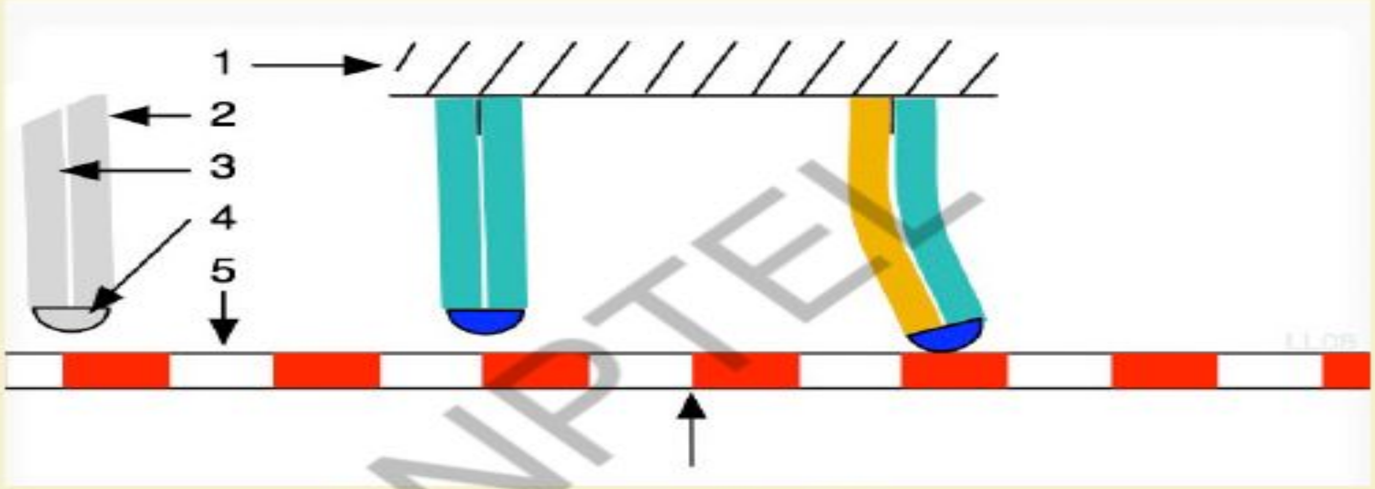


Fig: A piezo motor using SMA

Mechanical Actuators

- A mechanical actuator converts rotary motion into linear motion to execute some movement.
- It involves gears, rails, pulleys, chains and other devices to operate.

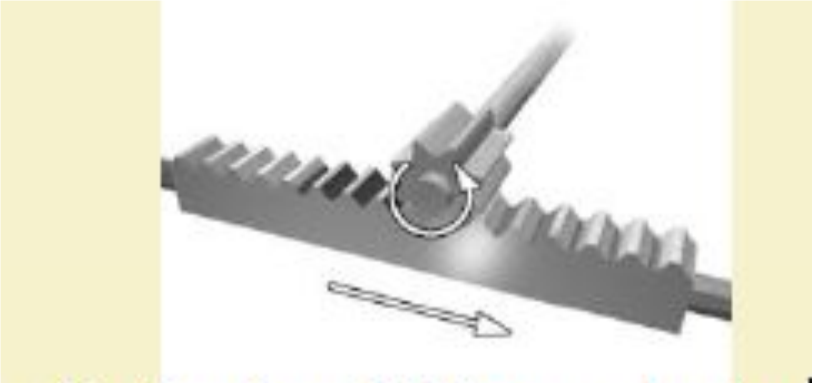


Fig: A rack and pinion mechanism

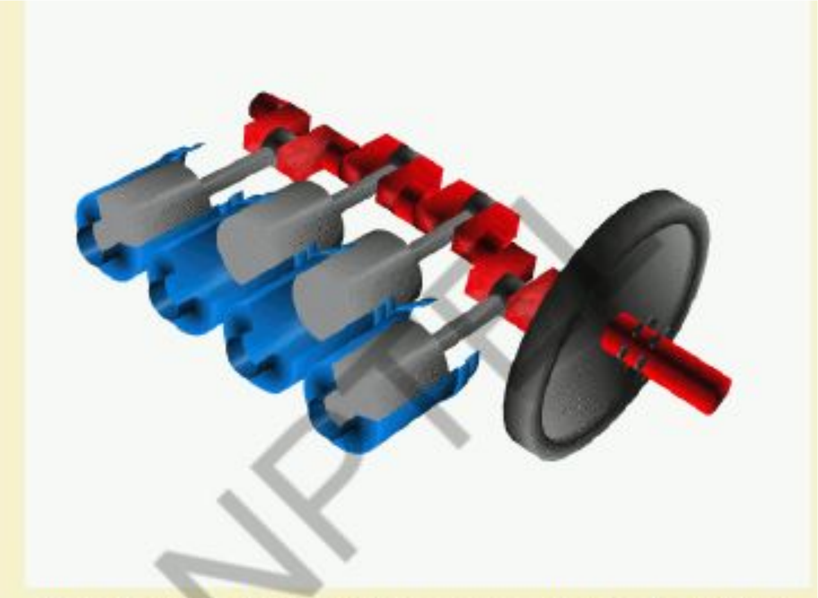
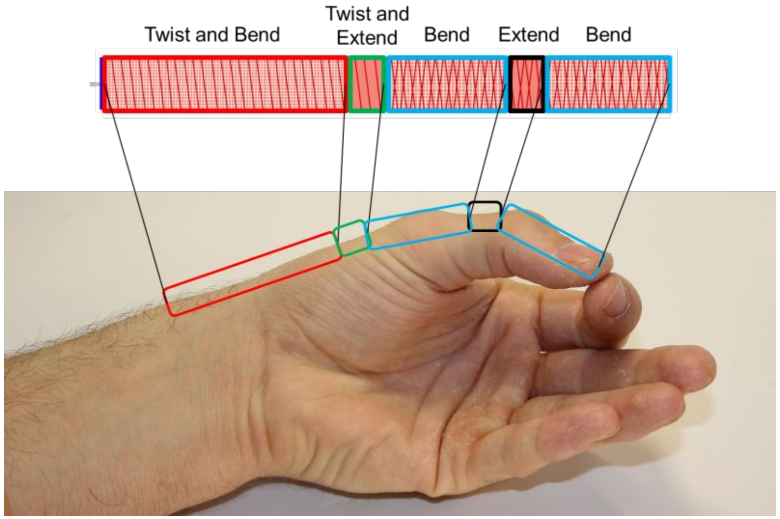


Fig: A crank shaft acting as a mechanical actuator

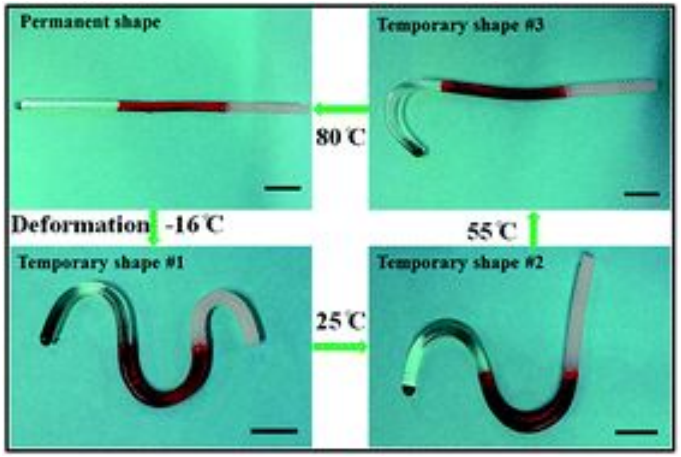
Soft Actuators

- Soft actuators (e.g. polymer based) are designed to handle fragile objects like fruit harvesting in agriculture or manipulating the internal organs in biomedicine.
- They typically address challenging tasks in robotics.
- Soft actuators produce flexible motion due to the integration of microscopic changes at the molecular level into a macroscopic deformation of the actuator materials.



Shape Memory Polymers

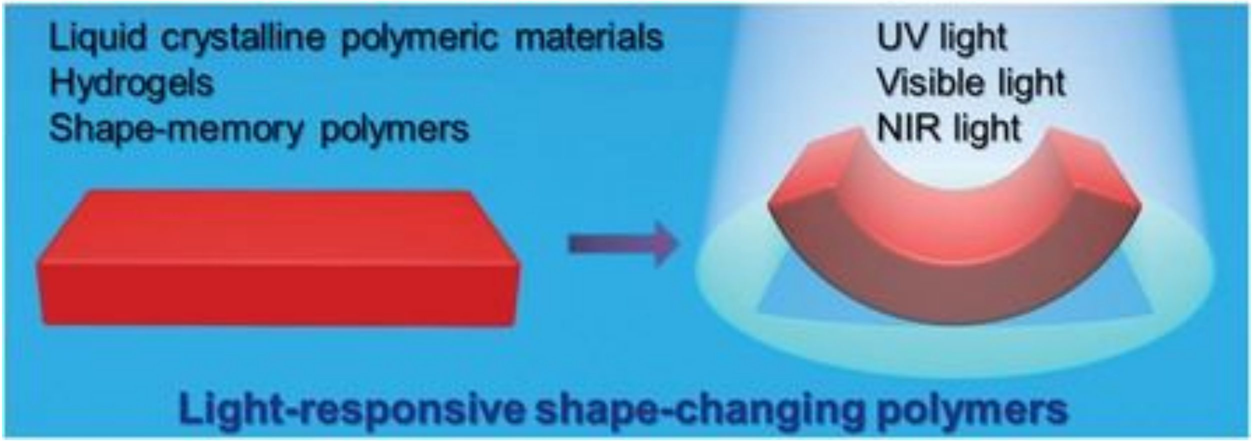
- Shape memory polymer (SMP) actuators function similar to our muscles, even providing a response to a range of stimuli such as light, electrical, magnetic, heat, pH, and moisture changes.
- SMP exhibits surprising features such a low density, high strain recovery, biocompatibility, and biodegradability.



This polymers with well-separated glass transition temperatures can change their shapes in a pre-defined way over a broad temperature range with almost full recovery of all temporary shape

Light Activated Polymers

- Photopolymer/light activated polymers (LAP) are a special type of SMP that are activated by light stimuli.
- The LAP actuators have instant response.
- They can be controlled remotely without any physical contact, only using the variation of light frequency or intensity.


Liquid crystalline polymeric materials
Hydrogels
Shape-memory polymers
UV light
Visible light
NIR light
Light-responsive shape-changing polymers

Challenges for IOT

Security: As the infrastructure increases, the number of connected devices increases which also increase the opportunity to exploit the vulnerabilities that expose the user data to outer world.

To protect the user data, data-encryption technique can be used. Here we can use Secure Socket Layer(SSL) to encrypt and protect user data which is flown online.

Data Authentication is another issue in IOT where the chances of device itself being hacked may exist. We can not authenticate the source to which we want to communicate or from which we need data. Anyone can make up fake data and send it to a sensor which is instructed to send correct data.

Side-channel attack focuses less on the information and more on how that information is being presented. It can access the timing information, power consumption, electromagnetic leak etc.

Challenges for IOT

Hardware security issues are another challenges for IOT. Many companies like ARM, Intel are manufacturing the processor for different IOT. To address different issues, they are adding advance features and complex technologies in the chip which increase the cost of it. Also it will consume more energy which is definitely a challenges for IOT.

Privacy: Most of the IOT are fabricated with integrated devices which sense the environments without human interaction. This become more prevalent when it comes to consumer devices such as tracking devices for car smart TV etc. Voice recognition and vision features are now integrated in smart TV. These feature can listen continuously to conversation or look for activity and transmit data selectively to cloud services for processing. These cloud services may sometimes include third parties which is a matter of concern.

Challenges for IOT

Scalability: One should consider the present and future needs when the IOT are deploying. The system should be scalable i.e. it will be able to accommodate future expansion when shift in technologies occur without affection present scenario. This is because , the system or network can adapt when failures occur and remain mostly operational until the issue is repaired.

Bandwidth Management: Since the IOT has limited capacity, the bandwidth required for transmitting data is also limited. So huge amount of data should be aggregated or filtered out , so that the data to be transmitted should be fit within this limited bandwidth. Which may filter out some other important data. This problem is going foreword with IOT and every solution is going to be different and required different approaches to solve the network latency and bandwidth problem.

Challenges for IOT

Interoperability: Since so many industries manufacture different IOTs for same operation, interoperability issues may arises between each connection.

Different devices that are not made by same manufacturer cannot be integrated.

Different devices can not be run on same operating system.

There may be problem in connecting new version devices with older version devices.

Different types of connectors or connectivity networks, communication protocols may not work properly.

Data Storage: The IOT creates different types of data with huge volume needs to be stored. This data are of two types.

Challenges for IOT

One type is large-file data such as images and video captured from IOT devices typically accessed sequentially where as second types of short-files data like log-files captured from sensors are accessed randomly. So for storage such different pattern of data, the first task is to determine the types and volume of data the project will generate. The data center like cloud-storage may be one solution.

Data Analytic: The analysis of IOT generated data is crucial as the data generated in huge number with different pattern from different sources. So it is important to determine what amount of data and what pattern of data actually has value for a specific problem and rejecting unnecessary data which may confuse the user. The IOT analytic needs to identify complex pattern or trends to occur over time.

Challenges for IOT

Here we are often interested in data which deviates from normal pattern which may helpful to find abnormal pattern for example sudden increase in temperature etc. Classic time series and forecasting models basically used for identifying the pattern.

Standards: All the manufacturer of IOT devices must maintain the universal standards. With absence of standards, manufacturer may design products that operates in any number of disruptive way online without regard for their impact. And these devices may have negative consequences for networking resources they connect to.

Regulation: Legal issues concerning IOT devices are not limited to potential violation of civil right. Other issue like cross-boarder data flow, legal liabilities, privacy lapses, security breaches should be considered.

INTRODUCTION

Internet of Things:-(IOT): It is the network of
09:00 Physical devices, vehicles, home appliences, Which
enables these Objects to get connected and
10:00 exchange data.

11:00 Each physical device is identifiable in the
12:00 system and also able to inter-operate within
LUNCH the existing network without any human
02:00 intervention. IOT unifies different technology
03:00 like low power embedded System, cloud Computing,
04:00 big data, machine learning and networking.

05:00

IOT also use RFID(Radio frequency
Eve
Identification), NFC (Near Field Communication)
barcode, QRCode, digital watermarking etc.
for connecting and data exchange
Notes:
purposes.

Characterstiz of IOT:-

The fundamental Characterstics of IOT are.

1) Inter Connectivity :- Anything Can be interConnected With the global informaton and Communicaton Infrastructure.

2) Things related Services :- It provids Things related Services within the Constraint of things and their associated virtual things.

3) Heterogeneity :- In IOT, the devices are heterogeneous in nature. They differ in hardwareplatform, application and networks. So They Can interact With other devices or service platform through

February 2018

01 5th Week • 032-333

Thursday

January '18				
1	8	15	22	29
2	9	16	23	30
3	10	17	24	31
4	11	18	25	
5	12	19	26	
6	13	20	27	
7	14	21	28	

01

February '18		
	5	12
	6	13
	7	14
1	8	15
2	9	16
3	10	17
4	11	18

O

different Network.
09.00
4) Dynamic Changes:- The State of devices Can
10.00
Change dynamically. i.e the devices may be
11.00
on Sleeping ~~and~~ or Waking up mode, Connected
12.00
or disconnected mode, location and speed
LUNCH
of transmission.
02.00
5) Enormous Scale:- The devices should not Combin
03.00
within a specific area. The number of devices
04.00
that need to be managed and that Communic
05.00
with one-another will be atleast in order of
Eve.
magnitude larger than the devices Connected
to the Current internet. This cause ~~f~~
generation of huge amount of data and
Notes :
their interpretation which need to be
handled Carefully.

6) Safty:- IOT must be made or designed with safty in mind which includes safty to Personal data and safty of Physical devices.

7) Connectivity:- Connectivity enables network accessibility and compatibility. ~~Accessibility~~ Each device should be accessible in the network while Compatibility provides the Common ability to Consume and produce data. The device should be disconnected when it not in used and Connected if it ~~needs data~~ wants to receive or send data.

8) Naming and Addressing:- In IOT environment each device should be Addressable uniquely. The conversion and translation of addresses from one network to other should be done efficiently

January '18					01
1	8	15	22	29	M
2	9	16	23	30	T
3	10	17	24	31	W
4	11	18	25		T
5	12	19	26		F
6	13	20	27		S
7	14	21	28		S

February '18					02
	5	12	19	26	M
	6	13	20	27	T
	7	14	21	28	W
1	8	15	22		T
2	9	16	23		F
3	10	17	24		S
4	11	18	25		S

09.00 Aplication of IOT!-

10.00 1) Smart home:- In this Concept, all the home

11.00 appliences Can Communicate With each other.

12.00 Like The air Conditioning should switched on

LUNCH before reaching home or Switched off light

02.00 and fan When we left home. Also unlock

03.00 the door to a friend for temporry access

04.00 When we are not in home, etc.

05.00 ~~20~~ 2) Wearables:- These devices are installed with

Eve. Sensors and Software which Collect data and

information about the users. For Examples

fitness check, health and Entartainment

Notes: etc, These data later pre-processed to

extract essential insight about user

3) Connected Car or Smart Car:- A connected car is a vehicle which is able to-optimize it's own operation, maintenance as well as Comfort of passengers using on-board Sensors and Internet Connectivity is ensured. It also identify the behavior of another Car Coming in opposite direction.

4) Smart Industries:- It is also known as Industrial Internet of Things. (IIOT). This Smart industry is equipped with Sensor, software and bigdata analytic to Create brilliant and intelligent machine. These machines are more accurate and Consistent.

Notes :

09.00 5) Smart Cities:- It includes Smart Surveillance,
10.00 automated transportation, Smarter energy mgmt
11.00 Systems, water distribution, urban Security,
12.00 enviornment monitoring, ~~etc~~. pollution control,
LUNCH traffic Congestion control, Energy distribution
02.00 System etc.

03.00 6) Smart Agriculture:- smart farming is one of the
04.00 fastest growing field in IOT. Here the IOT is
05.00 used for sensing soil moisture, soil nutrients,
Eve. controlling water usage, determining custom
fertilizers etc. From this IOT data the farmers
are getting meaningful insight to yield
Notes : better return on investment.

7) Smart Retail: IOT provids an opportunity to the retailers to get Connected with the Customers to enhance the in-store experience. Interacting with the Smart phone and beacon technology can help retailler to serve their Customer in a better manner.

8) Energy management: The Smart grid concept is to Collect data in an automated fashion and analyse the behavier of Electricity Consumers and Suppliers for improving efficiency as well as economics of electricity uses.

9) Smart Healthcare:- IOT in healtcare is aimed at empowering people to live healthier life by wearing Connected devices. The collected data are analysied and provide ~~so~~ Strategy to Combat illness.

February 2018

07 6th Week • 038-327

Wednesday

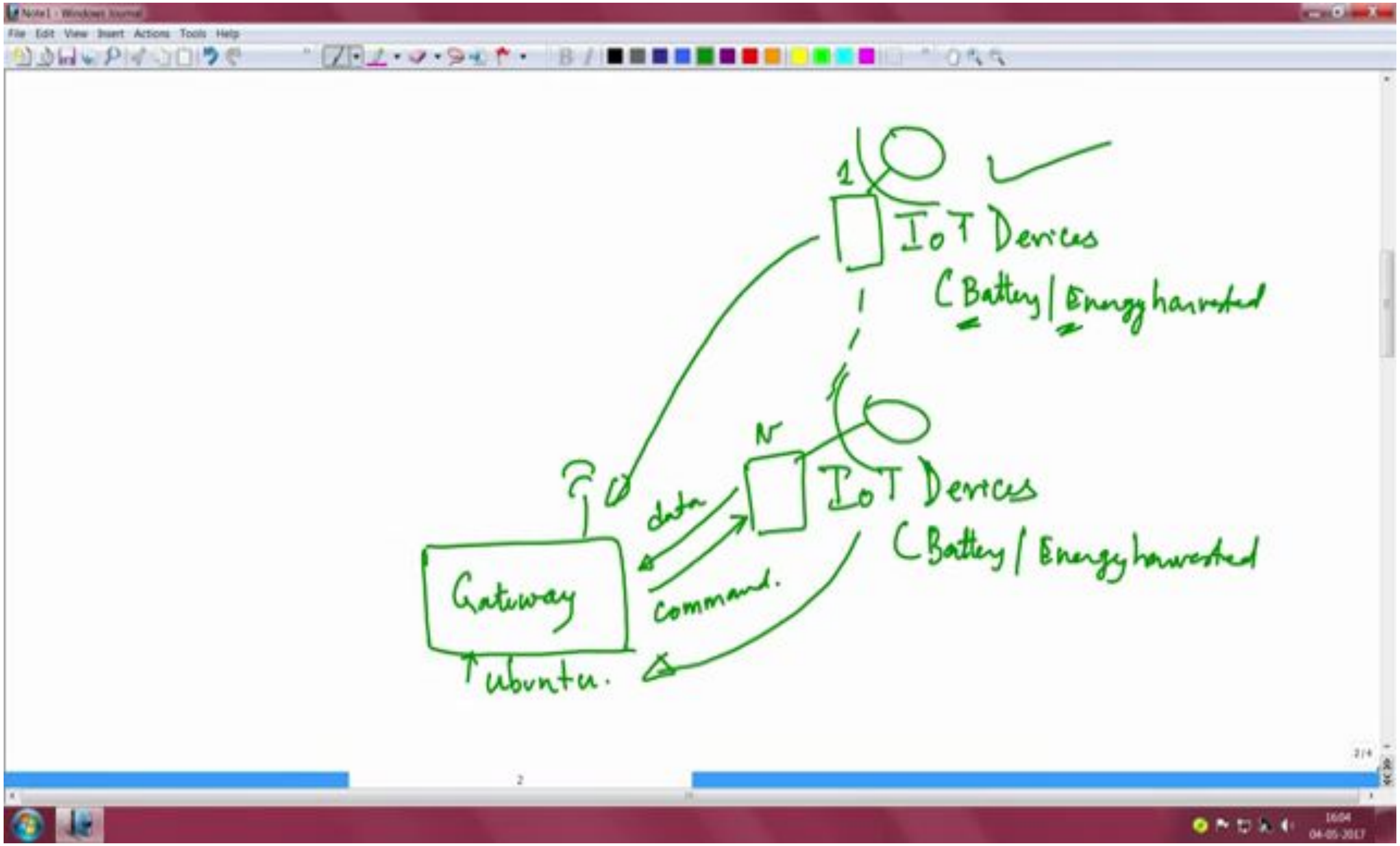
January '18				
1	8	15	22	29
2	9	16	23	30
3	10	17	24	31
4	11	18	25	
5	12	19	26	
6	13	20	27	
7	14	21	28	

February '18				
	5	12	19	26
	6	13	20	27
	7	14	21	28
1	8	15	22	
2	9	16	23	
3	10	17	24	
4	11	18	25	

09.00 10) Smart Dust:- This Consist of Sensors at the
10.00 nanotechnology level that can be deployed in
11.00 large number. With a variety of applicatons.
12.00 They are ~~com~~ very Small Computers like grain size
LUNCH Can be sprayed or injected almost anywhere
02.00 to measere the Chemical in the soil or
03.00 to diagnose problems in human body. They
04.00 are a single package with Sensing, Computing,
05.00 Communicating and power to Collect data
Eve. and report is back to home base.

Notes :

IOT Architecture

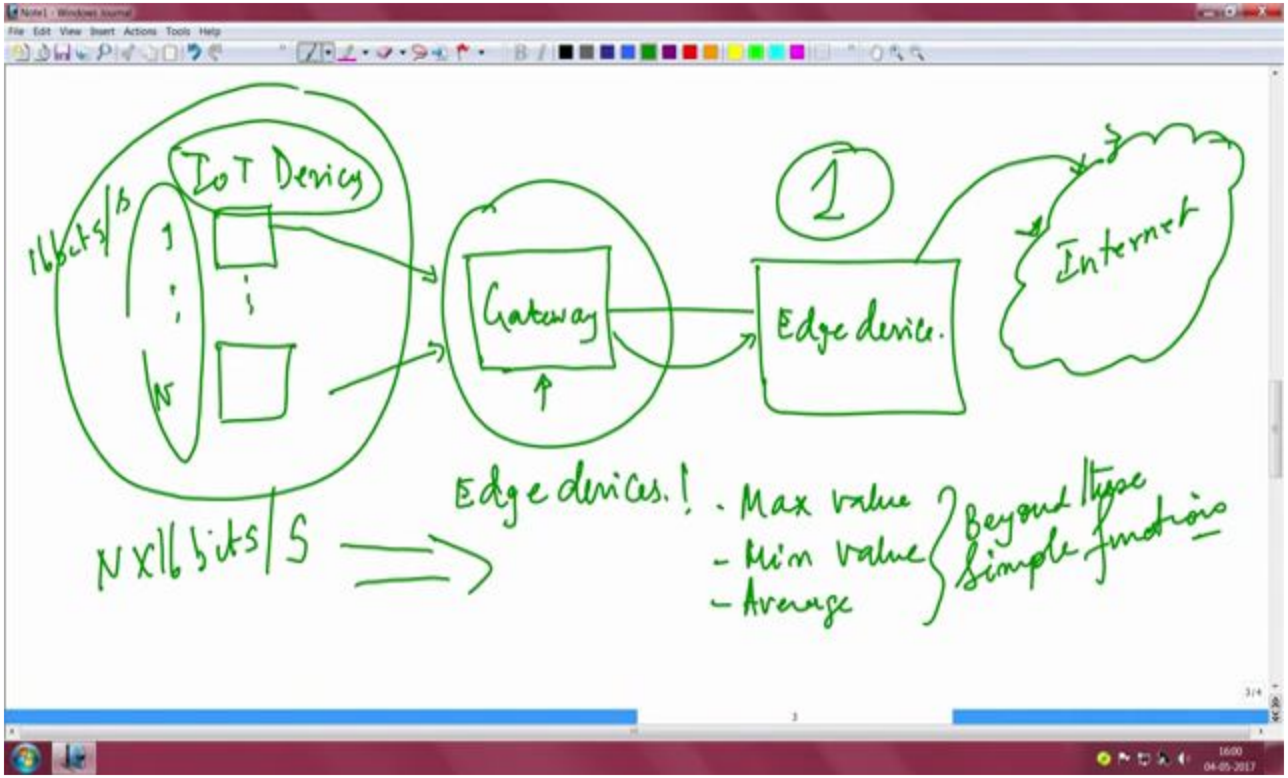

Note1 - Windows Journal
File Edit View Insert Actions Tools Help
1
IoT Devices
(Battery / Energy harvested
N
IoT Devices
(Battery / Energy harvested
data
command.
Gateway
ubuntu.
2
2/4
16:04
04-05-2017

IOT Architecture

- This picture essentially has IoT devices.
- 1 to n such devices may be present
- These IoT devices either battery driven or energy harvested .
- All these IoT devices basically are essentially a basically equipped with sensors which sense something from the environments .
- Basically all these sensing environments are analog by nature, but nowadays analog information is directly processed .
- Essentially, one could be processing the analog information analog data into digital data and making it available directly onto these devices.
- All of them are pushing data to what is known as this gateway device.

- An IoT device may not even exceed the size of an ATM card
- This is a small PCB which has some circuitry , chip, connectors and has some sort of a storage.
- But the gateway essentially is bigger in size which includes the raspberry pi ,beagleboard , odroid ,Edison, Galileo ,arduino based boards etc.
- Arduino based board gatewaye, is an important hardware which is available and very popular in its usage.
- These gateway run an embedded OS typically something like open source ubuntu which is a LINUX distribution.

IOT Architecture



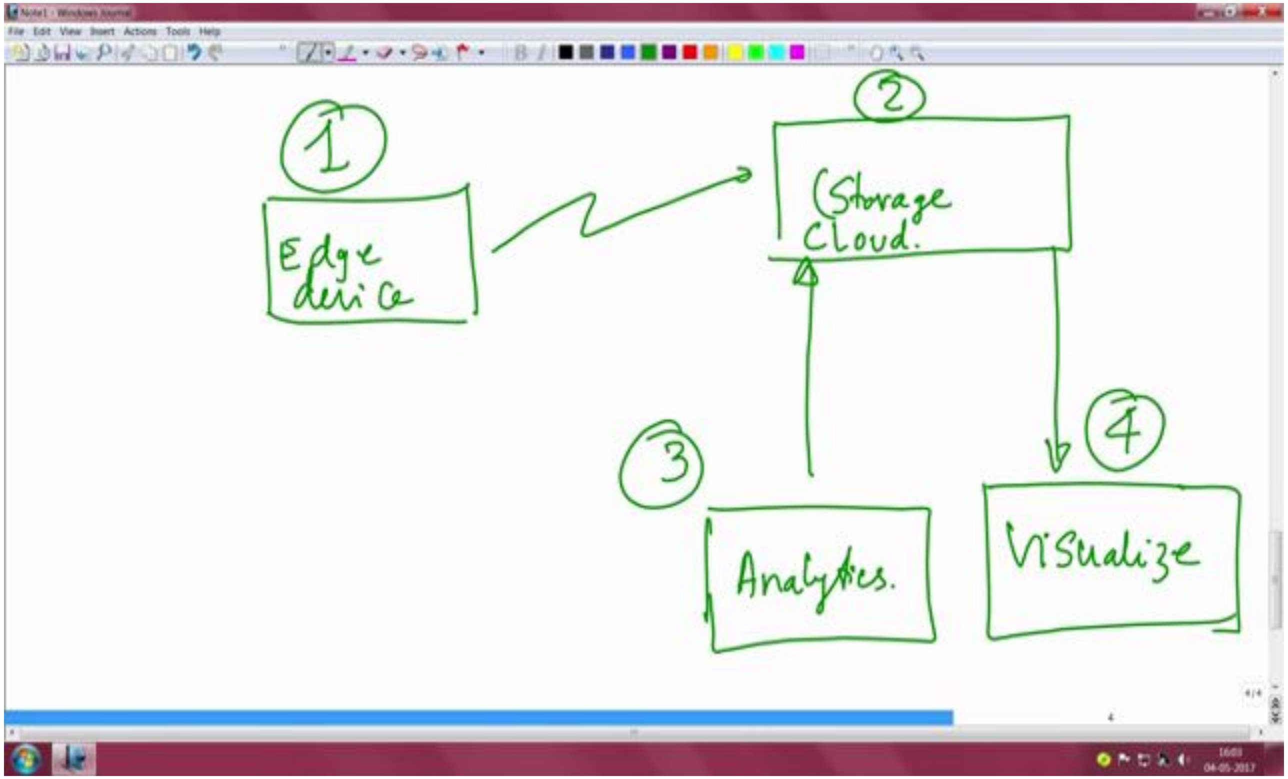
IOT Architecture

- These IOT devices sense any event occurs in the environment and send it to gateway.
- Since the sensors are low power devices they might not able to send the data directly to the any storage like cloud through internet because it needs to convert these data to TCP/IP protocol which consume more energy.

IOT Architecture

- Hence the sensed data will be sent to Gateway which is capable of converting these data to TCP/IP and send it to cloud through internet.
- But the data should be refined before send it to the cloud. This job can be done through the EDGE devices.
- Hence the edge device analyze the data received from gateway and send it to the cloud through any network.

IOT Architecture



IOT categories

IOT can be divided into two categories

Industrial IOT:

- The IOT used in the industries are called as Industrial IOT. In IIOT a device connects to both an IP network and the global network.
- Connectivity between the nodes is done using regular as well as industry specific technologies.
- The IIOT generates massive amount of data which need different computing requirements like Big Data, Cloud Computing, machine Learning etc.

Consumer IOT: The IOT used for consumer benefit purposes are called Consumer IOT. These IOT devices communicate within the locally networked devices via ZigBee, Bluetooth, Wifi etc . All the local communication and the communication to outside internet is done through gateway. It is about human interaction with the physical or virtual objects.

IOT categories

Consumer IOT:

- The IOT used for consumer benefit purposes are called Consumer IOT.
- These IOT devices communicate within the locally networked devices via ZigBee, Bluetooth, Wifi etc .
- All the local communication and the communication to outside internet is done through gateway.
- It is about human interaction with the physical or virtual objects.

IOT Enablers and Connectivity Layers

IOT enablers can be based on three factors

- Implementation Perspective: It includes smart industries, smart homes, smart factories, smart vehicles, smart healthcare etc.
- Connectivity method: It includes different network technology like ZigBee, RFID, Bluetooth, WiFi, 6LoWAPN, LORA etc.
- Enabling Technology: it include cloud computing, Artificial Intelligence, Big Data, Machine Learning, Deep Learning, Fog Computing etc.

<table>
<tr><td><b>Implementation</b>
(Smart Industries, Smart Home, Smart healthcare)</td><td rowspan="2"><b>Enabling Technology</b>
(Machine Learning, Big Data, AI, Deep Learning)</td></tr>
<tr><td><b>Connectivity</b>
(ZigBee, RFID, LORA, Bluetooth, WiFi)</td></tr>
</table>

Baseline Technologies

Three baseline technologies which are closely related to IOT.

Machine-to-Machine communication:

- This refers to communication and interaction between machines and devices.
- M2M is used to describe any technology that enables the networked devices to exchange information and perform actions without the manual assistance of humans.
- M2M communication is an important aspect of warehouse management, remote control, robotics, traffic control, logistic services, supply chain management, fleet management and tele-medicine.
- The key components of a M2M system includes sensors, RFID, Wi-Fi or cellular communication link and automatic computing software programmed to help a networked device interpret data and make decision.

Cyber-Physical System:

- These are the intelligent ICT systems that are interconnected, interdependent, collaborative, autonomous and provide computing and communication, monitoring and control of physical components in various applications.
- Some latest features like adaptability, reactivity, optimality and security are also embedded in the CPS system.
- CPS are systems that integrate computing elements with the physical components and processes.
- The computing elements coordinate and communicate with sensors, which monitor cyber and physical indicators and actuators.
- CPS use sensors to connect all distributed intelligence in the environment to gain the deeper knowledge of the environment which enables more accurate actions and tasks.
- Sensor –based and communication-based systems are applications of CPS.

Web Of Things:

- WOT is a software architectural styles and programming patterns that allow real-world objects to be a part of World Wide Web. It simplifies the creation of IOTs.
- WOT reuses the existing web standards like REST, HTTP, JSON, JSON-LD, Microdata, WebSocket etc.
- It enables access and control over IOT resources and applications using HTML5.0, JavaScript, Ajax, PHP etc.
- WOT is based on Representational State Transfer (REST) principle which enables both developers and deployers to benefit from the popularity and maturity of web technology.

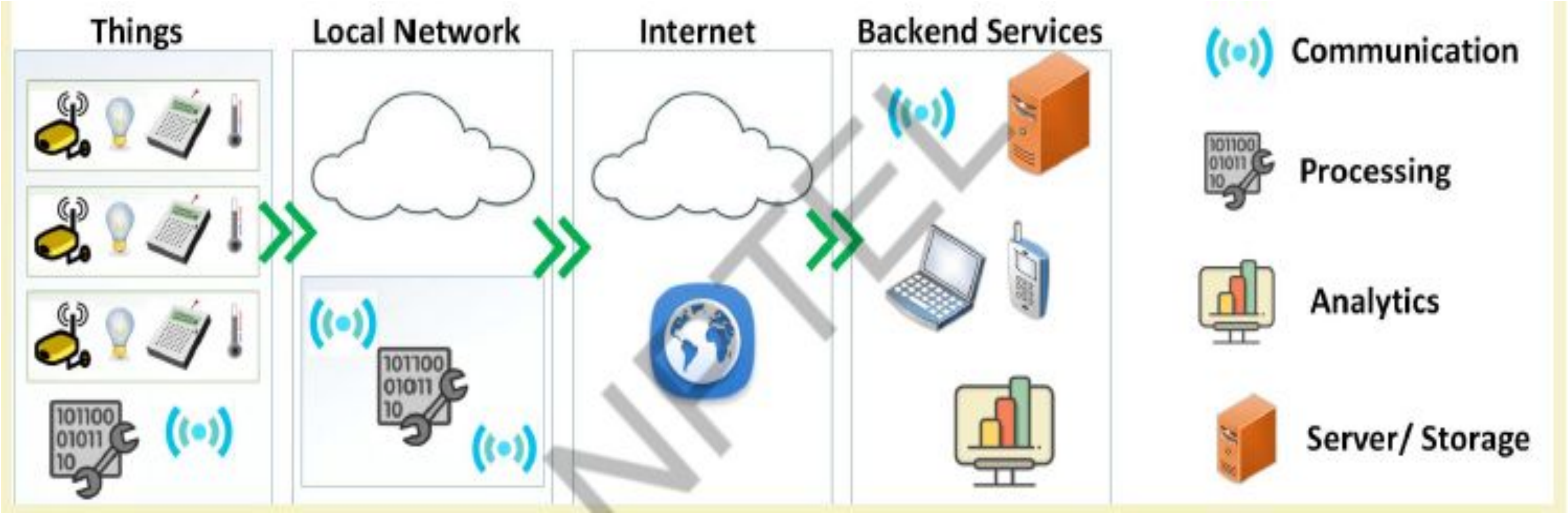
IOT Components and Implementation

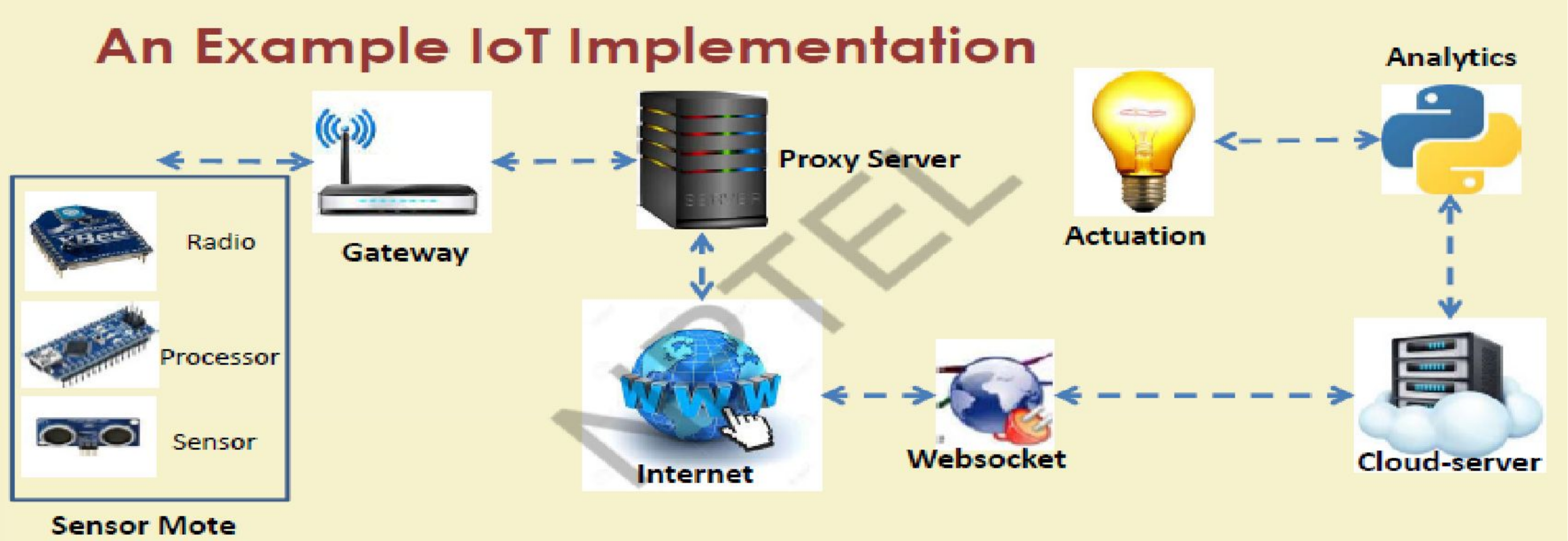
The functional components of IOT includes the following entities..

- Device (Things): various devices connected to IOT forms the components for interaction and communication with other IOT devices. Examples are mobile and non-mobile devices fitted with sensors.
- Local network: this component analyzing and processing the local operation.
- Internet: this is the global network through which the data is transferred for remote processing and analytic.
- Back-end-service: handling web service application comprising of processors, servers and also activating actuators.
- Application: this includes the applications for communication, processing , analytic and storage.

IOT Components and Implementation

- User interface: this is the interface between human and the machine to access IOT.

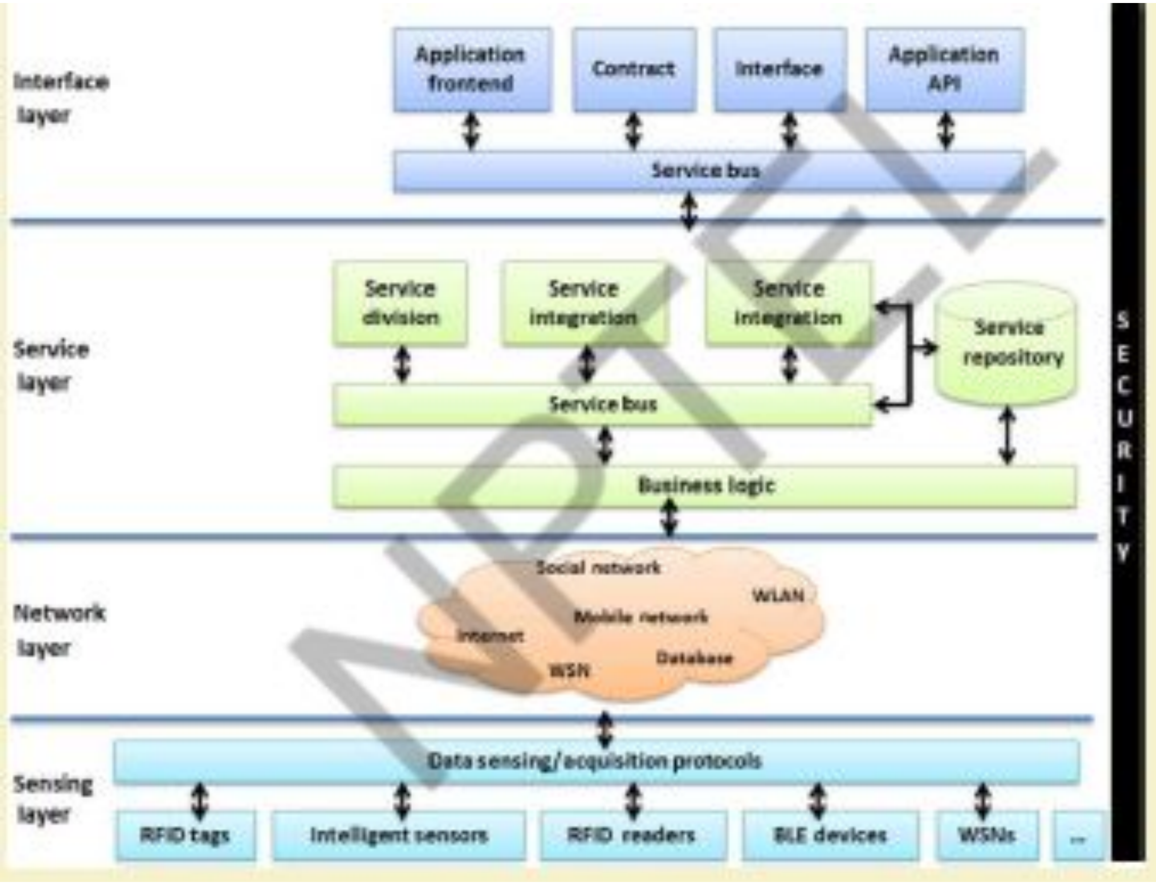




For implementing, the IOT nodes like processors, sensor fitting equipment, radio etc sense the data and talk to each other under the jurisdiction of gateway. This gateway address these devices in that Local Area Network. The gateway sends this data to the internet through the proxy server. Then the data will reach at the cloud server through web Socket. Based on the analytics and inferences, the actuation takes place on the sensed data.

Service Oriented Architecture

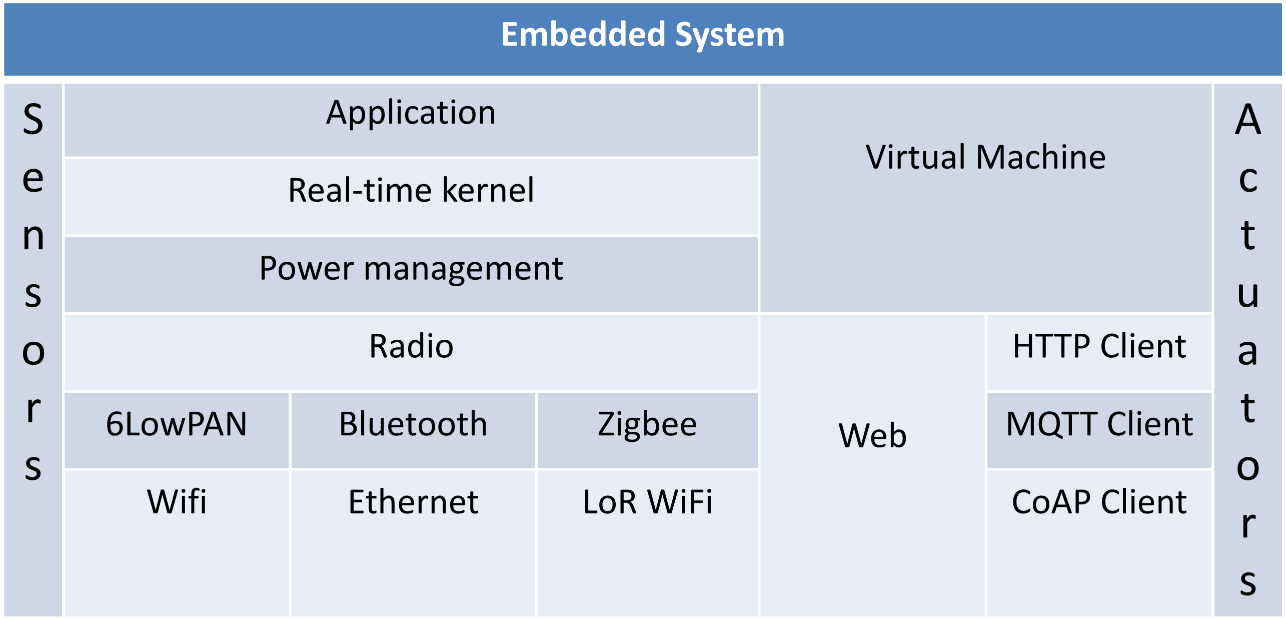
- The service oriented architecture has four layers namely sensing layer, network layer, service layer, interface layer.



IOT interdependencies

- Several interdependencies are involved in the implementation of IOT.
- In between sensors and actuators there exist a lots of technologies which are interdependent. The sensor sense the data and the data will meet the application requirements. The operating system which has a real-time kernel and power management unit , performs the duty cycle of sensors. Various radios like 6LowPAN, bluetooth, Zigbee, Wifi etc. are the connectivity technologies that help to communicate the sensed data to other nodes.

- The virtual machine will take care the virtualization of the nodes. And finally the applications like HTTP client, MQTT client etc. help for the functioning of IOT. All these devices together constitute the embedded devices.



Embedded System

Sensors

Application

Real-time kernel

Power management

Radio

6LowPAN

Bluetooth

Zigbee

Wifi

Ethernet

LoR WiFi

Virtual Machine

Web

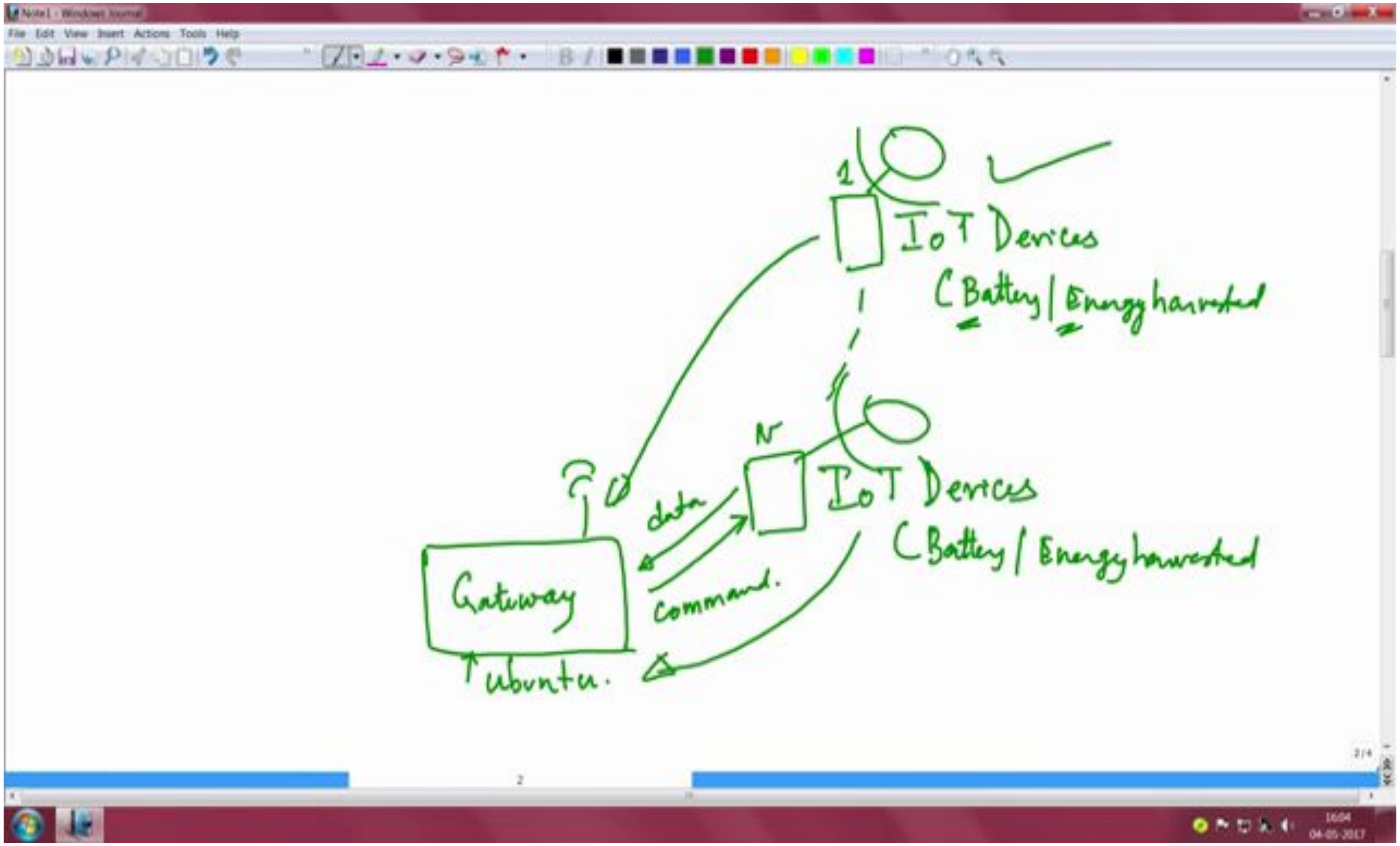
HTTP Client

MQTT Client

CoAP Client

Actuators

IOT Implementation

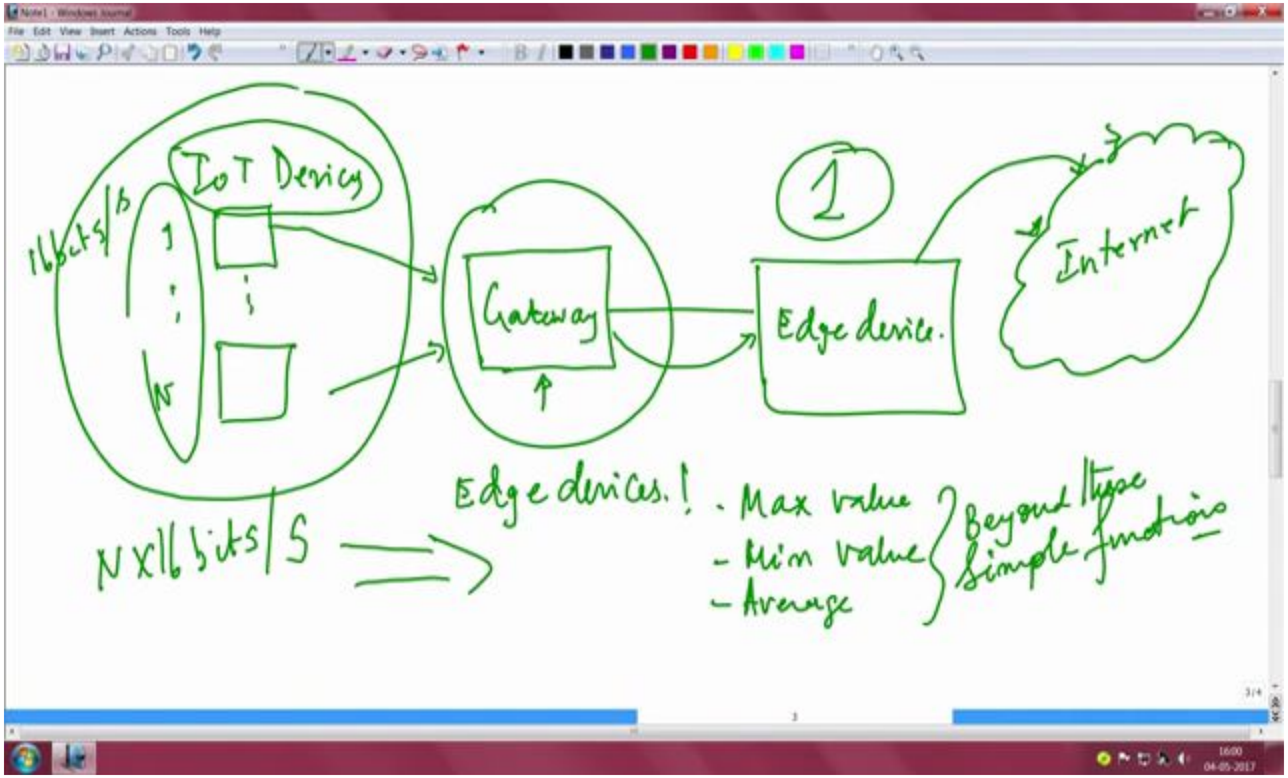

Note1 - Windows Journal
File Edit View Insert Actions Tools Help
1
IoT Devices
(Battery | Energy harvested
N
IoT Devices
(Battery / Energy harvested
data
command.
Gateway
ubuntu.
2
2/4
16:04
04-05-2017

IOT Implementation

- This picture essentially has IoT devices.
- 1 to n such devices may be present
- These IoT devices either battery driven or energy harvested .
- All these IoT devices basically are essentially a basically equipped with sensors which sense something from the environments .
- Basically all these sensing environments are analog by nature, but nowadays analog information is directly processed .
- Essentially, one could be processing the analog information analog data into digital data and making it available directly onto these devices.
- All of them are pushing data to what is known as this gateway device.

- An IoT device may not even exceed the size of an ATM card
- This is a small PCB which has some circuitry , chip, connectors and has some sort of a storage.
- But the gateway essentially is bigger in size which includes the raspberry pi ,beagleboard , odroid ,Edison, Galileo ,arduino based boards etc.
- Arduino based board gatewaye, is an important hardware which is available and very popular in its usage.
- These gateway run an embedded OS typically something like open source ubuntu which is a LINUX distribution.

IOT Implementation



Edge devices!
- Max value
- Min value
- Average

Beyond these simple functions

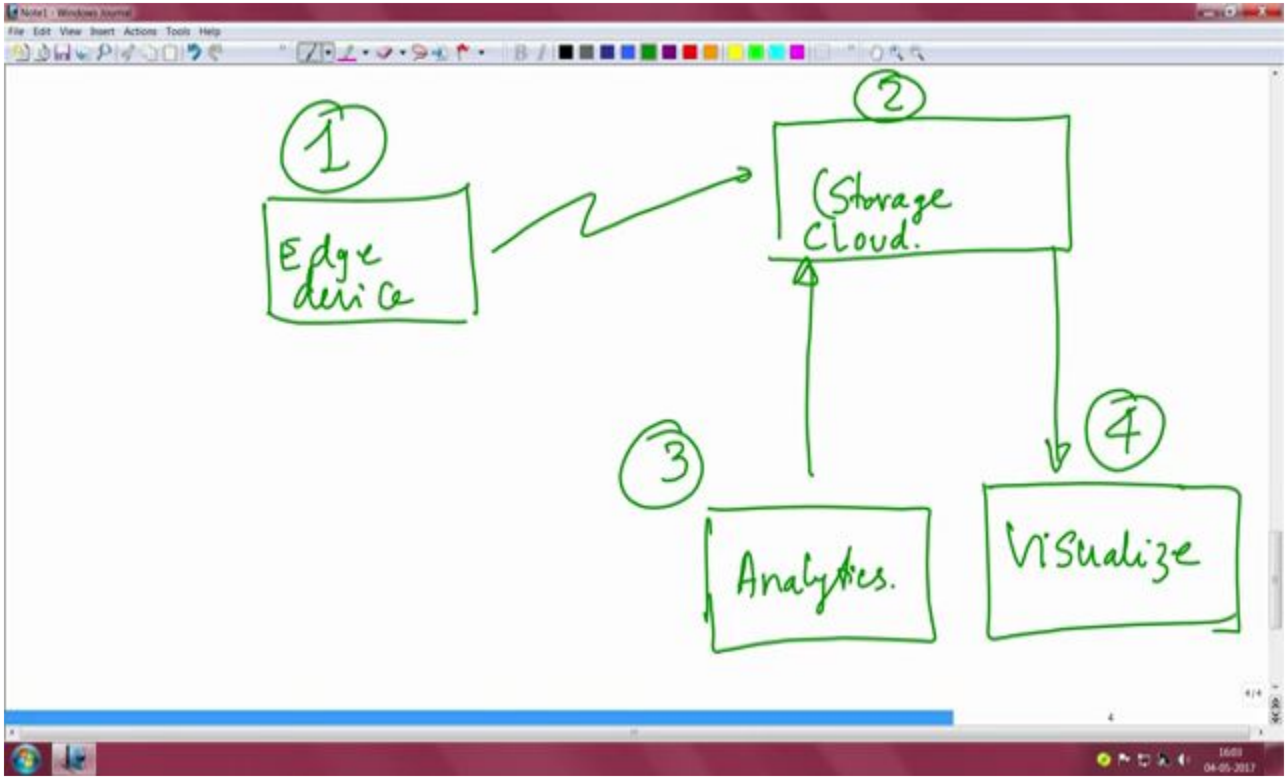
IOT Implementation

- These IOT devices sense any event occurs in the environment and send it to gateway.
- Since the sensors are low power devices they might not able to send the data directly to the any storage like cloud through internet because it needs to convert these data to TCP/IP protocol which consume more energy.

IOT Implementation

- Hence the sensed data will be sent to Gateway which is capable of converting these data to TCP/IP and send it to cloud through internet.
- But the data should be refined before send it to the cloud. This job can be done through the EDGE devices.
- Hence the edge device analyze the data received from gateway and send it to the cloud through any network.

IOT Implementation

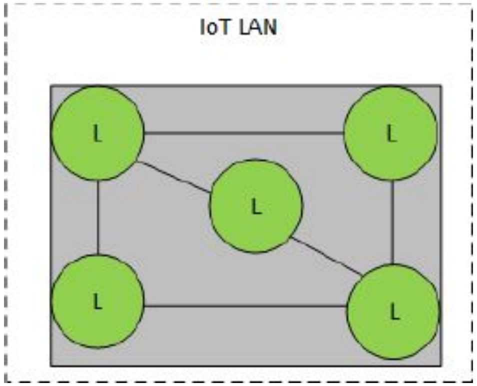


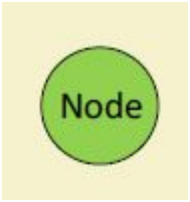
IoT Networking

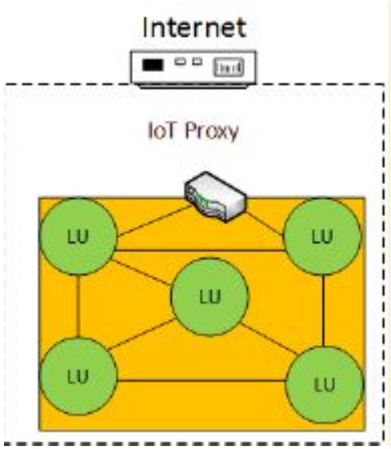
Connectivity Terminologies

The basic connectivity terminologies in IoT are

- IoT node: These are machines, things or computers connected to other nodes inside a LAN.



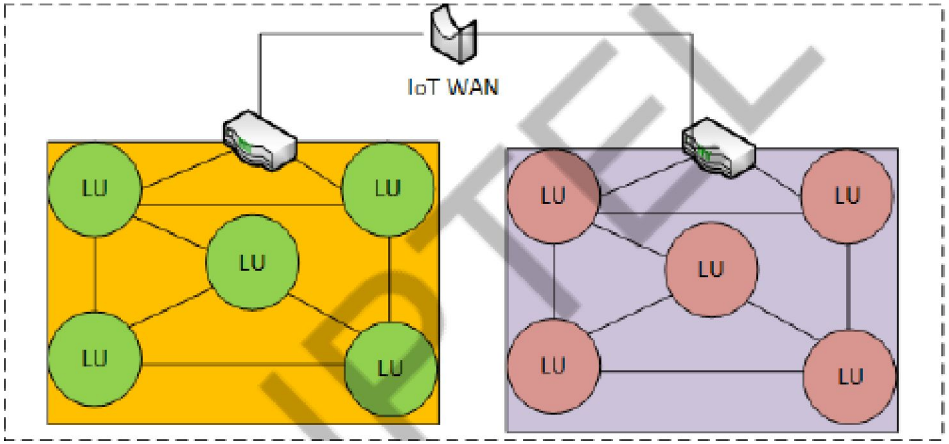




- IoT LAN: These are local, short range communication network such as building or organization.
- IoT Proxy: It performs active application layer functions between IoT nodes and other entities.

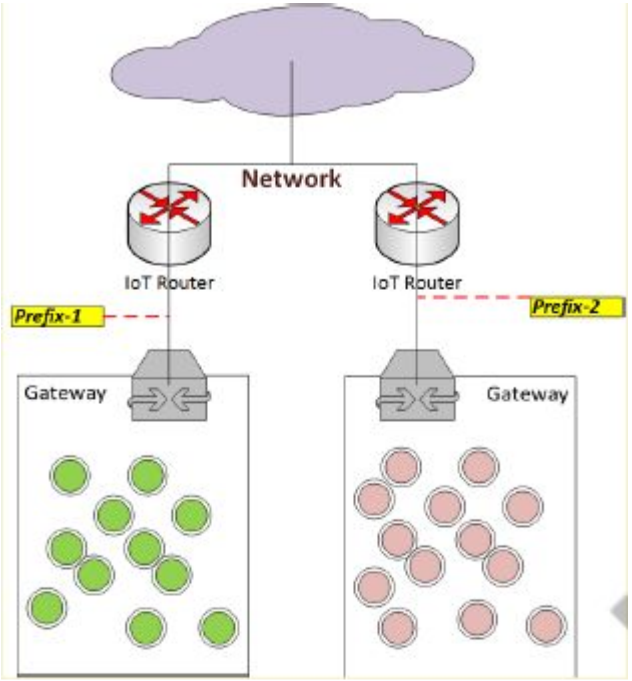
IoT WAN: It is the connection of various network segments organizationally and geographically wide which connect o the internet.

IoT Gateway: it is a router connecting the IoT LAN to WAN or to the internet. It can implement several LAN and WAN. The main responsibility of IoT gateway is to forward packets between LAN and WAN on the IP layer.



Gateway prefix Allotment:

- In IoT network, each device has a unique IP address .
- Nodes within the gateway's jurisdiction have addresses that are valid within the gateway's domain only.
- The same address may be repeated in the domain of another gateway.
- The gateway has unique network prefix which can be used to identify them globally.
- The network connected to the internet has routers with their set of addresses and ranges.
- The router may have multiple gateways connected to them which can forward packets from the nodes to the internets via the routers.
- These routers assign prefixes to gateway under them so that the gateway can be identified with them.



Multihoming

- Multihoming is the practice of connecting a host or a computer network to more than one network.
- This can be done in order to increase reliability or performance.
- A typical host or end-user network is connected to just one network. Connecting to multiple networks can increase reliability because if one connection fails, packets can still be routed through the remaining connection.
- Connecting to multiple networks can also improve performance because data can be transmitted and received through the multiple connections simultaneously multiplying throughput and, depending on the destination, it may be more efficient to route through one network or the other.

There are several different ways to perform multihoming.

Host multihoming:

- A single host may be connected to multiple networks.
- For example, a mobile phone might be simultaneously connected to a Wi-Fi network and a 3G network, and a desktop computer might be connected to both a home network and a VPN.
- A multihomed host usually is assigned multiple addresses, one per connected network.

Classical multihoming:

- In classical multihoming, a network is connected to multiple providers, and uses its own range of addresses .
- The network's edge routers communicate with the providers using a dynamic routing protocol, typically BGP, which announces the network's address range to all providers.
- If one of the links fails, the dynamic routing protocol recognizes the failure within seconds or minutes, and reconfigures its routing tables to use the remaining links, transparently to the hosts.

- Classical multihoming is costly, since it requires the use of address space that is accepted by all providers, a public Autonomous System (AS) number, and a dynamic routing protocol. Since multihomed address space cannot be aggregated, it causes growth of the global routing table.

Multihoming with multiple addresses:

- In this approach, the network is connected to multiple providers, and assigned multiple address ranges, one for each provider.
- Hosts are assigned multiple addresses, one for each provider.
- Multihoming with multiple addresses is cheaper than classical multihoming, and can be used without any cooperation from the providers.

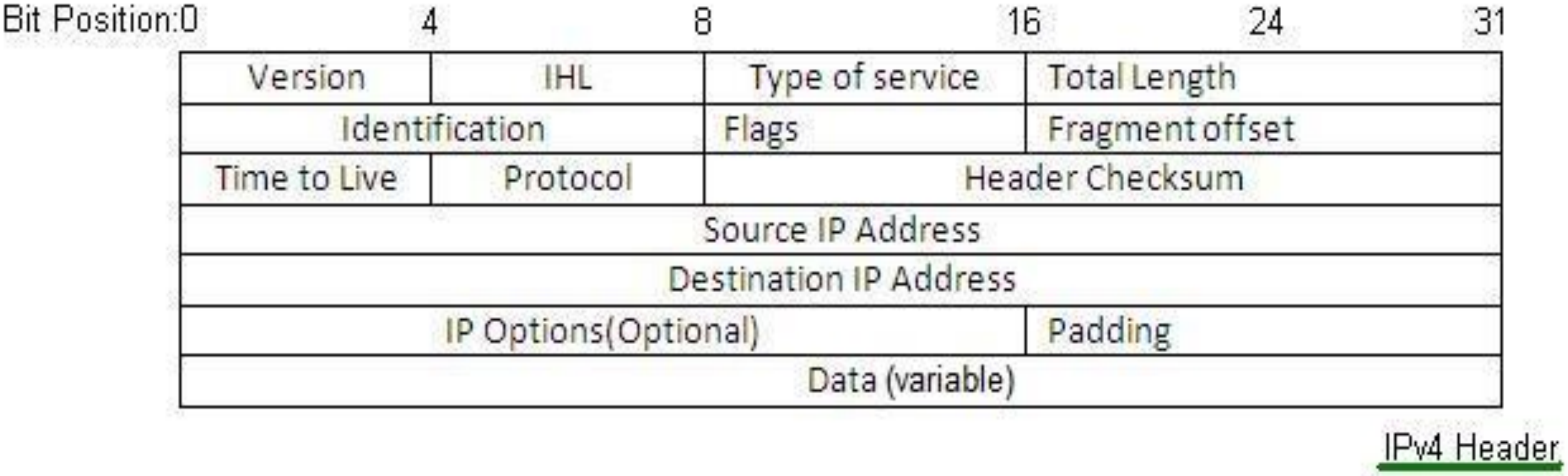
- Multihoming with multiple address requires additional technology in order to perform routing:
  - for incoming traffic, hosts must be associated with multiple A or AAAA DNS records so that they are reachable through all providers;
  - for outgoing traffic, a technique such as source-specific routing must be used to route packets through the correct provider, and reasonable source address selection policies must be implemented by hosts.

Internet Protocol (IP)

- Both IPV4 and IPV6 are IP address schemes available to assign IP addresses to the computers connected to the network.
- IP is a layer-3 protocol in OSI stack.
- Both IPV4 and IPV6 coexist in a multiprotocol configuration.
- To do this task, network access layer should support multiplexing of IPV4 and IPV6 packets.
- There are three methods of transition strategies from IPV4 to IPV6 protocol format. They are dual stack, tunnelling and header translation.
- Packets are routed with the help of destination address.
- The functions of IP are as follows:
  - Connectionless best effort data delivery based on destination ip address.
  - Fragmentation and re-assembly of datagrams to support links with different MTUs(Maximum Transmission Units).

IPv4

IPV4 has 32 bit IP address space. There are different classes Class A, Class B, Class C, Class D and Class E. The figure depicts IPV4 header fields used in the IP protocol. IPV4 is defined in RFC 791.



Bit Position:0 4 8 16 24 31

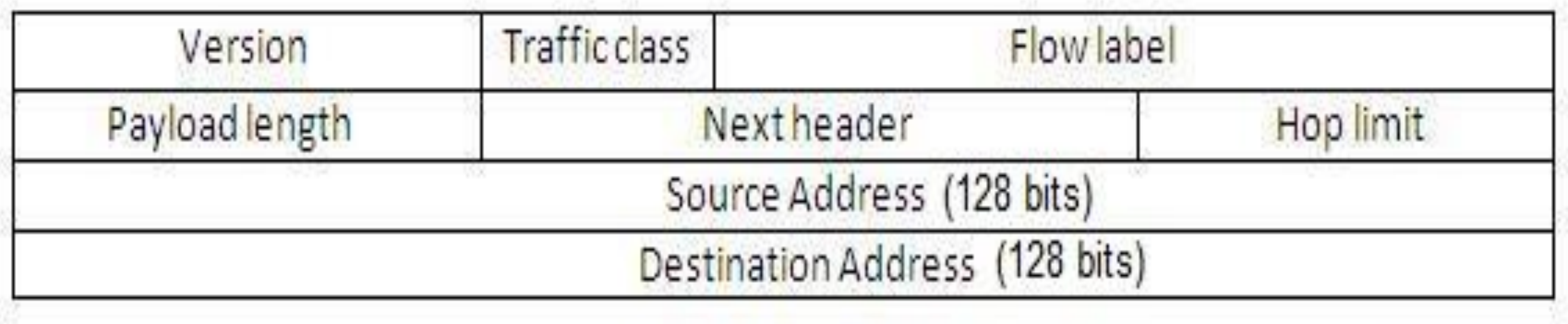
Version	IHL	Type of service	Total Length
Identification		Flags	Fragment offset
Time to Live	Protocol	Header Checksum	
Source IP Address			
Destination IP Address			
IP Options(Optional)			Padding
Data (variable)			

IPv4 Header

IPV4 header field	Description
Version	It signifies version of IPV4 or IPV6 in use
IHL(Header length)	datagram length in 32 bit words
Type of Service	specify how upper layers would like datagram to be handled.
Total Length	(data + header) size in bytes before fragmentation
Identification	helps in reassembly of fragments of datagram. Same ID has been assigned to all the fragments of a datagram
Flags(3bits)	2 lower bits are used, lowest one signify whether the packet can be fragmented or not, middle one specify whether the packet is the last one in the series of fragments of the packet
Fragment offset	indicates offset position of data fragment from starting position of datagram
Time to Live	counter which decrements to zero at the point of discarded datagram. This helps to prevent any misrouted packet.
Protocol	Indicates whether higher layer is TCP (value 6) or UDP(value 17)
Header hecksum	Requires for error detection at the destination host
Source address	Address of sending node
Destination address	Address of receiving node
Options	Support various other options such as security etc.
Data	Contains upper layer data information

IPv6

- IPV6 has 128 bit IP address, which helps support one billion networks, hence extends the drawbacks of IPV4 system. The figure depicts IPV6 header fields used in the IP protocol. IPv6 is defined in RFC 2460.



<table>
<tr><td>Version</td><td>Traffic class</td><td colspan="2">Flow label</td></tr>
<tr><td>Payload length</td><td colspan="2">Next header</td><td>Hop limit</td></tr>
<tr><td colspan="4">Source Address (128 bits)</td></tr>
<tr><td colspan="4">Destination Address (128 bits)</td></tr>
</table>

IPv6 Header

Following Table mentions all the fields of IPV6 header with functional description.

IPV6 header field	Description
Version(4bit)	contains 0110
Traffic Class(8 bit)	equivalent to Type of Service field of IPV4, used to classify IPv6 priorities
Flow label(20 bit)	Used by source node to label sequence of packets for which it requests a special type of handling by IPv6 routers.
Payload length(16 bit)	Length of payload in bytes
Next Header(8 bit)	indicates type of header IPv6 follows such as TCP, UDP, ICMPv4 or ICMPv6
Hop Limit(8 bit)	Decrements 1 by 1 after forwarding the packets by each nodes. When zero packets are discarded and error message is being returned This is equivalent to Time to Live field of IPv4 protocol header.
Source address(128 bit)	Origin of IPv6 packet
Destination address(128 bit)	destination of IPv6 packet

Difference between IPv4 and IPv6

IPv4	IPv6
In IPv4, there are only 2^{32} possible ways to represent address, which is about 4 billion.	In IPv6, there are 2^{128} possible ways.
IPv4 address is written by dotted decimal notation Example: 121.2.8.12	IPv6 address is written in hexadecimal notation consisting of 8 groups with 4 hexadecimal digits or 8 groups of 16 bits each. Example: FABC:AC77:7834:2222:FACB:AB98:5432:4567
IPv4 header consists of 20 (minimum) to 60 bytes (maximum). It houses 13 fields.	IPv6 header consists of 40 bytes in length and houses only 8 fields.
IPv4 header has checksum which must be computed by each router.	IPv6 header does not use any checksum.
IPv4 header contains 8-bit field called service type.	IPv6 header contains 8-bit field called traffic class field.
IPv4 node has only stateful auto configuration.	IPv6 node has both stateful and stateless address auto configuration mechanism.
Security in IPv4 networks is limited to tunneling between two networks.	IPv6 has been designed to satisfy growing and expanded need for network security.
Source and destination addresses are 32 bits in length.	Source and destination addresses are 128 bits in length.
IPsec support is optional.	IPsec support is required.
No identification of packet flow for QoS handled by routers is present within IPv4 header.	Packet flow identification for QoS handled by routers is included in IPv6 header using "flow label field".
ARP (Address Resolution Protocol) uses broadcast ARP request frames to resolve an IPv4 address to a link layer address.	ARP request frames are replaced with multicast neighbour solicitation messages.
Must be configured either manually or through DHCP.	Does not require manual configuration or DHCP.
Header includes options.	All optional data is moved to IPv6 extension headers.
"ICMP router discovery" is used to determine IPv4 address of the best default gateway and it is optional.	"ICMP router discovery" is replaced with "ICMPv6 router solicitation and router advertisement" message and it is required.

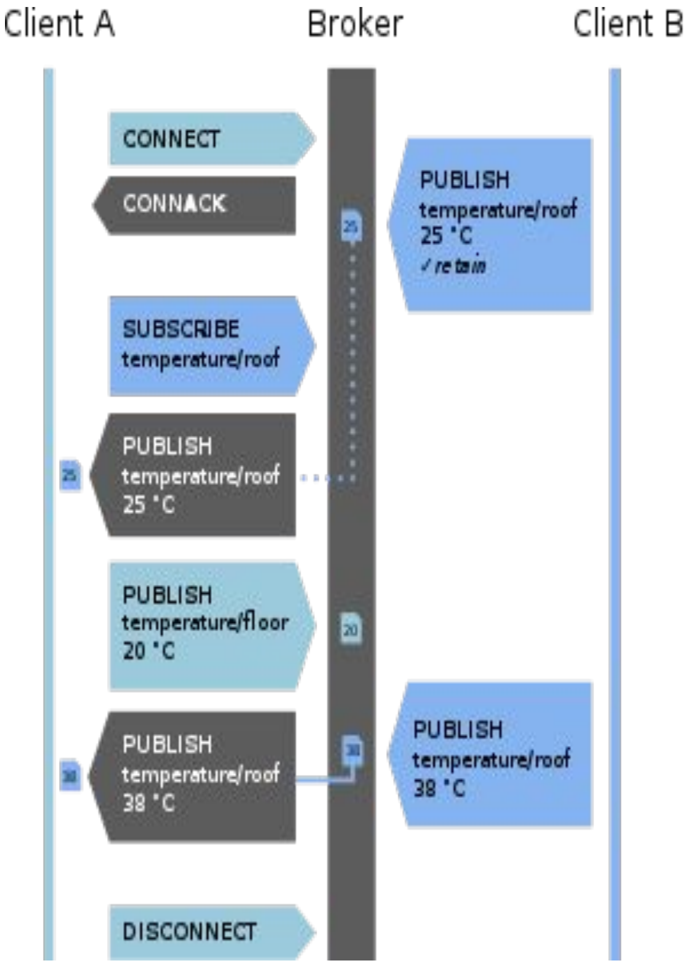
MQTT Protocol

- The Message Queuing Telemetry Transport (MQTT) is a lightweight, session layer, publish-subscribe network protocol that transports messages between devices. The protocol usually runs over TCP/IP; however, any network protocol that provides ordered, lossless, bi-directional connections can support MQTT.

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

- The MQTT protocol defines two types of network entities: a message broker and a number of clients.
- The **broker** acts as a post office, MQTT doesn't use the address of the intended recipient but uses the subject line called “Topic”, and anyone who wants a copy of that message will subscribe to that topic.
- An MQTT broker that receives all messages from the clients and then routes the messages to the appropriate destination clients.
- An **MQTT client** is any device (from a micro controller up to a fully-fledged server) that runs an MQTT library and connects to an MQTT broker over a network.
- Multiple clients can receive the message from a single broker (one to many capability). Similarly, multiple publishers can publish topics to a single subscriber (many to one).

- Information is organized in a hierarchy of **topics** which is a **simple string** that can have more hierarchy levels, which are separated by a slash.
- A sample topic for sending temperature data of the living room could be *house/living-room/temperature*.
- When a publisher has a new item of data to distribute, it sends a control message with the data to the connected broker.
- The broker then distributes the information to any clients that have subscribed to that topic.
- The publisher does not need to have any data on the number or locations of subscribers, and subscribers, in turn, do not have to be configured with any data about the publishers.



MQTT connection (QoS 0) with connect, publish / subscribe, and disconnect. The first message from client B is stored due to the retain flag

- If a broker receives a message on a topic for which there are no current subscribers, the broker discards the message unless the publisher of the message designated the message as a **retained message**.
- A retained message is a normal MQTT message with the retained flag set to true.
- The broker stores the last retained message and the corresponding QoS for the selected topic.
- Each client that subscribes to a topic pattern that matches the topic of the retained message receives the retained message immediately after they subscribe.
- The broker stores only one retained message per topic. This allows new subscribers to a topic to receive the most current value rather than waiting for the next update from a publisher.

SMQTT Protocol

- **SMQTT (Secure Message Queue Telemetry Transport)** is an extension of MQTT protocol which uses encryption based on lightweight attribute encryption. It is a session layer protocol.
- Like MQTT, it uses broker based subscribe and publish architecture.
- The main advantage of this encryption is that it has a broadcast encryption feature. In this features, one message is encrypted and delivered to multiple other nodes.
- The key generation and encryption algorithms are not standardized.
- The process of message transfer and receiving consists of four major stages: Setup, encryption, publish and decryption.

- **Setup:** In this phase, the publishers and subscribers register themselves to the broker and get a secret master key.
- **Encryption:** Before publishing the data is being encrypted. When the data is published to broker, it is encrypted by broker.
- **Publish:** The broker publishes the encrypted message to the subscribers.
- **Decryption:** Finally the received message is decrypted by subscribers with the same master key.

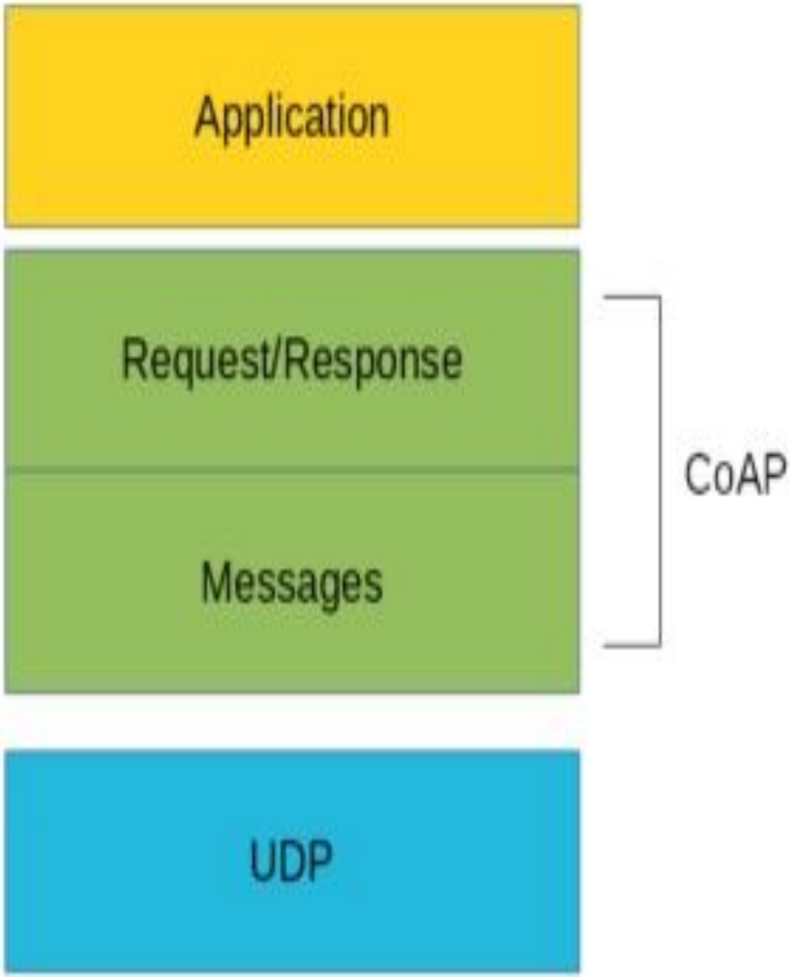
CoAP Protocol

- CoAP is an IoT Session layer protocol.
- CoAP stands for Constrained Application Protocol, and it is defined in RFC 7252.
- CoAP is a simple protocol with low overhead specifically designed for constrained devices (such as microcontrollers) and constrained networks.
- This protocol is used in M2M data exchange and is very similar to HTTP.

The main features of CoAP protocols are:

- Web protocol used in M2M with constrained requirements
- Asynchronous message exchange
- Low overhead and very simple to parse
- URI and content-type support
- Proxy and caching capabilities

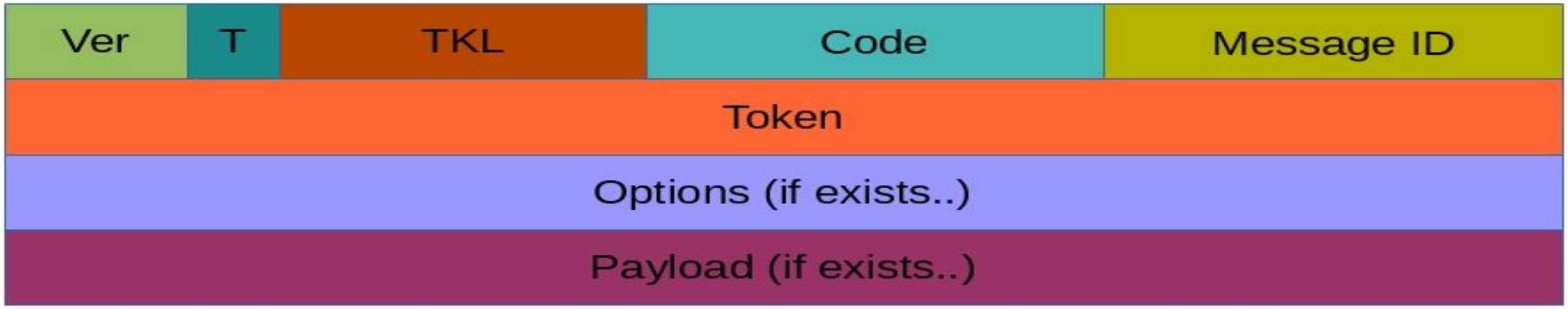
CoAP Architecture :



- It contains four layers namely Application, Request / Response, Message and UDP.
- The two main sub layers in CoAp are Messages and Request /Response sub-layer.
- The Messages layer deals with UDP and with asynchronous messages. It is responsible for reliability and remove duplication of messages.
- The Request/Response layer deals with communications i.e. manages request/response interaction based on request/response messages.

CoAP Message Format

The CoAP is the meat for constrained environments, and for this reason, it uses compact messages. To avoid fragmentation, a message occupies the data section of a UDP datagram. A message is made by several parts:



Ver T TKL Code Message ID

Token

Options (if exists..)

Payload (if exists..)

- **Ver**: It is a 2 bit unsigned integer indicating the version
- **T**: it is a 2 bit unsigned integer indicating the message type: 0 confirmable, 1 non-confirmable
- **TKL**: Token Length is the token 4 bit length
- **Code**: It is the code response (8 bit length)
- **Message ID**: It is the message ID expressed with 16 bit

Following terms are used in CoAp protocol:

- **Endpoint**: An entity that participates in the CoAP protocol. Usually, an Endpoint is identified with a host.
- **Sender**: The entity that sends a message
- **Recipient**: The destination of a message
- **Client**: The entity that sends a request and the destination of the response
- **Server**: The entity that receives a request from a client and sends back a response to the client

CoAP has four messaging mode

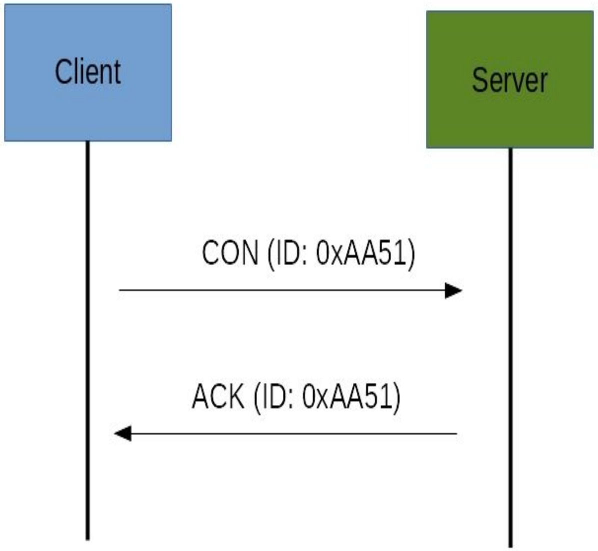
- Confirmable and Non-confirmable mode of messaging are used in Message layer.
- Piggyback and Separate mode of messaging are used in Request /Response layer.

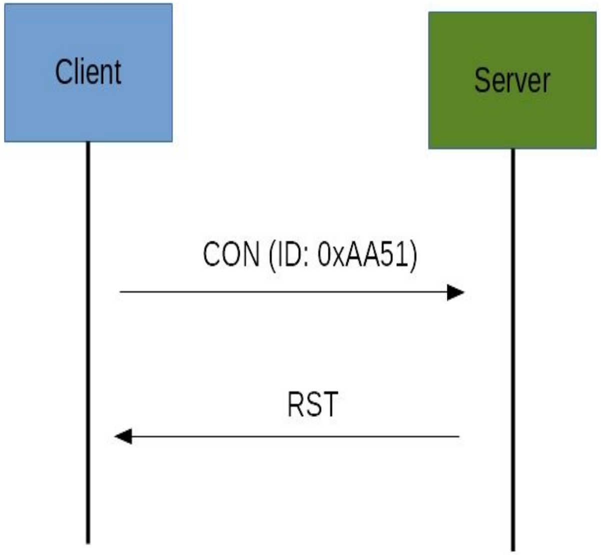
CoAP Messages Layer Model

- This is the lowest layer of CoAP.
- This layer deals with UDP exchanging messages between endpoints.
- Each CoAP message has a unique ID; this is useful to detect message duplicates.
- A CoAP message is built by these parts: A binary header, A compact options, Payload

Confirmable (CON) message Mode:

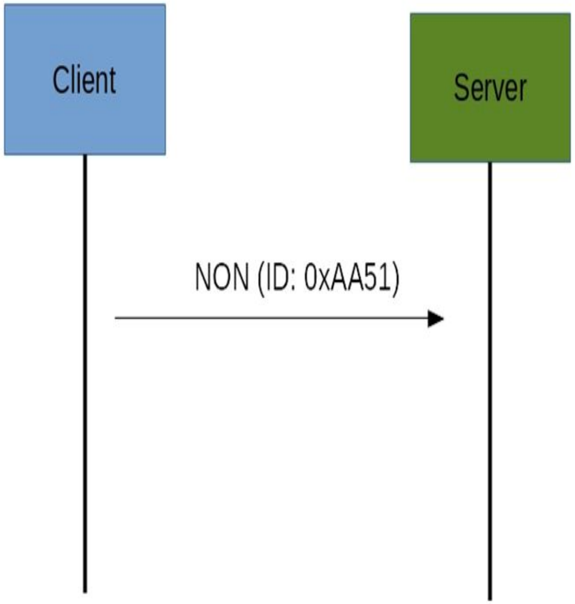
- A confirmable message is a reliable message.
- In CoAP, a reliable message is obtained using a Confirmable message (CON).
- Using this kind of message, the client can be sure that the message will arrive at the server.
- A Confirmable message is sent again and again until the other party sends an acknowledge message (ACK).
- The ACK message contains the same ID of the confirmable message (CON).
- If the server has troubles for the incoming request, it can send back a Reset message (RST) instead of the Acknowledge message (ACK)





Non-Confirmable (NON) message Mode:

- These are messages that don’t require an Acknowledge by the server.
- Here the client send a message and does not wait for the acknowledgement.
- They are unreliable messages or in other words messages that do not contain critical information that must be delivered to the server.
- This category belongs messages that contain values read from sensors.
- Even if these messages are unreliable, they have a unique message ID for supervising in case of retransmission.



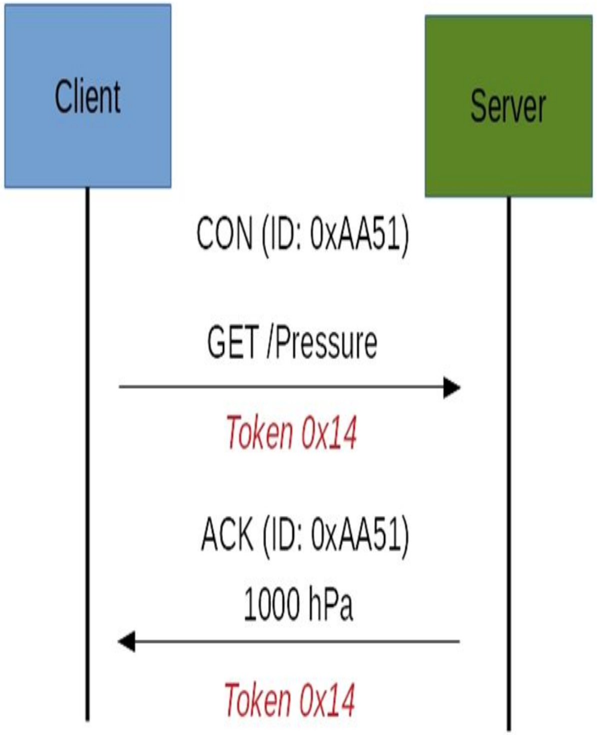
CoAP Request/Response Layer Model :

- The CoAP Request/Response is the second layer in the CoAP abstraction layer.
- The request is sent using a Confirmable (CON) or Non-Confirmable (NON) message.
- There are several scenarios depending on if the server can answer immediately to the client request or the answer if not available.

.

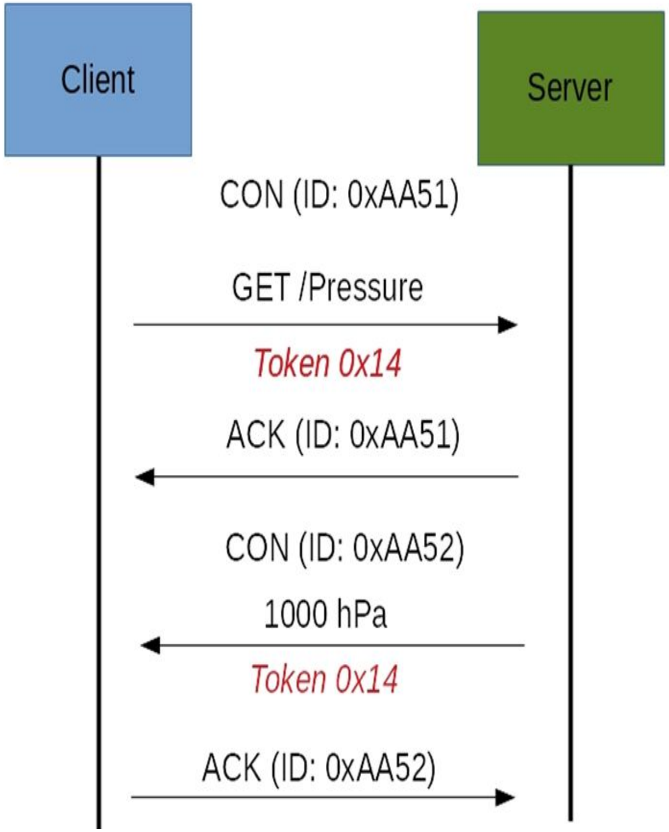
Piggyback messaging mode:

- The client sends request using CON or NON type message and receives ACK with confirmable message
- If the server can answer immediately to the client request, then if the request is carried using a Confirmable message (CON), the server sends back to the client an Acknowledge message .
- For successful response ACK contains response message (token) and for failure message ACK contains failure message (Token) code.
- Here the Token is used to match the request and the response.



Separate messaging mode:

- If the server can't answer to the request coming from the client immediately, then it sends an Acknowledge message with an empty response.
- The server wait for sometimes and as soon as the response is available, then the server sends a new Confirmable message to the client containing the response.
- After receiving the response from the server, the client sends back an Acknowledge message.
- If the request coming from the client is a NON-confirmable message, then the server answer using a NON-confirmable message.



AMQP Protocol

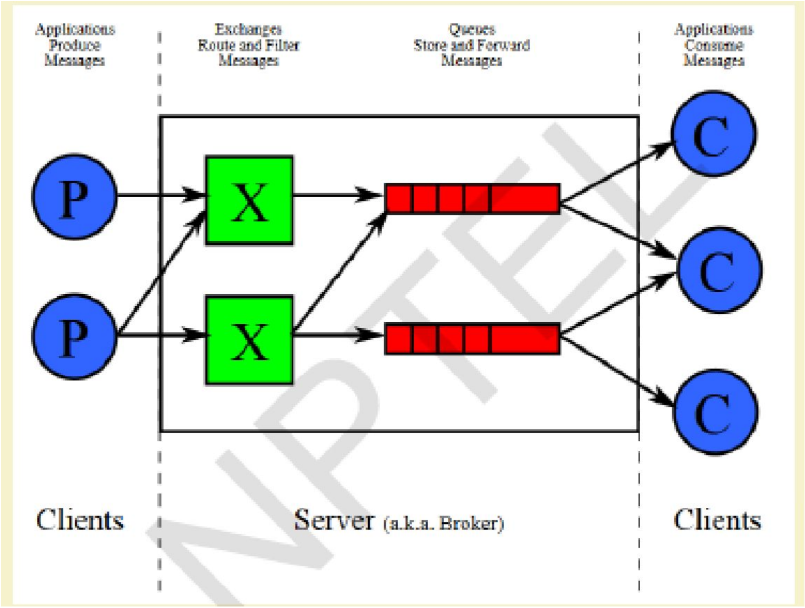
- Advanced message Queuing Protocol is an open standard application layered protocol for message-oriented middleware.
- AMQP is a binary protocol which enables encrypted and interoperable messaging between organization and applications.
- AMQP connects systems, feeds business processes with the information they need and reliably transmits onwards the instructions that achieve their goals.
- AMQP connects across organizations, technologies, time and space.
- AMQP is efficient, portable, multi channel and secure protocol. It is fast and features guaranteed delivery with acknowledgement of received message.

AMQP Features:

- AMQP is reliable, interoperable, open standard, secure protocol having routing and queuing features.
- Targeted QoS (Selectively offering QoS to link).
- Persistence (message delivery guarantee)
- Delivery of message to multiple consumers.
- Possibility of ensuring multiple consumption.
- Possibility of preventing multiple consumption.
- High speed protocol.

AMQP Architecture

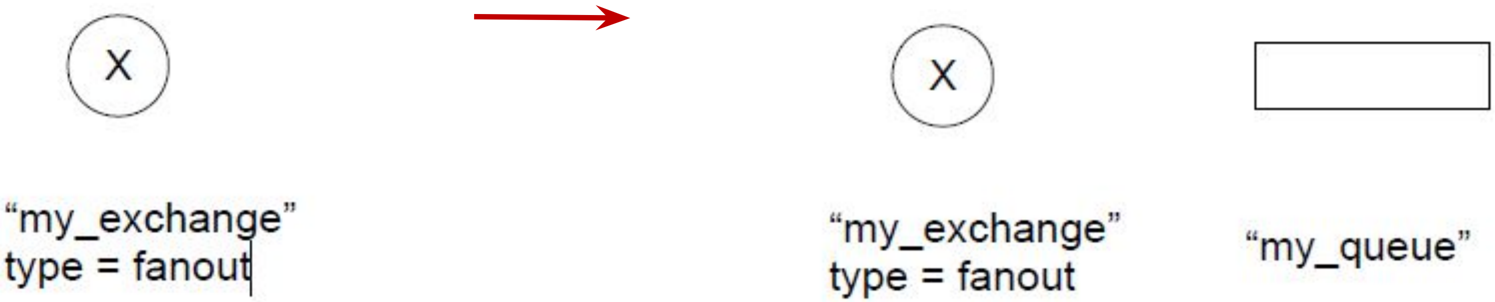
- There are applications that produce messages on one end (P).
- There are applications that consume messages on other end (C).
- X represents exchanges that route and filter the messages.
- The server in between producers and consumers contains brokers that receives messages and routes them to queues.
- The queue store and forward the messages to business clients.
- There will be a separate queues for separate business process.
- The consumer receive the messages from these queue which is done by proper binding rules.


Applications Produce Messages
Exchanges Route and Filter Messages
Queues Store and Forward Messages
Applications Consume Messages
P
P
X
X
C
C
C
Clients
Server (a.k.a. Broker)
Clients

AMQP uses 4 types of exchange.

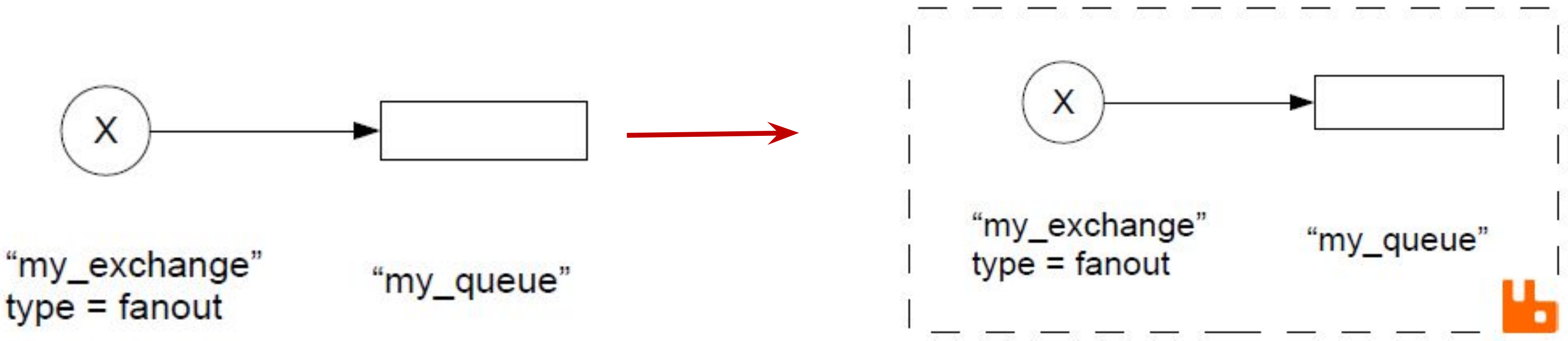
- Fan-out exchange: It completely ignores the routing key and sends any message to all the queues bound to it. This is usually done for the purpose of distributing messages like notifications, sharing messages, updates and application states to multiple clients. The exchange will discard all messages which do not follow any binding rules.

Create an exchange... ...create a queue...



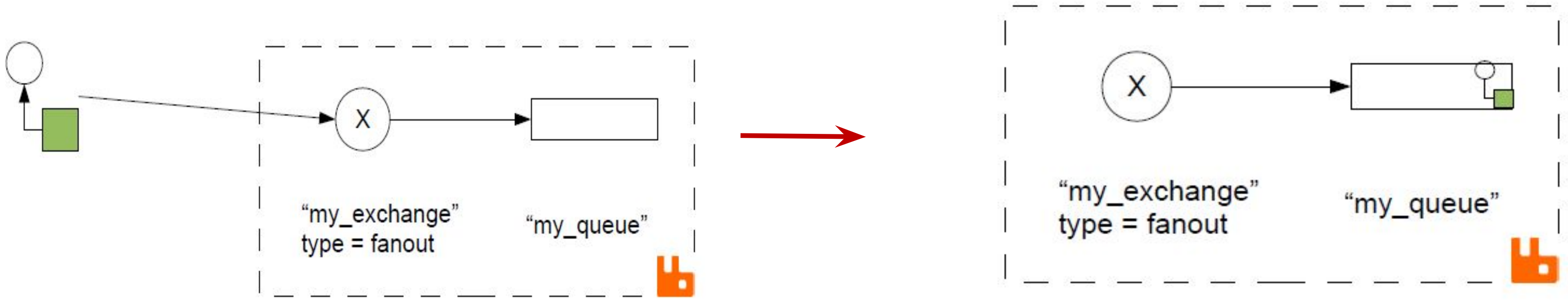
...add a binding...

...all inside a broker...

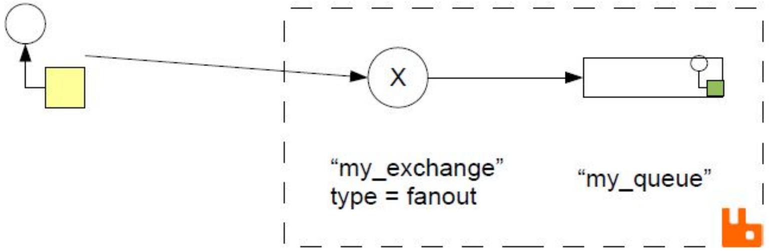


...publish a message...

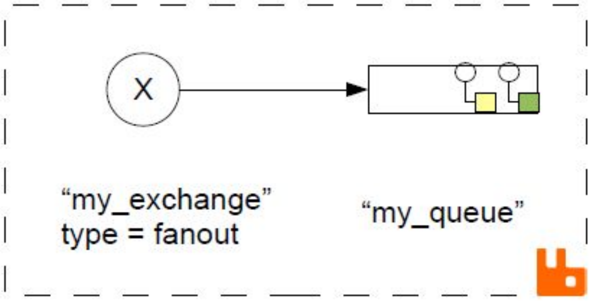
...message sits in queue.



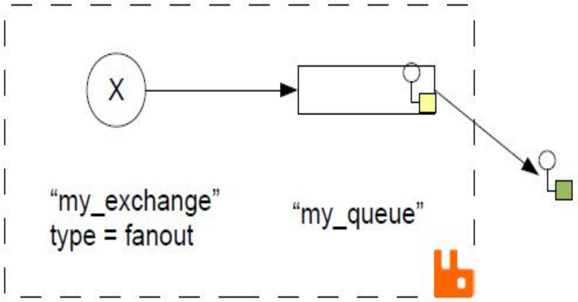
Publish another message...



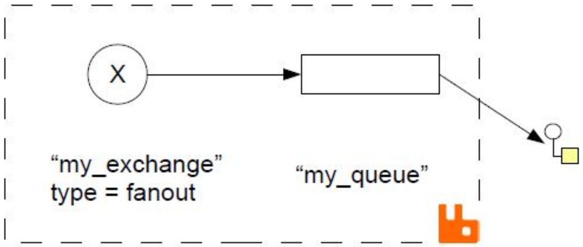
...it also sits in queue.



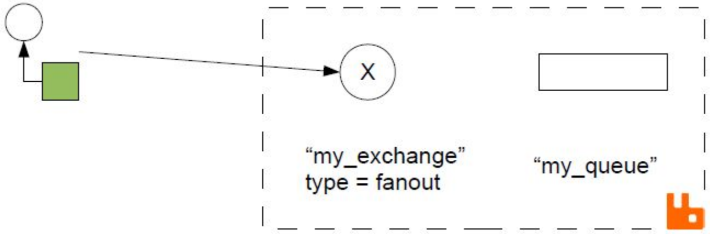
Consumer retrieves messages in order...



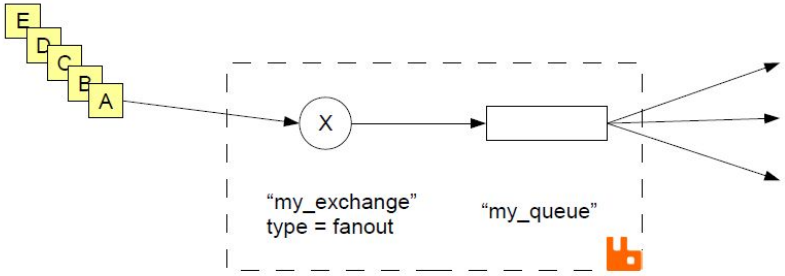
Consumer retrieves messages in order...



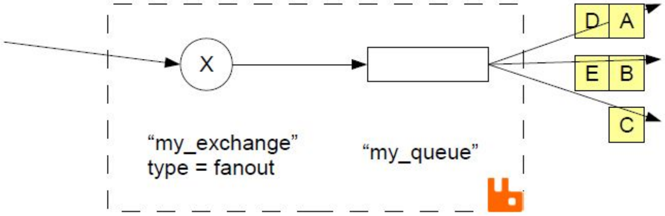
With no binding, the message is discarded
(producer can ask to be notified)



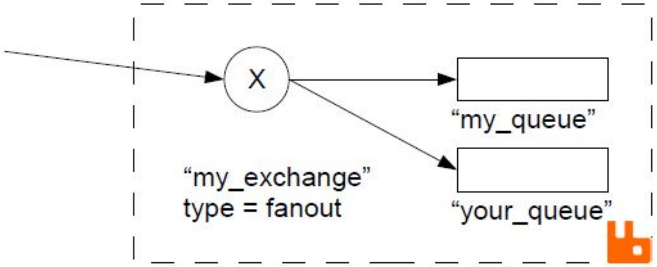
Publish several messages...



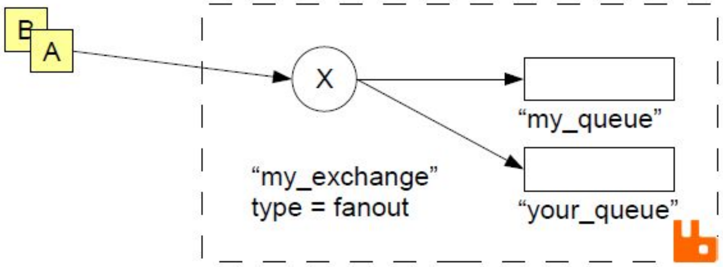
and they are distributed amongst consumers
on the same queue...



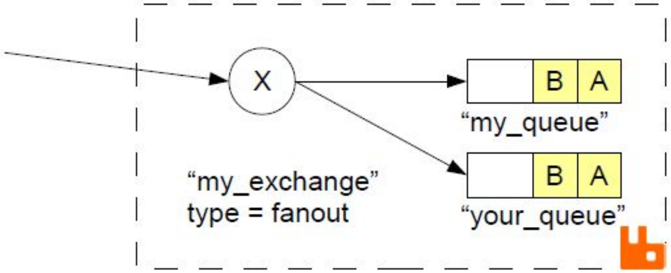
Let's create another queue and bind it to the same exchange...



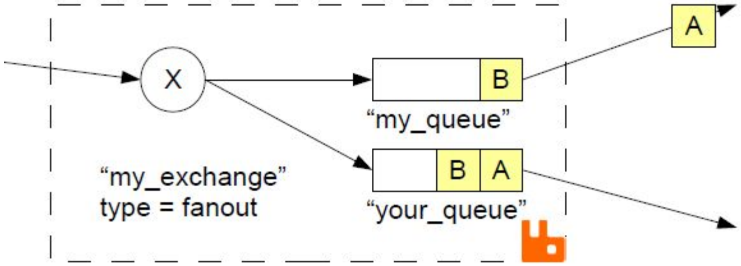
..with a *fanout* exchange...



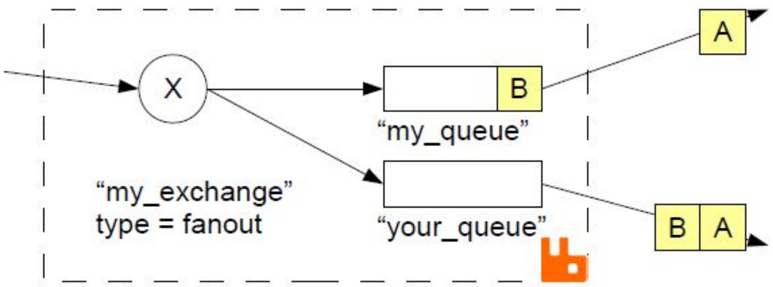
..all messages go to *every* queue...



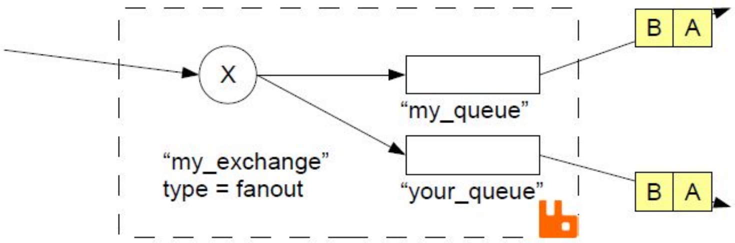
...and can be consumed...



...at different rates...



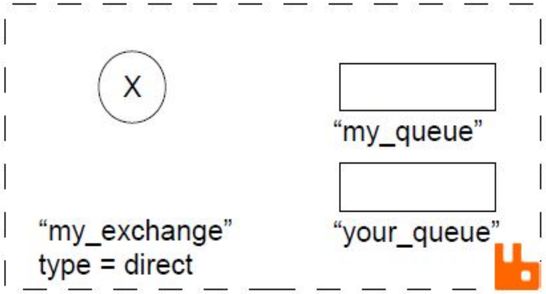
...depending on the consumer, network and broker



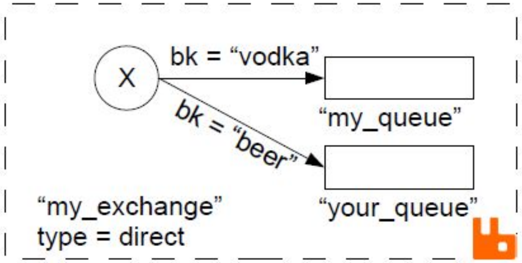
Direct Exchange:

- It involves the delivery of messages to queues based on routing keys.
- Routing keys can be considered as additional data defined to set where the message will go.
- It is basically used in load balancing task in a round robin way between the workers.

Let's change the exchange to be a *direct* type...

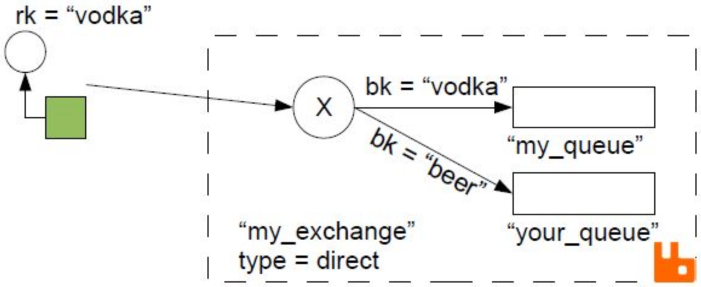

X
"my_queue"
"your_queue"
"my_exchange"
type = direct

...then the bindings have to have a key...


X
bk = "vodka"
"my_queue"
bk = "beer"
"your_queue"
"my_exchange"
type = direct

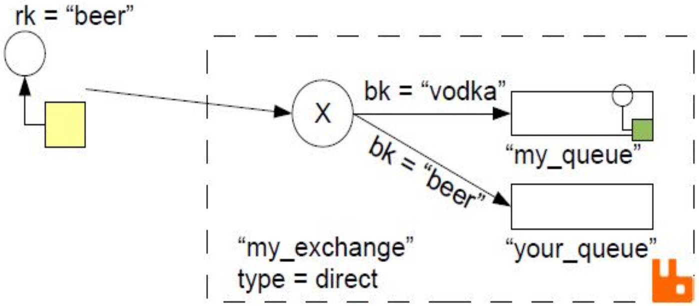
bk = Binding Key

...and the messages have to have a routing key...


rk = "vodka"
X
bk = "vodka"
"my_queue"
bk = "beer"
"your_queue"
"my_exchange"
type = direct

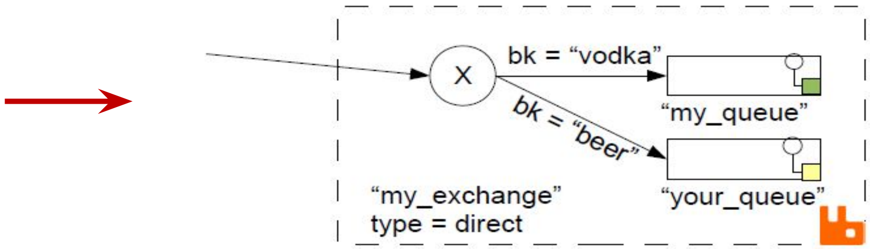
rk = Routing Key
bk = Binding Key

...a direct exchange matches the routing key to a binding key...


rk = "beer"
X
bk = "vodka"
"my_queue"
bk = "beer"
"your_queue"
"my_exchange"
type = direct

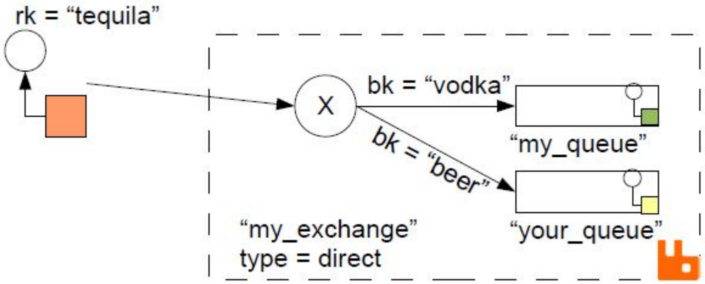
rk = Routing Key
bk = Binding Key

...a direct exchange matches the routing key to a binding key...



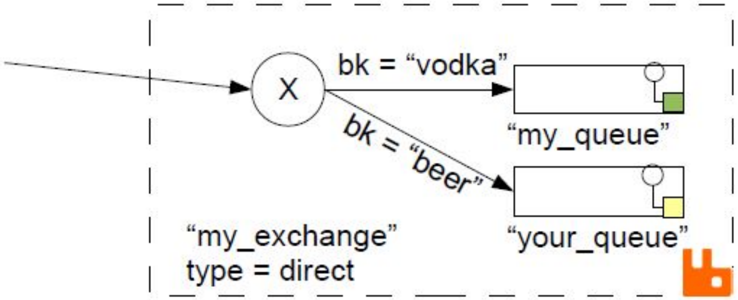
rk = Routing Key
bk = Binding Key

...messages with no matching binding...



rk = Routing Key
bk = Binding Key

...are discarded



rk = Routing Key
bk = Binding Key

Header Exchange:

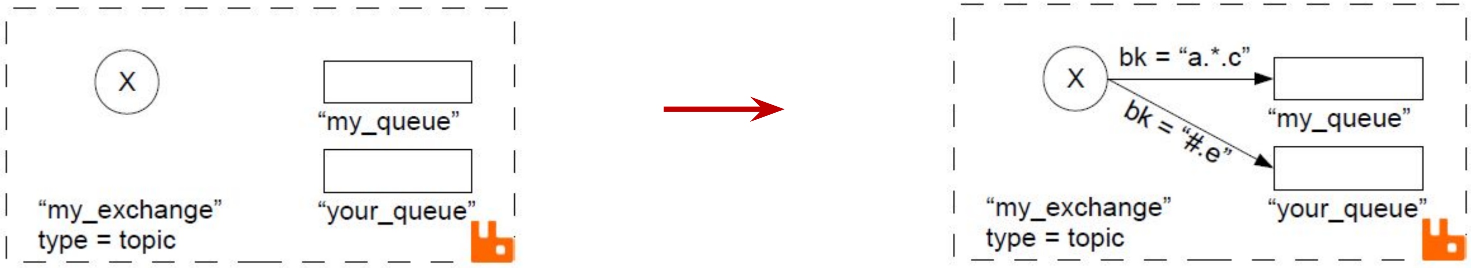
- It is similar to direct exchange type but uses a additional header coupled with messages instead of depending on routing key for routing to queues.

Topic Exchange:

- It is mainly used for publish/subscribe patterns.
- Using this type of transferring , a routing key alongside binding of queues to exchange , are used to match and send messages.
- Whenever a specialized involvement of consumer is necessary, topic exchange comes in handy to distribute messages accordingly based on keys and patterns.

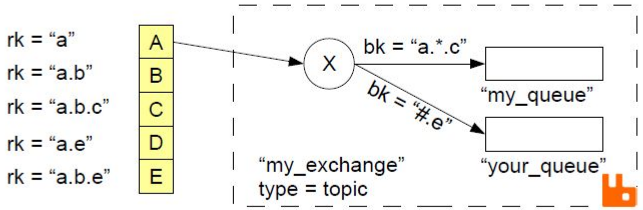
Let's change the exchange to be a *topic* type..

...permits wildcard in bindings...



bk = Binding Key

Binding keys are period(.)-separated:
* matches 1 element
matches 0 or more elements



rk = “a”
rk = “a.b”
rk = “a.b.c”
rk = “a.e”
rk = “a.b.e”

A
B
C
D
E

X

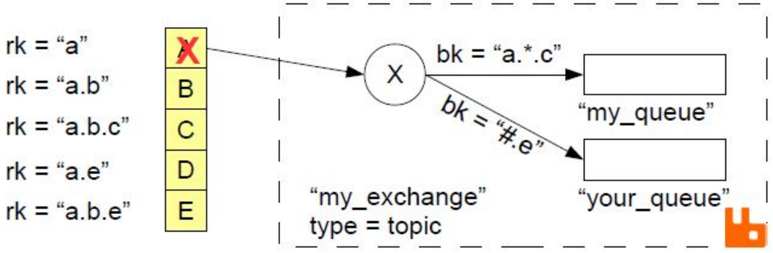
bk = “a.*.c”
bk = “#.e”

“my_queue”
“your_queue”

“my_exchange”
type = topic

rk = Routing Key
bk = Binding Key

Binding keys are period(.)-separated:
* matches 1 element
matches 0 or more elements



rk = “a”
rk = “a.b”
rk = “a.b.c”
rk = “a.e”
rk = “a.b.e”

A
B
C
D
E

X

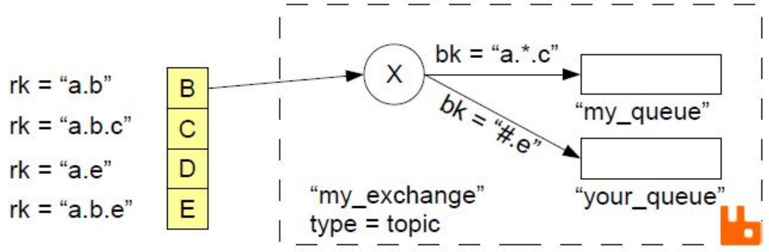
bk = “a.*.c”
bk = “#.e”

“my_queue”
“your_queue”

“my_exchange”
type = topic

rk = Routing Key
bk = Binding Key

Binding keys are period(.)-separated:
* matches 1 element
matches 0 or more elements



rk = “a.b”
rk = “a.b.c”
rk = “a.e”
rk = “a.b.e”

B
C
D
E

X

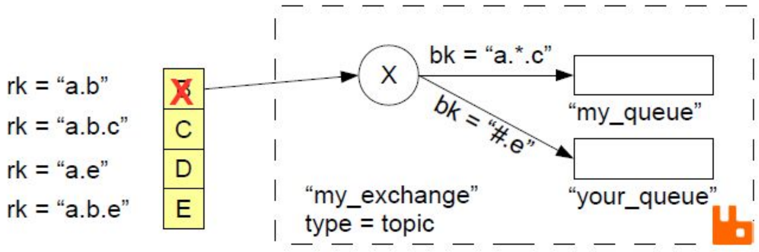
bk = “a.*.c”
bk = “#.e”

“my_queue”
“your_queue”

“my_exchange”
type = topic

rk = Routing Key
bk = Binding Key

Binding keys are period(.)-separated:
* matches 1 element
matches 0 or more elements



rk = “a.b”
rk = “a.b.c”
rk = “a.e”
rk = “a.b.e”

B
C
D
E

X

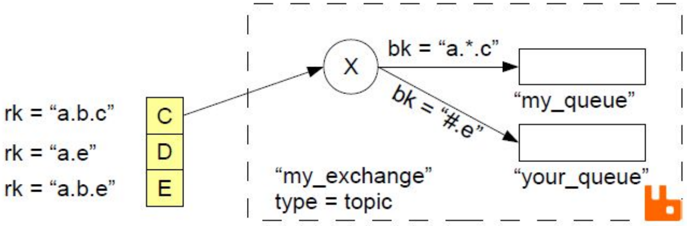
bk = “a.*.c”
bk = “#.e”

“my_queue”
“your_queue”

“my_exchange”
type = topic

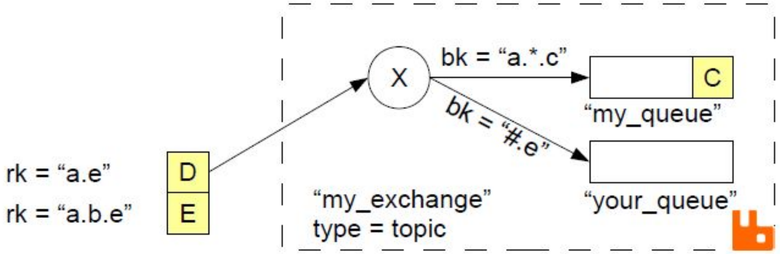
rk = Routing Key
bk = Binding Key

Binding keys are period(.)-separated:
* matches 1 element
# matches 0 or more elements



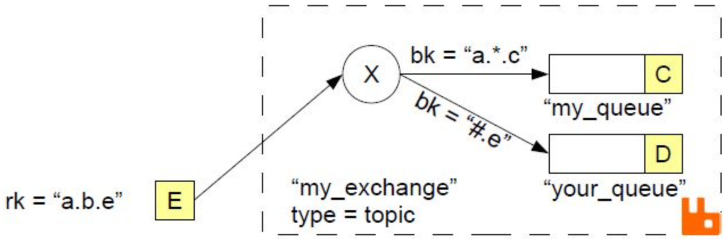
rk = Routing Key
bk = Binding Key

Binding keys are period(.)-separated:
* matches 1 element
# matches 0 or more elements



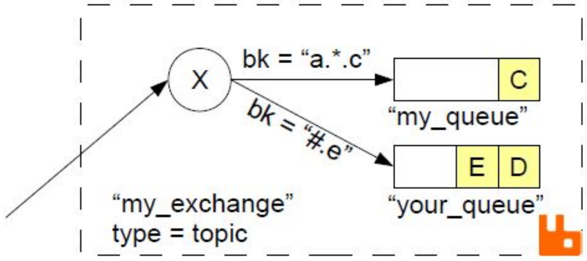
rk = Routing Key
bk = Binding Key

Binding keys are period(.)-separated:
* matches 1 element
# matches 0 or more elements



rk = Routing Key
bk = Binding Key

Binding keys are period(.)-separated:
* matches 1 element
# matches 0 or more elements



rk = Routing Key
bk = Binding Key

SENSORS

Sensors

- A sensor detects (senses) changes in the ambient conditions or in the state of another device or a system, and forwards or processes this information in a certain manner .
- It is a device that detects and responds to some types of inputs from the physical environment.
- They perform some input functions by sensing or feeling the physical changes in characteristics of a system in response to a stimuli.
- For example heat is converted to electrical signals in a temperature sensor, or atmospheric pressure is converted to electrical signals in a barometer.

Transducers

- Transducers convert or transduce energy of one kind into another.
- For example, in a sound system, a microphone (input device) converts sound waves into electrical signals for an amplifier to amplify (a process), and a loudspeaker (output device) converts these electrical signals back into sound waves.

Sensor vs. Transducer

- The word “Transducer” is the collective term used for both **Sensors** which can be used to sense a wide range of different energy forms such as movement, electrical signals, radiant energy, thermal or magnetic energy etc., and **Actuators** which can be used to switch voltages or currents.

Features or Characteristics of sensors

- Range: Since the range of the output signal is always limited, the output signal will eventually reach a minimum or maximum, when the measured property exceeds the limits. The full scale range of a sensor defines the maximum and minimum values of the measured property.
- Drift: If the output signal slowly changes independent of the measured property, this is defined as **drift**. Long term drift over months or years is caused by physical changes in the sensor.
- Sensitivity: It is defined as the change in the output per unit change in input of the property being measured. The sensitivity of a sensor under real conditions may differ from the value specified. This is called a sensitivity error. It is constant for the entire range of sensors or vary exponentially if the sensor is non-linear.

- Selectivity: It is only sensitive to the measured property. It is insensitive to any other property likely to be encountered in its application (e.g. A temperature sensor does not bother about light or pressure while sensing the temperature).

- Resolution: The resolution of a sensor is the smallest change it can detect in the quantity that it is measuring. The resolution of a sensor with a digital output is usually the smallest resolution the digital output it is capable of processing. The more is the resolution of a sensor, the more accurate is its precision.

- Response and Recovery Time: the response time is the time taken by the sensor for its outputs to reach 95% of its final value when it is exposed to target materials.

- Hysteresis: A hysteresis error causes the sensor output value to vary depending on the sensor’s previous input values. If a sensor's output is different depending on whether a specific input value was reached by increasing or decreasing the input, then the sensor has a hysteresis error. The present reading depends on the past input values.

- Linearity: If the sensitivity of a sensor is constant for the range, then it is called linearity of the sensor. Nonlinearity is deviation of a sensor's transfer function (TF) from a straight line transfer function. This is defined by the amount the output differs from ideal TF behavior over the full range of the sensor, which is denoted as the percentage of the full range.

- Full-Scale Output: It is the difference between the output for maximum input and the output for minimum input. Since the range of the output signal is always limited , the output signal will reach a minimum or maximum value, when the measured property exceeds the limits.

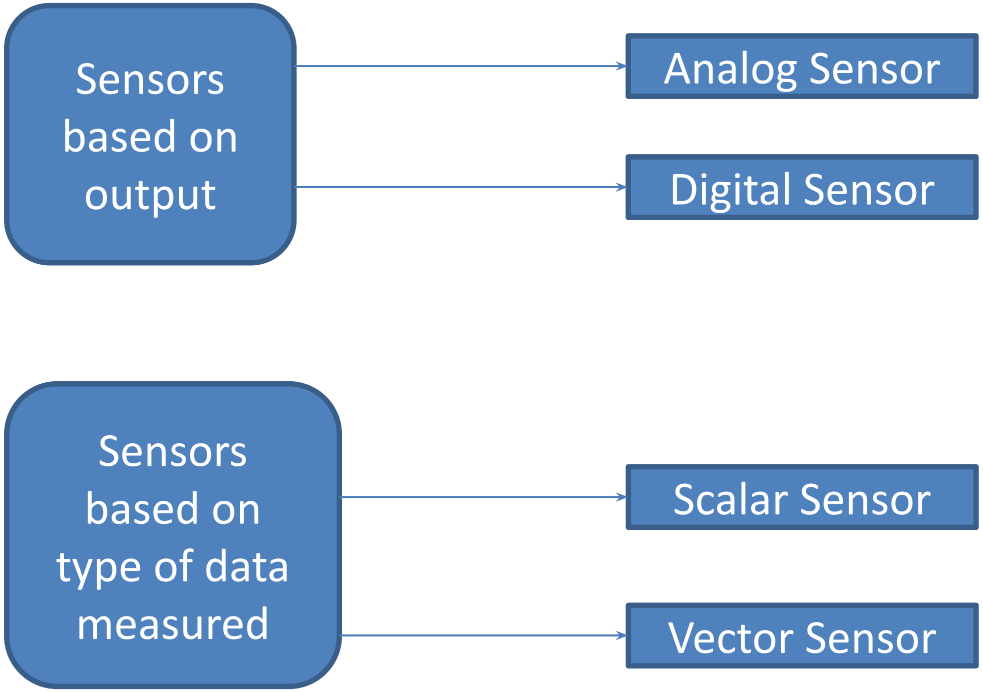
- Accuracy: It defines how close the output is to the real value. It defines the maximum error the sensor may produce.

- Precision: It the ability to produce same output when repeatedly measured for same input

- Calibration: To make meaningful measurement, it is necessary to tune the output of the sensor with accurately known input.

Classification Of Sensors

Sensors can be classified into two broad categories based on output and based on type of data measured.



Analog Sensor:

- Analog Sensors produce a continuous output signal or voltage which is generally proportional to the quantity being measured. Physical quantities such as Temperature, Speed, Pressure, displacement, Strain etc. are all analog quantities as they tend to be continuous in nature.

- For example- The temperature of a liquid can be measured using a thermometer which continuously responds to temperature changes as the liquid is heated up or cooled down.

Digital Sensor:

- **Digital Sensors** produce discrete digital output signals or voltages that are a digital representation of the quantity being measured.
- Digital sensors produce a binary output signal in the form of a logic “1” or a logic “0”, (“ON” or “OFF”).
- Digital signal only produces discrete (non-continuous) values, which may be output as a single “bit” (serial transmission), or by combining the bits to produce a single “byte” output (parallel transmission).
- These are mainly used in waste water management and industrial processes.

Scalar Sensor:

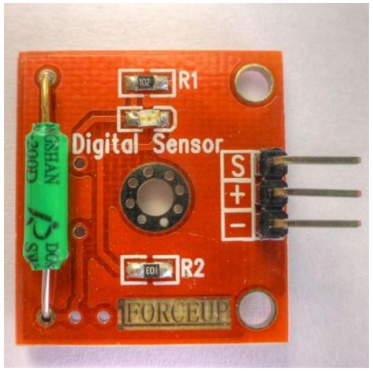
- **Scalar Sensors** produce output signal or voltage which is generally proportional to the magnitude of the quantity being measured.
- Physical quantities such as temperature, colour, pressure, strain, etc. are all scalar quantities as only their magnitude is sufficient to convey an information.
- For example, the temperature of a room can be measured using a thermometer or thermocouple, which responds to temperature changes irrespective of the orientation of the sensor or its direction.

Vector or Multimedia Sensor:

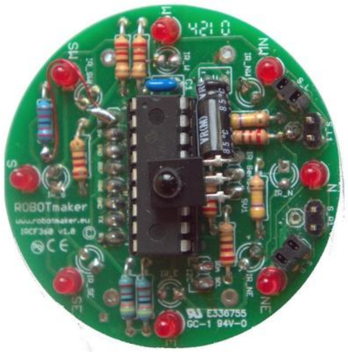
- **Vector Sensors** produce output signal or voltage which is generally proportional to the magnitude, direction, as well as the orientation of the quantity being measured.
- Physical quantities such as sound, image, velocity, acceleration, orientation, etc. are all vector quantities, as only their magnitude is not sufficient to convey the complete information.
- For example, the acceleration of a body can be measured using an accelerometer, which gives the components of acceleration of the body with respect to the x, y, z coordinate axes.

Sensor Types

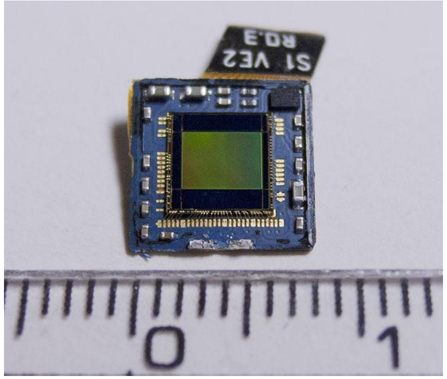
Light	• Light Dependent resistor • Photo-diode
Temperature	• Thermocouple • Thermistor
Force	• Strain gauge • Pressure switch
Position	• Potentiometer, Encoders •Opto-coupler
Speed	• Reflective/ Opto-coupler • Doppler effect sensor
Sound	• Carbon Microphone • Piezoelectric Crystal
Chemical	• Liquid Chemical sensor • Gaseous chemical sensor



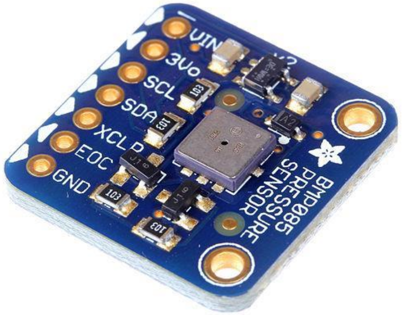
Tilt Sensor



Infrared
Motion Sensor



Camera Sensor



Pressure Sensor

Ultrasonic
Distance Sensor

Analog
Temperature
Sensor