INTERNET OF THINGS (IoT)

UNIT – I INTRODUCTION

Introduction to Internet of Things:

The 'Thing' in IoT can be any device with any kind of built-in-sensors with the ability to collect and transfer data over a network without manual intervention. The embedded technology in the object helps them to interact with internal states and the external environment, which in turn helps in decisions making process.

IoT is a concept that connects all the devices to the internet and let them communicate with each other over the internet. IoT is a giant network of connected devices – all of which gather and share data about how they are used and the environments in which they are operated.

By doing so, each of your devices will be learning from the experience of other devices, as humans do. IoT is trying to expand the interdependence in human- i.e *interact*, *contribute* and c*ollaborate* to things. I know this sounds a bit complicated, let's understand this with an example.

A developer submits the application with a document containing the standards, logic, errors & exceptions handled by him to the tester. Again, if there are any issues Tester communicates it back to the Developer. It takes multiple iterations & in this manner a smart application is created.

Similarly, a room temperature sensor gathers the data and sends it across the network, which is then used by multiple device sensors to adjust their temperatures accordingly. For example, refrigerator's sensor can gather the data regarding the outside temperature and accordingly adjust the refrigerator's temperature. Similarly, your air conditioners can also adjust its temperature accordingly. This is how devices can interact, contribute & collaborate.

Definition and Characteristics of IoT:

IoT definition: a network of connected devices with 1) unique identifiers in the form of an IP address which 2) have embedded technologies or are equipped with technologies that enable them to sense, gather data and communicate about the environment in which they reside and/or themselves.

The Internet of Things (IoT) is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction."

Definition: - The internet of things (IoT) is a computing concept that describes the idea of everyday physical objects being connected to the internet and being able to identify themselves to other devices. It has dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual "things" have identities, physical attributes and virtual network and use intelligent interfaces.

- Dynamic and Self-Adapting:- IoT devices and system may have the capability to change dynamically depending upon the system and operating conditions or sensed environment.
  For example, the surveillance cameras can change their modes based on day or night.

- Self-configuring:- IoT devices have self-configuring capability which allows large number of devices to work together to work provide certain functionality they can change their networking and update the software automatically.
- Interoperable Communication Protocol: - IoT devices can communicate with number of interoperable (communicate with other devices without special effort) communication protocols.
  Unique ID: - IoT devices have a unique identity differentiated with unique IP address.
- Integrated into Information Network: - IoT devices are integrated into the information network that allows them to communicate and exchange data with other devices and system.

There are 7 crucial IoT characteristics:

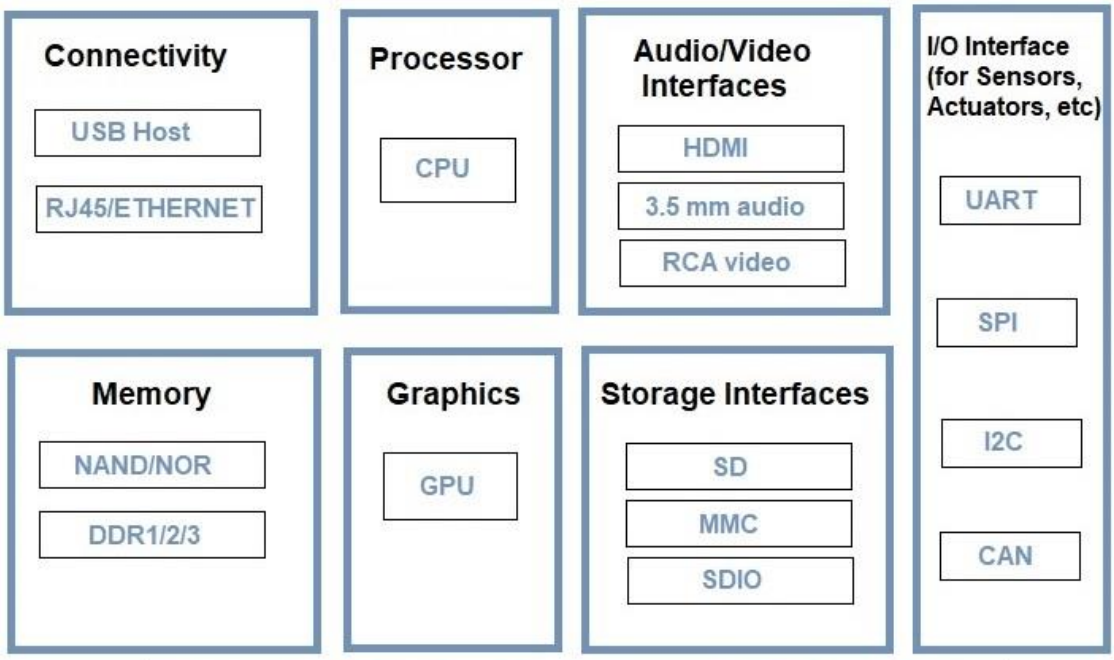
1. **Connectivity.** This doesn't need too much further explanation. With everything going on in IoT devices and hardware, with sensors and other electronics and connected hardware and control systems there needs to be a connection between various levels.
2. **Things**. Anything that can be tagged or connected as such as it's designed to be connected. From sensors and household appliances to tagged livestock. Devices can contain sensors or sensing materials can be attached to devices and items.
3. **Data**. Data is the glue of the Internet of Things, the first step towards action and intelligence.
4. **Communication**. Devices get connected so they can communicate data and this data can be analyzed. Communication can occur over short distances or over a long range to very long range. Examples: Wi-Fi, LPWA network technologies such as LoRa or NB-IoT.
5. **Intelligence**. The aspect of intelligence as in the sensing capabilities in IoT devices and the intelligence gathered from big data analytics (also artificial intelligence).
6. **Action**. The consequence of intelligence. This can be manual action, action based upon debates regarding phenomena (for instance in smart factory decisions) and automation, often the most important piece.
7. **Ecosystem**. The place of the Internet of Things from a perspective of other technologies, communities, goals and the picture in which the Internet of Things fits. The Internet of Everything dimension, the platform dimension and the need for solid partnerships.

PHYSICAL DESIGN OF IoT

Things in IoT

- The word "Things" refers to IoT devices which have unique identities and can perform remote sensing, actuating and monitoring capabilities. These devices can exchange and communicate with eachother.
- The IoT devices consists of several interfaces for connection to other devices both wired and wireless which includes

1. I/O interfaces for sensors
2. Interfaces for internet connectivity
3. Memory and storage
4. Audio and video interface



Generic Block Diagram of IoT Devices

Things are is main part of IoT Application. IoT Devices can be various type, Sensing Devices, Smart Watches, Smart Electronics appliances, Wearable Sensors, Automobiles, and industrial machines. These devices generate data in some forms or the other which when processed by data analytics systems leads to useful information to guide further actions locally or remotely.

For example, Temperature data generated by a Temperature Sensor in Home or other place, when processed can help in determining temperature and take action according to users. Above picture, shows a generic block diagram of IoT device. It may consist of several interfaces for connections to other devices. IoT Device has I/O interface for Sensors, Similarly for Internet connectivity, Storage and Audio/Video. IoT Device collect data from on-board or attached Sensors and Sensed data communicated either to other device or Cloud based sever. Today many cloud servers available for

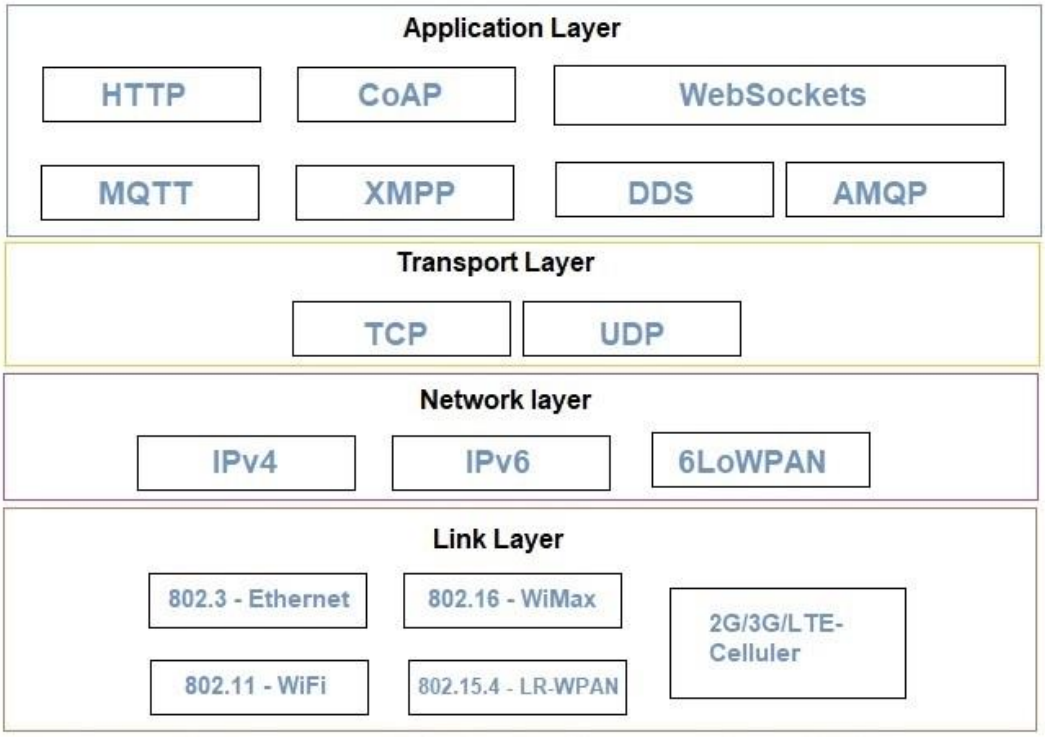
especially IoT System. These Platfrom known as IoT Platform. Actually these cloud especially design for IoT purpose. So here we can analysis and processed data easily.

Some of the abbreviations are :

- **SDIO: SECURE DIGITAL INPUT OUTPUT**
- **SD- SECURE DIGITAL**
- **GPU- GRAPHICS PROCESSING UNIT**
- **DDR STANDS FOR DOUBLE DATA RA**
- **HDMI -HIGH-DEFINITION MULTIMEDIA INTERFACE**
- **RADIO CORPORATION OF AMERICA - RCA CONNECTOR**
-
- **I^2C- SERIAL COMMUNICATION NETWORK**
- **CAN: A CONTROLLER AREA NETWORK**
- **UART :UNIVERSAL ASYNCRONOUS TRASMITTER & RECEIVER**
- **SPI : SERIAL PHERIPHERAL INTERFACE**

IoT Protocols

IoT protcols help to establish Communication between IoT Device (Node Device) and Cloud based Server over the Internet. It help to sent commands to IoT Device and received data from an IoT device over the Internet. An image is given below. By this image you can understand which protocols used.



Link Layer

Link layer protocols determine how data is physically sent over the network's physical layer or medium (Coxial calbe or other or radio wave). This Layer determines how the packets are coded and signaled by the hardware device over the medium to which the host is attached (eg. coxial cable).

Here we explain some Link Layer Protocols:

802.3 – Ethernet : Ethernet is a set of technologies and protocols that are used primarily in LANs. It was first standardized in 1980s by IEEE 802.3 standard. IEEE 802.3 defines the physical layer and the medium access control (MAC) sub-layer of the data link layer for wired Ethernet networks. Ethernet is classified into two categories: classic Ethernet and switched Ethernet.

802.11 – WiFi : IEEE 802.11 is part of the IEEE 802 set of LAN protocols, and specifies the set of media access control (MAC) and physical layer (PHY) protocols for implementing wireless local area network (WLAN) Wi-Fi computer communication in various frequencies, including but not limited to 2.4 GHz, 5 GHz, and 60 GHz frequency bands.

802.16 – Wi-Max : The standard for WiMAX technology is a standard for Wireless Metropolitan Area Networks (WMANs) that has been developed by working group number 16 of IEEE 802, specializing in point-to-multipoint broadband wireless access.

Following are the important differences between Wifi and WiMax.

Sr. No.	Key	Wifi	WiMax
1	Definition	Wifi stands for Wireless Fidelity.	WiMax stands for Wireless Inter-operability for Microwave Access.
2	Usage	WiFi uses Radio waves to create wireless high-speed internet and network connections. A wireless adapter is needed to create hotspots.	WiMax uses spectrum to deliver connection to network and handle a larger inter-operable network.
3	IEEE	Wifi is defined under IEEE 802.11x standards where x defines various WiFi versions.	WiMax is defined under IEEE 802.16y standards where y defines various WiMax versions.
4	Usage	Wifi is used in LAN applications.	WiMax is used in MAN applications.
5	QoS	Wifi does not gurrantee Quality of Service, QoS.	WiMax guarantees Quality of Service, QoS.
6	Network Range	Wifi network ranges at max 100 meters.	WiMax network ranges to max 90 kms.
7	Transmission speed	Wifi transmission speed can be upto 54 mbps.	WiMax transmission speed can be upto 70 mbps.

802.15.4 -LR-WPAN : A collection of standards for Low-rate wireless personal area network. The IEEE's 802.15.4 standard defines the MAC and PHY layer used by, but not limited to, networking specifications such as Zigbee®, 6LoWPAN, Thread, WiSUN and MiWi™ protocols. The standards provide low-cost and low-speed communication for power constrained devices.

2G/3G/4G- Mobile Communication : These are different types of telecommunication generations. IoT devices are based on these standards can communicate over the celluer networks.

Network Layer

Responsible for sending of IP datagrams from the source network to the destination network. Network layer performs the host addressing and packet routing. We used IPv4 and IPv6 for Host identification. IPv4 and IPv6 are hierarchical IP addrssing schemes.

IPv4 :
An **Internet Protocol address** (**IP address**) is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication. An IP address serves two main functions: host or network interface identification and location addressing. Internet Protocol version 4 (IPv4) defines an IP address as a 32-bit number. However, because of the growth of the Internet and the depletion of available IPv4 addresses, a new version of IP (IPv6), using 128 bits for the IP address, was standardized in 1998. IPv6 deployment has been ongoing since the mid-2000s.

IPv6 : Internet Protocol version 6 (**IPv6**) is successor of IPv4. IPv6 was developed by the Internet Engineering Task Force (IETF) to deal with the long-anticipated problem of IPv4 address exhaustion. In December 1998, IPv6 became a Draft Standard for the IETF, who subsequently ratified it as an Internet Standard on 14 July 2017. IPv6 uses a 128-bit address, theoretically allowing 2^{128}, or approximately 3.4×10^{38} addresses.

6LoWPAN : It is an acronym of *IPv6 over Low-Power Wireless Personal Area Networks*. 6LoWPAN is the name of a concluded working group in the Internet area of the IETF. This protocol allows for the smallest devices with limited processing ability to transmit information wirelessly using an internet protocol. 6LoWPAN can communicate with 802.15.4 devices as well as other types of devices on an IP network link like WiFi.

Transport Layer

This layer provides functions such as error control, segmentation, flow control and congestion control. So this layer protocols provide end-to-end message transfer capability independent of the underlying network.

TCP : TCP (Transmission Control Protocol) is a standard that defines how to establish and maintain a network conversation through which application programs can exchange data. TCP works with the Internet Protocol (IP), which defines how computers send packets of data to each other. Together, TCP and IP are the basic rules defining the Internet. The Internet Engineering Task Force (IETF) defines TCP in the Request for Comment (RFC) standards document number 793.

UDP : User Datagram Protocol (UDP) is a Transport Layer protocol. UDP is a part of Internet Protocol suite, referred as UDP/IP suite. Unlike TCP, it is unreliable and connectionless protocol. So, there is no need to establish connection prior to data transfer.

Application Layer

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

HTTP : *Hypertext Transfer Protocol (HTTP)* is an application-layer protocol for transmitting hypermedia documents, such as HTML. It was designed for communication between web browsers and web servers, but it can also be used for other purposes. HTTP follows a classical client-server model, with a client opening a connection to make a request, then waiting until it receives a response. HTTP is a stateless protocol, meaning that the server does not keep any data (state) between two requests.

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. This is the foundation for data communication for

the World Wide Web (i.e. internet) since 1990. HTTP is a generic and stateless protocol which can be used for other purposes as well using extensions of its request methods, error codes, and headers.

Basically, HTTP is a TCP/IP based communication protocol, that is used to deliver data (HTML files, image files, query results, etc.) on the World Wide Web. The default port is TCP 80, but other ports can be used as well. It provides a standardized way for computers to communicate with each other. HTTP specification specifies how clients' request data will be constructed and sent to the server, and how the servers respond to these requests.

Basic Features

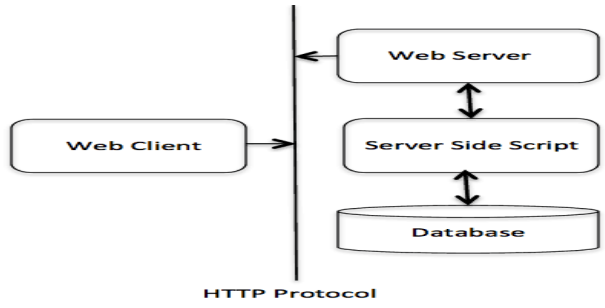
There are three basic features that make HTTP a simple but powerful protocol:

- **HTTP is connectionless:** The HTTP client, i.e., a browser initiates an HTTP request and after a request is made, the client waits for the response. The server processes the request and sends a response back after which client disconnect the connection. So client and server knows about each other during current request and response only. Further requests are made on new connection like client and server are new to each other.
- **HTTP is media independent:** It means, any type of data can be sent by HTTP as long as both the client and the server know how to handle the data content. It is required for the client as well as the server to specify the content type using appropriate MIME-type.
- **HTTP is stateless:** As mentioned above, HTTP is connectionless and it is a direct result of HTTP being a stateless protocol. The server and client are aware of each other only during a current request. Afterwards, both of them forget about each other. Due to this nature of the protocol, neither the client nor the browser can retain information between different requests across the web pages.

HTTP/1.0 uses a new connection for each request/response exchange, where as HTTP/1.1 connection may be used for one or more request/response exchanges

Basic Architecture

The following diagram shows a very basic architecture of a web application and depicts where HTTP sits:



The HTTP protocol is a request/response protocol based on the client/server based architecture where web browsers, robots and search engines, etc. act like HTTP clients, and the Web server acts as a server.

Client

The HTTP client sends a request to the server in the form of a request method, URI, and protocol version, followed by a MIME-like message containing request modifiers, client information, and possible body content over a TCP/IP connection.

Server

The HTTP server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME-like message containing server information, entity meta information, and possible entity-body content.

CoAP : CoAP-Constrained Application Protocol is a specialized Internet Application Protocol for constrained devices, as defined in RFC 7252. It enables devices to communicate over the Internet. The protocol is especially targeted for constrained hardware such as 8-bits microcontrollers, low power sensors and similar devices that can't run on HTTP or TLS.
Constrained Application Protocol is specifically designed for constrained (limited) Hardware. The hardware that doesn't support HTTP or TCP/IP can use CoAP Protocol. So, basically the designers of this protocol taking inspiration by the HTTP had designed the CoAP protocol using UDP and IP protocol. It is a lightweight protocol that needs low power IOT application like for communication between battery powered IOT devices. Like HTTP, it also follows client-server model. The clients can GET, PUT, DELETE or POST informational resources over the network.

CoAP makes use of two message types – requests and responses. The messages contain a base header followed by message body whose length is specified by the datagram. The messages or data packets are small in size, so that they can be communicated among constraint devices without data losses. The messages can be confirmable or non-confirmable. For confirmable messages, the client need to respond with an acknowledgement after receiving the data packet. The request messages can contain query strings to implement additional functionalities like search, sort or paging.

WebSocket : The WebSocket Protocol enables two-way communication between a client running untrusted code in a controlled environment to a remote host that has opted-in to communications from that code. The security model used for this is the origin-based security model commonly used by web browsers.

This protocol defines a full duplex communication from the ground up. Web sockets take a step forward in bringing desktop rich functionalities to the web browsers. It represents an evolution, which was awaited for a long time in client/server web technology.

The main features of web sockets are as follows −

- Web socket protocol is being standardized, which means real time communication between web servers and clients is possible with the help of this protocol.
- Web sockets are transforming to cross platform standard for real time communication between a client and the server.
- This standard enables new kind of the applications. Businesses for real time web application can speed up with the help of this technology.
- The biggest advantage of Web Socket is it provides a two-way communication (full duplex) over a single TCP connection.

MQTT: MQTT was developed by Andy Stanford-Clark (IBM) and Arlen Nipper (Eurotech; now Cirrus Link) in 1999 for the monitoring of an oil pipeline through the desert. The goals were to have a protocol, which is bandwidth-efficient and uses little battery power**,** because the devices were connected via satellite link and this was extremely expensive at that time. The protocol uses a publish/subscribe architecture in contrast to HTTP with its request/response paradigm. Publish/Subscribe is event-driven and enables messages to be pushed to clients. The central communication point is the MQTT broker, it is in charge of dispatching all messages between the senders and the rightful receivers. Each client that publishes a message to the broker, includes a topic into the message. **The topic is the routing information for the broker.** Each client that wants to

receive messages subscribes to a certain topic and the broker delivers all messages with the matching topic to the client. Therefore the clients don't have to know each other, they only communicate over the topic. This architecture enables highly scalable solutions without dependencies between the data producers and the data consumers.
Message Queuing Telemetry Transport is a lightweight messaging protocol. It uses publish-subscribe communication way and that's why it is used for M2M (machine to machine) communication. It is based on TCP-IP protocol and is designed to operate in limited bandwidth. In the protocol terminology, the limited network bandwidth is referred as 'small code footprint'. However, the exact meaning of limited network bandwidth is not clear in the specification.

This protocol has been specially designed for sensor networks and wireless sensor networks. MQTT allows devices to send or publish data information on a given topic to a server. There is a MQTT broker (Broker- Mosquitto) in between publisher and subscriber. The broker then transfers the information to the clients that are previously subscribed.

The sensors are interfaced with a broker which is an IOT device or server that reads and publish sensor data. The other devices that subscribe for and request sensor data are called clients. The sensors themselves are referred as publishers in the network. The client can be a laptop, smart phone, tablet or other mobile device. The client devices need to subscribe with the broker in the network to receive sensor data. For receiving data, the subscribed client devices have to establish connection with the broker and request data. The broker takes data from the publisher (wireless sensors) and sends it to the client requesting for it. Even if the connection with the client device is broken after the request has been made, the broker saves the data in a cache so that when the client device reconnects with the broker, it could receive the requested sensor data. Similarly, if the connection between the publisher and the broker is broken after the request has been made, the broker forwards appropriate instructions sent by the publisher, so that client device can reconnect and receive requested data.
An MQTT session can be divided into four stages –

1) Connection
2) Authentication
3) Communication
4) Termination
[Connection and Authentication – In this stage, the client (like a mobile device or laptop) initiates a TCP-IP connection with the broker (server) using a standard port or a port defined by the network operator. The standard ports are generally 1883 for non-encrypted communication and 8883 for encrypted communication through SSL or TLS. In encrypted communication, the server sends server certificate to authenticate itself by the client and client may also send a certificate to the server to authenticate itself. Though even in encrypted communication, the client authentication is not part of the specification and also not common. Still, usually, the client authentication is done by passing a username and password by the client to the broker (server).

Communication – The sensor data is communicated to client using small code footprints. Though the term 'small code footprint' is not exactly specified in the protocol, it usually contains 2-byte long header, an optional variable length header, sensor data as message payload limited to 250 Mb in size and Quality of Service Level Indication. There can be three Quality of Service (QoS) levels – Unacknowledged Service (QoS Level 0), Acknowledged Service (QoS Level 1) and Assured Service (QoS Level 2). Each exceeding QoS level requires more network bandwidth and tolerance to latency.

There can be four types of operations that can be performed by a client – publish, subscribe, unsubscribe and ping. The client can subscribe for specific topics in the network, like a mobile device might subscribe with a broker to read current temperature of a city. For subscribing, the client needs to send a SUBSCRIBE/SUBACK packet pair to the broker. Similarly, for unsubscribing (from a topic), the client needs to send a UBSUBSCRIBE/UNSUBACK packet pair to the server. The ping operation can be performed by a client device to check the live connection with the broker. In the publish operation, the data is communicated between the broker and the client on a specific topic.

4) Termination – The publisher or client can terminate the connection by sending a DISCONNECT message to the broker, after which, the TCP-IP connection is closed. If the connection is broken suddenly (due to limited network bandwidth) without termination request, the broker sends the cached messages from the publisher to the client.]

XMPP : Extensible Messaging and Presence Protocol (**XMPP**) is a communication protocol for message-oriented middleware based on XML (Extensible Markup Language). It enables the near-real-time exchange of structured yet extensible data between any two or more network entities.
Extensible Messaging and Presence Protocol (XMPP) is an XML based messaging protocol. XML is a mark up language for encoding documents which are both human readable as well as machine readable. XML evolved to extend HTML and allow addition of custom tags and elements to the web. As an extension of HTML, it allows structuring of data along with extensionality. Traditionally, XMPP has been used for real time communication like instant messaging, presence, multi-party chat, voice and video calls, collaboration, content syndication etc.

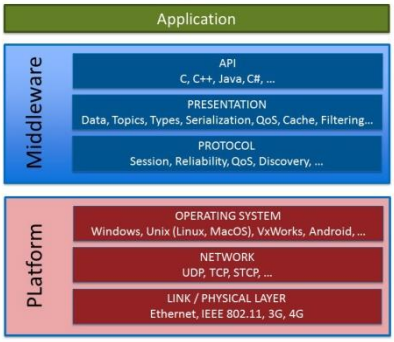
The use of XMPP for IOT allows real-time and scalable networking between devices or things. The things (devices) have one or more nodes and each node has several (informational) fields. Each field contains a readable and writable value. The nodes need to friend each other by sending and accepting friend requests. Once the friend request from one node is accepted by the other, it can receive updates from the other node. If other node also need to get updates from the first node, it also needs to send a friend request and need confirmation. If both nodes become mutually friend over the network, it is called dual subscription, otherwise, it is called single sided subscription. The data is communicated between the nodes on a one to one basis where a node can read or write field values in the other node.

All of the existing XMPP servers, clients, and programming libraries support the key features of an IM system, such as one-to-one and multi-party messaging, presence subscriptions and notifications, and contact lists. This wealth of code enables developers to easily build new applications in a secure and scalable way.

Ex: Whats app

DDS : The Data Distribution Service (DDS) is a middleware protocol and API standard for data-centric connectivity from the Object Management Group (OMG). It integrates the components of a system together, providing low-latency data connectivity, extreme reliability, and a scalable architecture that business and mission-critical Internet of Things (IoT) applications need.
Distribution Service is a M2M application layer protocol for real-time systems. Based on a publish – subscribe pattern like MQTT, the protocol architecture has nodes configured as publisher, subscriber or both. The protocol does not require any networking middleware, so the publishers can release information on specific topics (like a temperature sensor may release current temperature of a location) and the protocol itself manages to deliver it to the subscriber (like a mobile device showing current temperature of a location). The implementation of the protocol does not require any network programming as the protocol does not require verifying existence or location of the nodes and also does not need confirmation of the message delivery.

In a distributed system, middleware is the software layer that lies between the operating system and applications. It enables the various components of a system to more easily communicate and share data. It simplifies the development of distributed systems by letting software developers focus on the specific purpose of their applications rather than the mechanics of passing information between applications and systems.



There are many communications middleware standards and products. DDS is uniquely **data centric**, which is ideal for the Industrial Internet of Things. Most middleware works by sending information between applications and systems. Data centricity ensures that all messages include the contextual information an application needs to understand the data it receives.

AMQP : The AMQP – IoT protocols consist of a hard and fast of components that route and save messages within a broker carrier, with a set of policies for wiring the components together. The AMQP protocol enables patron programs to talk to the dealer and engage with the AMQP model.
Advanced Message Queue Protocol (AMQP) is also an open standard application layer protocol for message-oriented middleware. It is used for passing business messages between applications or organizations. It connects systems, feeds business processes with the information they need and reliably transmits the instructions that achieve their goals.

This is an interoperable and cross platform messaging standard. In the protocol architecture, the message along with a header is passed by the client to a broker or exchange. So, there is a single queue to which the message is passed by a producer. From the exchange or broker, the message can be passed on to one or many queues. The header contains information about each byte of the message. It also contains the routing information. The broker or exchange remains responsible to read headers and receive, route and deliver messages to the client applications. The communication in this protocol remains one to one between two nodes.

AMQP is the Internet Protocol for Business Messaging

The Advanced Message Queuing Protocol (AMQP) is an open standard for passing business messages between applications or organizations. It connects systems, feeds business processes with the information they need and reliably transmits onward the instructions that achieve their goals.

Key Capabilities

AMQP connects across:

- Organizations – applications in different organizations
- Technologies – applications on different platforms
- Time – systems don't need to be available simultaneously
- Space – reliably operate at a distance, or over poor networks

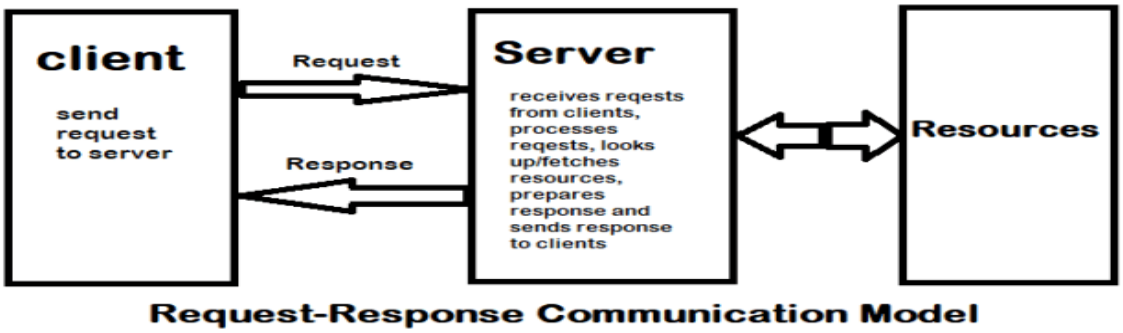
Key Features

AMQP was designed with the following main characteristics as goals:

- Security
- Reliability
- Interoperability
- Standard
- Open

➢ IoT Communication Model

- **Request-Response**



Request-Response Communication Model

In Request–Response communication model client sends request to the server and the server responds to the request. When the server receives the request it decides how to respond, fetches the data, retrieves resources, and prepares the response and sends to the client. R-R is a communication model where request and response pair is independent of each others.

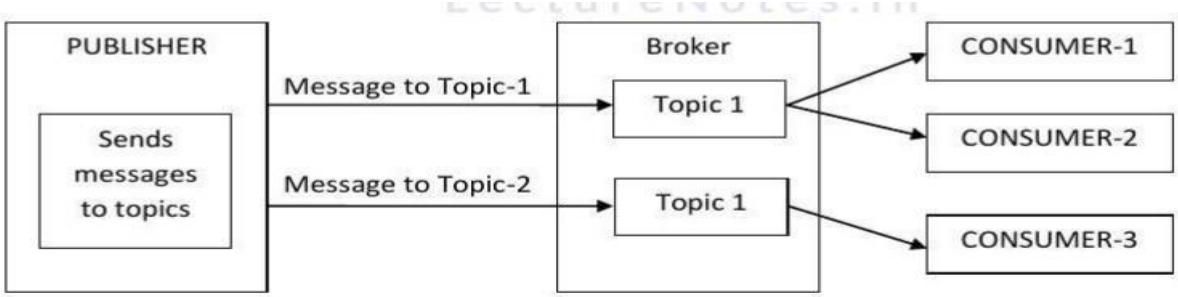
The **Request/Response** communication pattern is one of the most basic communication patterns. It allows a *client* to request information from a *server* in *real-time*. The words "client" and "server" are here used purely to illustrate the roles of the participants in the pattern, not to describe the hierarchy in the network. Most commonly, they are peers in the network.

XMPP provides an intrinsic method to implement a generic Request/Response mechanism, by use of the **iq** stanza built into XMPP. This allows one client to request information from another. What information is defined by the contents of the **iq** stanza. The receiver of the request is informed of whom originated the request, and the client is also informed from whence the response came.

- **Publish-Subscribe:**

This model involves publishers, brokers and consumers. Publishers are the sources of data. Publishers send the data to the topic which are managed by the broker. Publishers are not aware of the consumers.

Consumers subscribe to the topics which are managed by the broker. When broker receives the data from the Publisher, it sends to all the consumers.



Publish/Subscribe is another classic pattern where senders of messages, called publishers, do not program the messages to be sent directly to specific receivers, called subscribers. Messages are published without the knowledge of what or if any subscriber of that knowledge exist

Or

Pub/sub is shorthand for publish/subscribe messaging, an asynchronous communication method in which messages are exchanged between applications without knowing the identity of the sender or recipient.

Overview

Four core concepts make up the pub/sub model:

1. **Topic** – An intermediary channel that maintains a list of subscribers to relay messages to that are received from publishers
2. **Message** – Serialized messages sent to a topic by a publisher which has no knowledge of the subscribers
3. **Publisher** – The application that publishes a message to a topic
4. **Subscriber** – An application that registers itself with the desired topic in order to receive the appropriate messages

Advantages and disadvantages of pub/sub

As with all technology, using pub/sub messaging comes with advantages and disadvantages. The two primary advantages are loose coupling and scalability.

Loose coupling

Publishers are never aware of the existence of subscribers so that both systems can operate independently of each other. This methodology removes service dependencies that are present in traditional coupling. For example, a client generally cannot send a message to a server if the server process is not running. With pub/sub, the client is no longer concerned whether or not processes are running on the server.

Scalability

Pub/sub messaging can scale to volumes beyond the capability of a single traditional data center. This level of scalability is primarily due to parallel operations, message caching, tree-based routing, and multiple other features built into the pub/sub model.

Scalability does have a limit though. Increasing the number of nodes and messages also increases the chances of experiencing a load surge or slowdown. On top of that, the advantages of the pub/sub model can sometimes be overshadowed by the message delivery issues it experiences, such as:

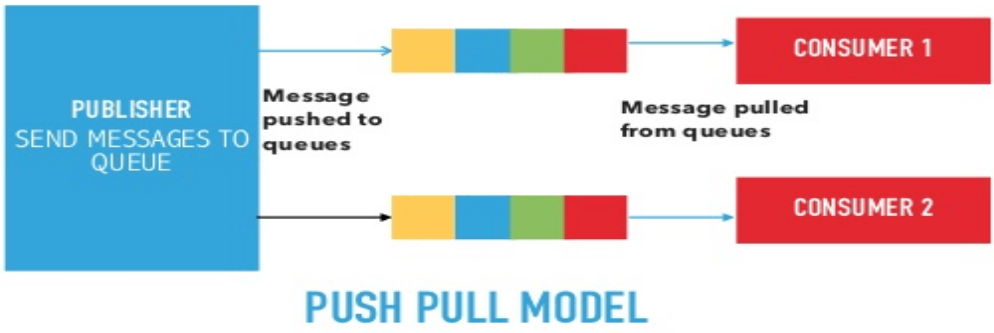
- A publisher may only deliver messages for a certain period of time regardless of whether the message was received or not.
- Since the publisher does not have a window into the subscriber it will always assume that the appropriate subscriber is listening. If the subscriber isn’t listening and misses an important message it can be disastrous for production systems.

How Pub/Sub Works

In the overview we covered how a publisher sends a message to a topic and how the topic forwards the message to the appropriate subscriber. From a topology point of view it is a simple process.

When it comes to coding the publish or the subscribe process the model can be a bit more confusing.

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



PUSH PULL MODEL

- **Exclusive Pair:**
  Exclusive pair is a bi-directional, fully duplex communication model that uses a persistent connection between the client and server, once the connection is established it remains open until the client sends a request to closer the connection. Client and server can send the message to each other after connection setup. In this model server is aware of all the open connection.

EXCLUSIVE PAIR COMMUNICATION MODEL

IOT Communication API's

The application program (or programming) interface, or API, is arguably what really ties together the connected "things" of the "internet of things." IoT APIs are the points of interaction between an IoT device and the internet and/or other elements within the network.
As API management company Axway puts it, "APIs are tightly linked with IoT because they allow you to securely expose connected devices to customers, go-to-market channels and other applications in your IT infrastructure."

Generally we used Two APIs For IoT Communication. These IoT Communication APIs are:

- REST-based Communication APIs
- WebSocket-based Communication APIs

REST-based Communication APIs

Representational state transfer (REST) is a set of architectural principles by which you can design Web services the Web APIs that focus on systems's resources and how resource states are addressed and transferred. REST APIs that follow the request response communication model, the rest architectural constraint apply to the components, connector and data elements, within a distributed hypermedia system.

The rest architectural constraint are as follows:

Client-server – The principle behind the client-server constraint is the separation of concerns. for example clients should not be concerned with the storage of data which is concern of the serve. Similarly the server should not be concerned about the user interface, which is concern of the clien. Separation allows client and server to be independently developed and updated.

Stateless – Each request from client to server must contain all the information necessary to understand the request, and cannot take advantage of any stored context on the server. The session state is kept entirely on the client.

Cache-able – Cache constraints requires that the data within a response to a request be implicitly or explicitly leveled as cache-able or non cache-able. If a response is cache-able, then a client cache is given the right to reuse that repsonse data for later, equivalent requests. caching can partially or completely eliminate some instructions and improve efficiency and scalability.

Layered system – layered system constraints, constrains the behavior of components such that each component cannot see beyond the immediate layer with they are interacting. For example, the client cannot tell whether it is connected directly to the end server or two an intermediaryalong the way. System scalability can be improved by allowing intermediaries to respond to requests instead of the end server, without the client having to do anything different.

Uniform interface – uniform interface constraints requires that the method of communication between client and server must be uniform. Resources are identified in the requests (by URIsin web based systems) and are themselves is separate from the representations of the resources data

returned to the client. When a client holds a representation of resources it has all the information required to update or delete the resource you (provided the client has required permissions). Each message includes enough information to describe how to process the message.

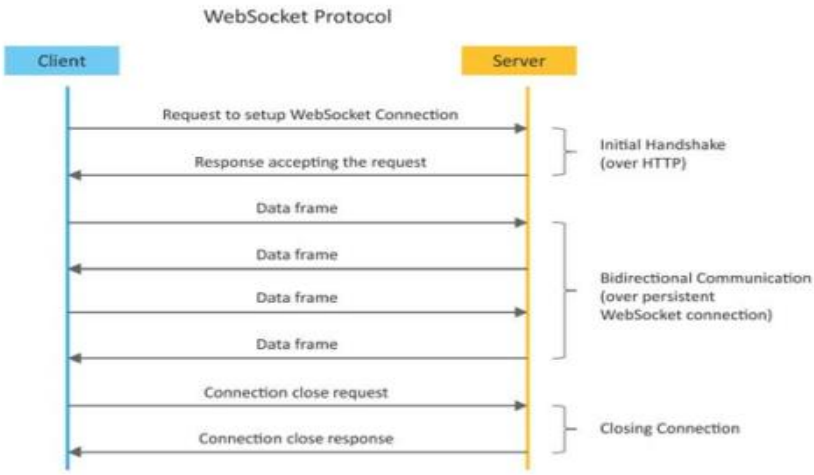
Code on demand – Servers can provide executable code or scripts for clients to execute in their context. this constraint is the only one that is optional.
A RESTful web service is a " Web API " implemented using HTTP and REST principles.
REST is most popular IoT Communication APIs.

HTTP methods

Uniform Resource Identifier (URI)	GET	PUT	PATCH	POST	DELETE
Collection, such as https://api.example.com/resources/	*List* the URIs and perhaps other details of the collection's members.	*Replace* the entire collection with another collection.	Not generally used	*Create* a new entry in the collection. The new entry's URI is assigned automatically and is usually returned by the operation.	*Delete* the entire collection.
Element, such as https://api.example.com/resources/item5	*Retrieve* a representation of the addressed member of the collection, expressed in an appropriate Internet media type.	*Replace* the addressed member of the collection, or if it does not exist, create it.	*Update* the addressed member of the collection.	Not generally used. Treat the addressed member as a collection in its own right and create a new entry within it.	*Delete* the addressed member of the collection.

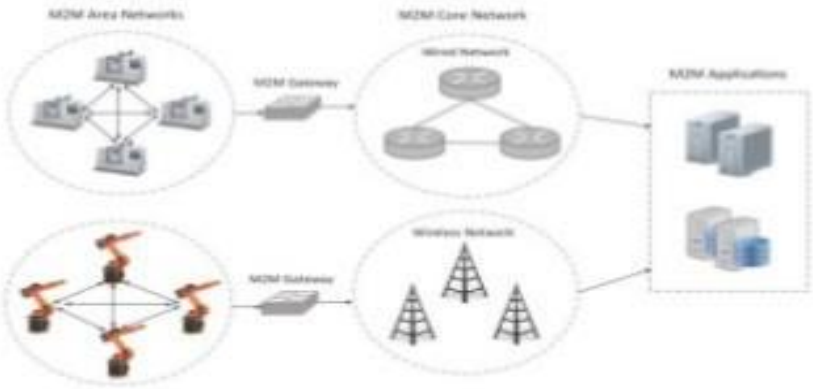
WebSocket based communication API

Websocket APIs allow bi-directional, full duplex communication between clients and servers. Websocket APIs follow the exclusive pair communication model. Unlike request-response model such as REST, the WebSocket APIs allow full duplex communication and do not require new coonection to be setup for each message to be sent. Websocket communication begins with a connection setup request sent by the client to the server. The request (called websocket handshake) is sent over HTTP and the server interprets it is an upgrade request. If the server supports websocket protocol, the server responds to the websocket handshake response. After the connection setup client and server can send data/mesages to each other in full duplex mode. Websocket API reduce the network traffic and letency as there is no overhead for connection setup and termination requests for each message. Websocket suitable for IoT applications that have low latency or high throughput requirements. So Web socket is most suitable IoT Communication APIs for IoT System.



Machine -to-Machine (M2M) communications:

Machine -to-Machine (M2M) refers to the communication or exchange of data between to two or more machines without human interfacing or interaction. The communication in M2M may be wired or wireless systems. The M2M uses a device such as sensor, RFID, meter, etc. to capture an 'events' like temperature, inventory level, etc., which is relayed through a network i.e., wireless, wired or hybrid to an application (software program), that translates the captured event into meaningful information. Unlike SCADA or other remote monitoring tools, M2M systems often use public networks and access methods -- for example, cellular or Ethernet -- to make it more cost-effective.



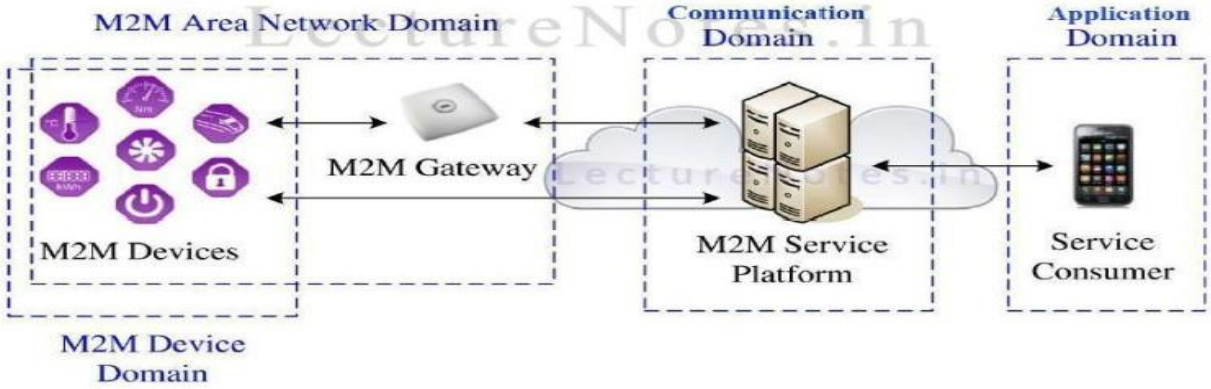
Machine-to-Machine (M2M)

- An M2M area network comprises of machines (or M2M nodes) which have embedded hardware modules for sensing, actuation and communication.
- Various communication protocols can be used for M2M local area networks such as ZigBee, Bluetooth, ModBus, M-Bus, Wirless M-Bus, Power Line Communication (PLC), 6LoWPAN, IEEE 802.15.4, etc.
- The communication network provides connectivity to remote M2M area networks.
- The communication network can use either wired or wireless networks (IP-based).
- While the M2M area networks use either proprietary or non-IP based communication protocols, the communication network uses IP-based networks.

M2M System Architecture

The main components of M2M system are:

1. M2M area networks
2. Communication networks
3. Application domains
4. M2M gateways.

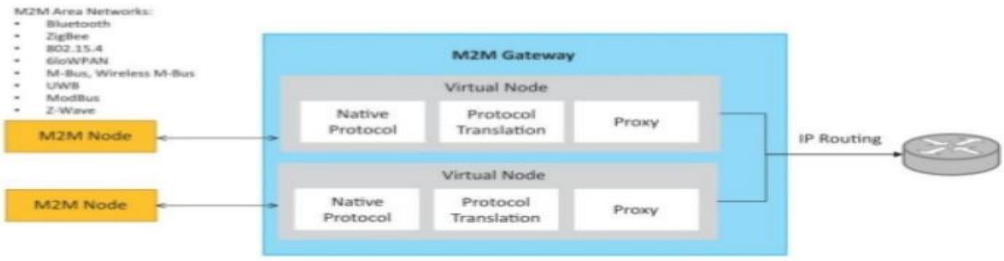


M2M area networks

- M2M network area consists of machines or M2M nodes which communicate with each other. The M2M nodes embedded with hardware modules such as sensors, actuators and communication devices.
- M2M uses communication protocol such as ZigBee, Bluetooth, Power line communication (PLC) etc. These communication protocols provide connectivity between M2M nodes within M2M area network.
- The M2M nodes communicate with in one network it can't communicate with external network node. To enable the communication between remote M2M area networks, M2M gateways are used.

M2M Gateways

- The Gateway module provides control and localization services for data collection. The gateways also double up in concentrating traffic to the operator's core. It supports Bluetooth, ZigBee, GPRS capabilities.
- M2M communication network serves as infrastructure for realizing communication between M2M gateway and M2M end user application or server. For this cellular network (GSM /CDMA), Wire line network and communication satellites may be used.



Communication networks

- The communication network provides the connectivity between M2M nodes and M2M applications.
- It uses wired or wireless network such as LAN, LTE, WiMAX, satellite communication etc.

Application domains

- It contains the middleware layer where data goes through various application services and is

used by the specific business-processing engines.

- M2M applications will be based on the infrastructural assets that are provided by the operator. Applications may either target at end users, such as user of a specific M2M solution, or at other application providers to offer more refined building blocks by which they can build more sophisticated M2M solutions and services.

IoT Enabling Technologies:

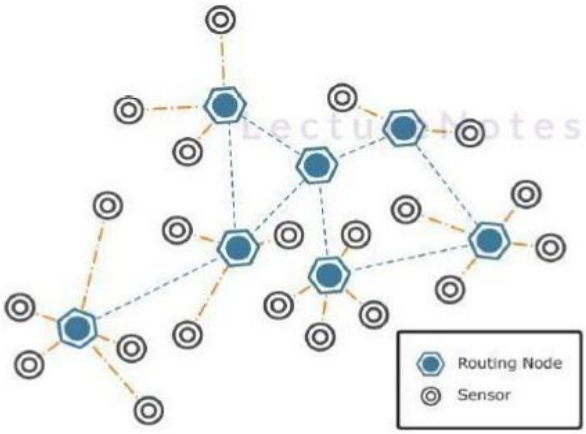
IoT is enabled by several technologies which includes WSN, cloud computing, big data analysis, embedded systems, web servers, mobile internet etc. Some the technologies which play the key role in IoT are:

➢ WIRELESS SENSOR NETWORK:

A Wireless Sensor Network (WSN) is a distributed network with large number of sensors which are used to monitor the environmental and physical conditions. A WSN consists of end nodes, routers and coordinator. End nodes have several sensors attached to them where the data is passed to coordinator with the help of routers. The coordinator also acts gateway that connects WSN to internet. WSN are enabled by wireless communication protocol such as 802.15.4. and ZigBee is one of the most popular wireless technologies.

The some of the examples where WSN is used in the IOT are:

1. Weather monitoring system uses WSN which collects data of temperature, humidity which are aggregated and analyzed.
2. Indoor air quality monitoring system use WSN.
3. Soil Moisture Monitoring system
4. Surveillance systems use WSN for collecting data.
5. Health monitoring system.



➢ <u>CLOUD COMPUTING:</u>

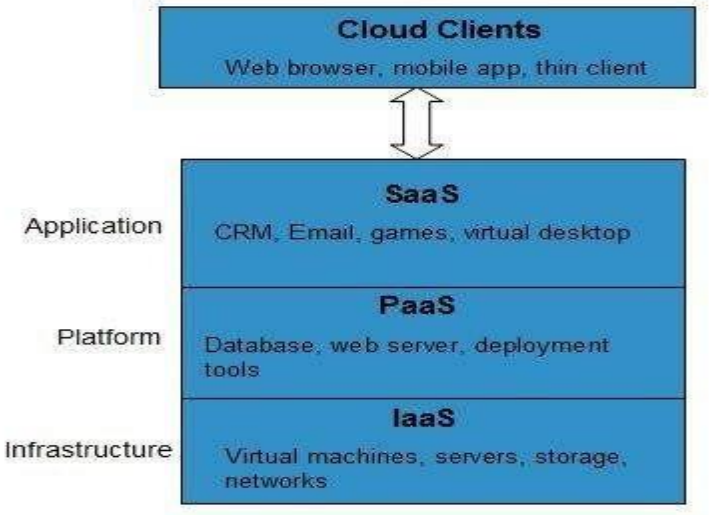
Cloud computing is the use of various services, such as software development platforms, servers, storage and software, over the internet, often referred to as the "cloud." Generally the cloud computing based on three characteristics such as:

1. The back-end of the application (especially hardware) is completely managed by a cloud vendor.
2. A user only pays for services used (memory, processing time and bandwidth, etc.).
3. Services are scalable.

The Cloud computing provides ability to “pay on demand” and scale quickly is largely a result of cloud computing vendors being able to pool resources that may be divided among multiple clients.

The cloud computing provides different services such as:

1. Infrastructure as a Service (IaaS)
2. Platform as a Service (PaaS)
3. Software as a Service (SaaS).



Infrastructure as a Service (IaaS)

Infrastructure as a service provides companies with computing resources including servers, networking, storage, and data center space on a pay-per-use basis.
The benefits of IaaS

- No need to invest in your own hardware
- Infrastructure scales on demand to support dynamic workloads
- Flexible, innovative services available on demand

Platform as a Service (PaaS)

Platform as a service provides a cloud-based environment with everything required to support the complete lifecycle of building and delivering web-based (cloud) applications — without the cost and complexity of buying and managing the underlying hardware, software, provisioning, and hosting.
The benefits of PaaS

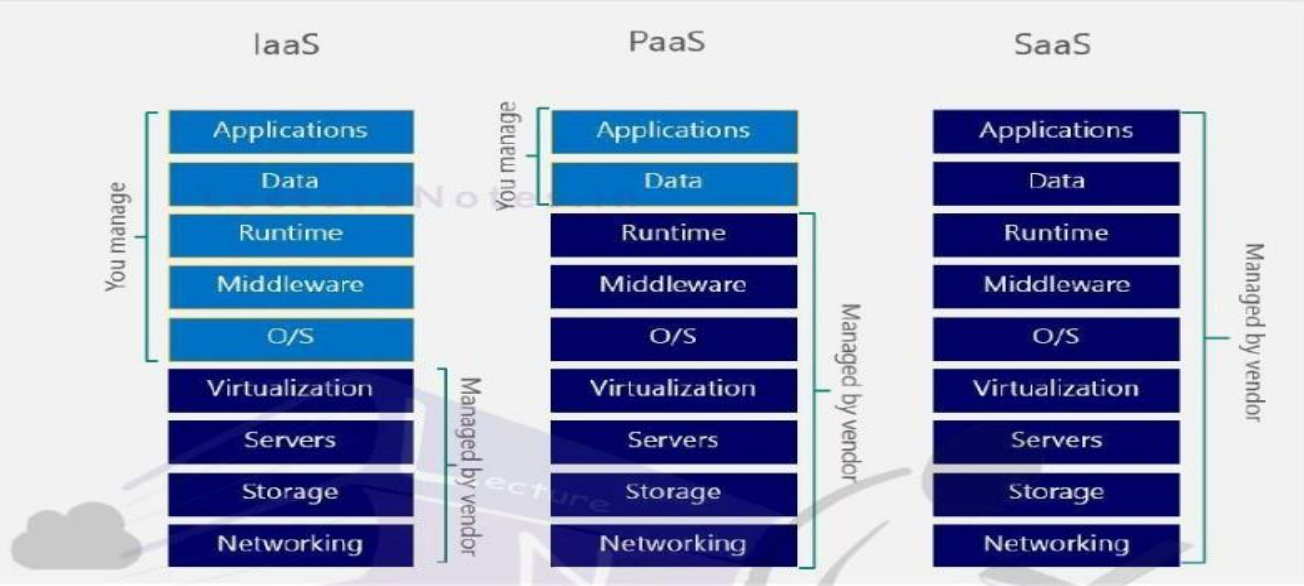
- Develop applications and get to market faster
- Deploy new web applications to the cloud in minutes
- Reduce complexity with middleware as a service

Software as a Service (SaaS)

Software as a service — run on distant computers "in the cloud" that are owned and operated by others and that connect to users' computers via the internet and, usually, a web browser.
The benefits of SaaS

- You can sign up and rapidly start using innovative business apps

- Apps and data are accessible from any connected computer
- No data is lost if your computer breaks, as data is in the cloud
- The service is able to dynamically scale to usage needs



Common Examples

- SaaSGoogle Apps, Dropbox, Salesforce, Cisco WebEx, Concur, GoToMeeting
- PaaSAWS Elastic Beanstalk, Windows Azure, Heroku, Force.com, Google App Engine, Apache Stratos, OpenShift
- IaaSDigitalOcean, Linode, Rackspace, Amazon Web Services (AWS), Cisco Metapod, Microsoft Azure, Google Compute Engine (GCE)SaaS: Software as a Service
- Software as a Service, also known as cloud application services

Benefits of Cloud Computing:

1. It doesn't require you to maintain or manage it(no need to have an IT expert).
2. Effectively infinite size, so no need to worry about running out of capacity.
3. You can access cloud based applications and services from anywhere

- **Big Data Analytics:**

Big data analytics refers to the strategy of analyzing large volumes of data, or big data. This big data is gathered from a wide variety of sources, including social networks, videos, digital images, sensors, and sales transaction records. The large data is difficult to store, manage, process and analyze the data using traditional databases and data processing tools. There are several steps which involves in analyzing big data are data cleansing, data managing, data processing and visualization.

The term **Big Data** refers to a huge volume of data that can not be stored processed by any traditional data storage or processing units. Big Data is generated at **a very large** scale and it is being used by many multinational companies to **process** and **analyse** in order to uncover insights and improve the business of many organisations.

Types of Big-Data

Big Data is generally categorized into three different varieties. They are as shown below:

- **Structured Data**
- **Semi-Structured Data**
- **Unstructured Data**

- **Structured Data** owns a dedicated data model, It also has a well-defined structure, it follows a consistent order and it is designed in such a way that it can be **easily accessed** and used by a person or a computer. Structured data is usually stored in well-defined columns and also Databases.

Example: Database Management Systems(**DBMS**)

- **Semi-Structured Data** can be considered as another form of Structured Data. It inherits a few properties of Structured Data, but the major part of this kind of data fails to have a definite structure and also, it does not obey the formal structure of data models such as an RDBMS.

Example:Comma Separated Values(**CSV**) File.

- **Unstructured Data** is completely a different type of which neither has a structure nor obeys to follow the formal structural rules of data models. It does not even have a consistent format and it found to be varying all the time. But, rarely it may have information related to data and time.

Example: Audio Files, Images etc

Characteristics of Big Data

Volume

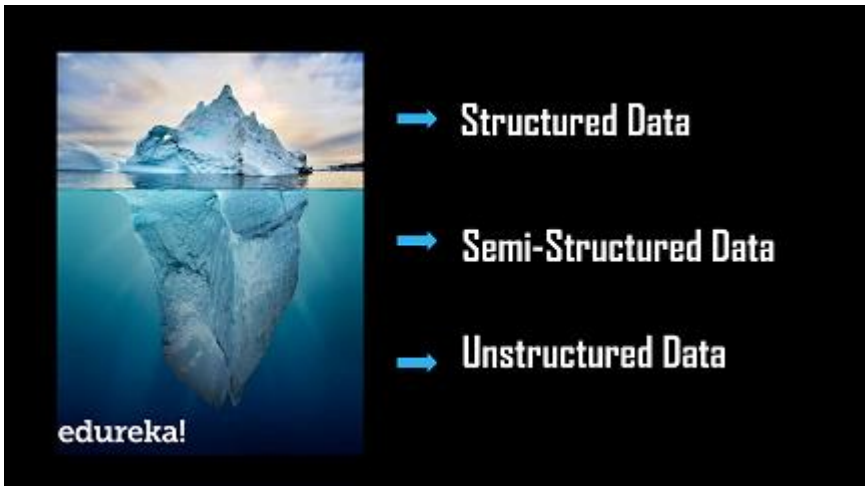
Volume refers to the unimaginable amounts of information generated every second from social media, cell phones, cars, credit cards, M2M sensors, images, video, and whatnot. We are currently using **distributed systems**, to store data in several locations and brought together by a software Framework like **Hadoop**.

Facebook alone can generate about **billion** messages, **4.5 billion** times that the "like" button is recorded, and over **350 million** new posts are uploaded **each day.** Such a huge amount of data can only be handled by Big Data Technologies.

Variety

As Discussed before, **Big Data** is generated in multiple varieties. Compared to the traditional data like phone numbers and addresses, the latest trend of data is in the form of photos, videos, and audios and many more, making about 80% of the data to be completely unstructured

Structured data is just the tip of the iceberg.



Veracity

Veracity basically means the degree of reliability that the data has to offer. Since a major part of the data is unstructured and irrelevant, Big Data needs to find an alternate way to filter them or to translate them out as the data is crucial in business developments

Value

Value is the major issue that we need to concentrate on. It is not just the amount of data that we store or process. It is actually the amount of valuable, reliable and trustworthy data that needs to be stored, processed, analyzed to find insights.

Velocity

Last but never least, Velocity plays a major role compared to the others, there is no point in investing so much to end up waiting for the data. So, the major aspect of Big Dat is to provide data on demand and at a faster pace.

Some examples of big data generated by IoT are:

1. Sensors data generated by weather monitoring stations.
2. Data generated by IoT systems for location and tracking of vehicles
3. Sensors embedded in industry and energy system.
4. Health and fitness data generated by IoT system such as fitness bands.

Communication Protocols:

- Backbone of IOT system
- Allows devices to exchange data over networks.
- Define data exchange formats
  - Data encoding
  - Addressing Schemes
  - Routing of packets from sources to destination
- Other Functions
  - Sequence control(ordering data packets)
  - Flow control(controlling transfer rate)
  - Retransmission of lost packets

Wireless communication protocols into the following 6 standards:

Satellite
WiFi

Radio Frequency (RF)

RFID

Bluetooth

NFC

1. Satellite

Satellite communications enable cell phone communication from a phone to the next antenna of about 10 to 15 miles. They are called GSM, GPRS, CDMA, GPRS, 2G / GSM, 3G, 4G / LTE, EDGE and others based on connectivity speed.

In Internet of Things language, this form of communication is mostly referred to as "M2M" (Machine-to-Machine) because it allows devices such as a phone to send and receive data through the cell network.

Pros and Cons of Satellite Communication

Pros:

- Stable connection
- Universal compatibility

Cons:

- No direct communication from smartphone to device (It has to go through satellite)
- High monthly cost
- High power consumption

Example of satellite connectivity would include utility meters that send data to a remote server, commercials updated on digital billboards, or cars via Internet connectivity.

Satellite is useful for communication that **utilize low data volumes**, mainly industrial purposes but in the changing near future where cost of satellite communication is gradually falling, **the use of satellite technology might become much more viable and interesting for consumers.**

2. WiFi

WiFi is a wireless local area network (WLAN) that utilizes the IEEE 802.11 standard through 2.4GhZ UHF and 5GhZ ISM frequencies. WiFi provides Internet access to devices that are within the range (about 66 feet from access point).

Pros and Cons of WiFi

Pros:

- Universal smartphone compatibility
- Affordable
- Well protected and controlled

Cons:

- Relatively high power usage
- Instability and inconsistency of WiFi

Example of WiFi connectivity would be Dropcam streaming live video via the local WiFi instead of streaming through a connected Ethernet LAN cable. WiFi is useful for many Internet of Things connections but **such connections typically connect to an external cloud-server** and **are not directly connected to the smartphone**. It is also not recommended for battery-powered devices due to its relatively high power consumption.

3. Radio Frequency (RF)

Radio frequency communications are probably the easiest form of communications between devices. Protocols like **ZigBee** or **ZWave** use a low-power RF radio embedded or retrofitted into electronic devices and systems.

Z-Wave's range is approximately 100 ft (30 m). The radio frequency band used is specific to its country. For example, Europe has a 868.42 MHz SRD Band, a 900 MHz ISM or 908.42 MHz band (United States), a 916 MHz in Israel, 919.82 MHz in Hong Kong, 921.42 MHz in the regions of Australia/New Zealand) and 865.2 Mhz in India.

ZigBee is based on the IEEE 802.15.4 standard. However, its low power consumption limits transmission distances to a range of 10 to 100 meters.

Pros and Cons of RF

Pros:

- Low energy and simplicity for its technology is not dependent on the new functionality of phones

Cons:

- Radio frequency technology is not used by smartphones and without a central hub to connect the RF devices to the internet, the devices cannot be connected

Example of radio frequency connectivity would be your typical television remote for it uses radio frequency, which enables you to switch channels remotely. Other examples include wireless light switches, electrical meters with in-home displays, traffic management systems and other consumer and industrial equipment that requires short-range low-rate wireless data transfer.

Radio frequency communication protocol is **useful for large deployments** such as hotels where high quantity of devices are required to be centrally and locally managed. However, in the near future, **the technology might become increasingly outdated and be replaced by Bluetooth mesh networks**.

4. RFID

Radio frequency identification (RFID) is the wireless use of electromagnetic fields to identify objects. Usually you would install an active reader, or reading tags that contain a stored information mostly authentication replies. Experts call that an Active Reader Passive Tag (ARPT) system. Short range RFID is about 10cm, but long range can go up to 200m. What many do not know is that Léon Theremin invented the RFID as an espionage tool for the Soviet Union in 1945.

An Active Reader Active Tag (ARAT) system uses active tags awoken with an interrogator signal from the active reader. Bands RFID runs on: 120–150 kHz (10cm), 3.56 MHz (10cm-1m), 433 MHz (1-100m), 865-868 MHz (Europe), 902-928 MHz (North America) (1-12m).

Pros and Cons of RFID

Pros:

- Does not require power
- Established and widely used technology

Cons:

- Highly insecure
- Ongoing cost per card
- Tags need to be present as identifier and be handed over before
- Not compatible with smartphones

Examples include animal identification, factory data collection, road tolls, and building access. RFID tag is also attached to an inventory such that its **production and manufacturing progress can be tracked through the assembly line**. As illustration, pharmaceuticals can be tracked through warehouses. We believe RFID technology will very soon be **replaced by near field communication (NFC) technology** in smartphone.

5. Bluetooth

Bluetooth is a wireless technology standard for exchanging data over short distances (using short-wavelength UHF radio waves in the ISM band from 2.4 to 2.485 GHz). If you look at the frequencies it is actually the same as WiFi such that these two technologies seem very similar. However they have different uses.The 3 different styles of Bluetooth technology that are commonly talked about are:

- **Bluetooth:** Remember the days where you associate Bluetooth as a battery drainer and black hole? Such Bluetooth are a heyday relic of a mobile past marked by bulky cell phone. Such Bluetooth technology are battery draining, insecure and are often complicated to pair.
- **BLE (Bluetooth 4.0, Bluetooth Low Energy):** Originally introduced by Nokia and presently used by all major operating systems such as iOS, Android, Windows Phone, Blackberry, OS X, Linux and Windows 8, BLE uses fast, low energy usage while maintaining the communication range.
- **iBeacon:** It is the trademark for a simplified communication technique based on Bluetooth technology that Apple uses. What it actually is: a Bluetooth 4.0 sender that transmits an ID called UUID, which is recognized by your iPhone. This simplifies the implementation effort many vendors would previously face. Moreover, even non-technically trained consumers can easily use iBeacons like Estimote.com or other alternatives. Although, different on a technical level, iBeacon technology can be compared to NFC on an abstract level.

Bluetooth exists in many products, such as telephones, tablets, media players, robotics systems. The technology is extremely useful when transferring information between two or more devices that are near each other in low-bandwidth situations. Bluetooth is commonly used to transfer sound data with telephones (i.e., with a Bluetooth headset) or byte data with hand-held computers (transferring files).Bluetooth protocols simplify the discovery and setup of services between devices. Bluetooth devices can advertise all of the services they provide. This makes using services easier, because relative to other communication protocols, it enables greater automations such as security, network address and permission configuration.

Pros & Cons of Bluetooth

Pros:

- Every smartphone has Bluetooth where the technology is continuously being upgraded and improved through new hardware
- Established and widely used technology

Cons:

- Hardware capabilities changes very fast and will need to be replaced
- Running on battery the lifetime of an iBeacon is between 1month to 2 years
- If people switch off Bluetooth, there are issues in usage.

Bluetooth technology mainly finds applications in the healthcare, fitness, beacons, security, and home entertainment industries.

Bluetooth technology is definitely the hottest technology right now but it is many times overrated or misunderstood in functionality. If the application goes beyond fun you will have to dig deep in configuration and different settings as different phones react differently to Bluetooth.

6. Near Field Communication (NFC)

Near-field communication uses electromagnetic induction between two loop antennas located within each other's near field, effectively forming an air-core transformer. It operates within the globally available and unlicensed radio frequency ISM band of 13.56 MHz on ISO/IEC 18000-3 air interface and at rates ranging from 106 kbit/s to 424 kbit/s. NFC involves an initiator and a target; the initiator actively generates an RF field that can power a passive target (an unpowered chip called a "tag"). This enables NFC targets to take very simple form factors such as tags, stickers, key fobs, or battery-less cards. NFC peer-to-peer communication is possible provided both devices are powered.

There are two modes:

- **Passive communication mode**: The initiator device provides a carrier field and the target device answers by modulating the existing field. In this mode, the target device may draw its operating power from the initiator-provided electromagnetic field, thus making the target device a transponder.
- **Active communication mode**: Both initiator and target device communicate by alternately generating their own fields. A device deactivates its RF field while it is waiting for data. In this mode, both devices typically have power supplies.

Pros & Cons of NFC

Pros:

- Offers a low-speed connection with extremely simple setup
- Can be used to bootstrap more capable wireless connections
- NFC has a short range and supports encryption where it may be more suitable than earlier, less private RFID systems

Cons:

- Short range might not be feasible in many situations for it is currently only available on new Android Phones and at Apple Pay on new iPhones

UNIT – II IOT SYSTEM MANAGEMENT

Software defined networks (SDN) :

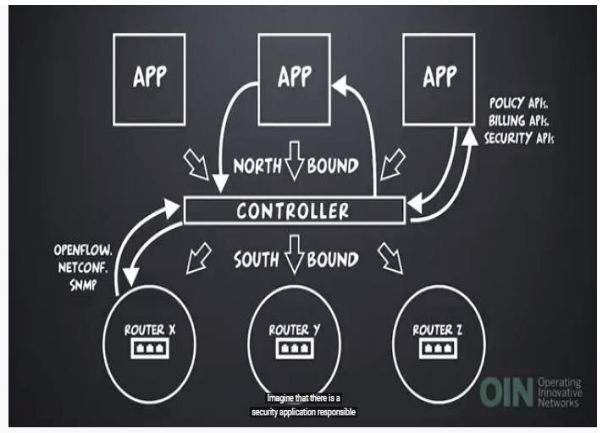
Software-defined networking (SDN) is an architecture designed to make a network more flexible and easier to manage. SDN centralizes management by abstracting the control plane from the data forwarding function in the discrete networking devices.

Traditional Networking:

- Traditional networking consists of the following traits:
- The functions of traditional networking are primarily implemented from dedicated devices, using one or more switches, as well as routers and application delivery controllers.
- The functionality of traditional networking is largely implemented in dedicated hardware, such as application-specific integrated circuits (ASIC). One of the negative aspects of this traditional hardware-centric networking is its limitations.

SDN elements

An SDN architecture delivers a centralized, programmable network and consists of the following:



- A controller, the core element of an SDN architecture, that enables centralized management and control, automation, and policy enforcement across physical and virtual network environments
- An **SDN controller** is an application in a software-defined networking (**SDN**) architecture that manages flow control for improved network management and application performance. The **SDN controller** platform typically runs on a server and uses protocols to tell switches where to send packets.
- Southbound APIs that relay information between the controller and the individual network devices (such as switches, access points, routers, and firewalls)

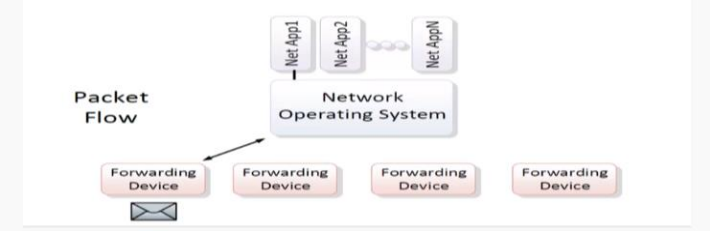
- Software-defined **southbound** application program interfaces (SDN **southbound APIs**) are used to communicate between the SDN Controller and the switches and routers of the network. They can be open-source or proprietary

- Northbound APIs that relay information between the controller and the applications and policy engines, to which an SDN looks like a single logical network device.
- Northbound APIs are the link between the applications and the SDN controller. The applications can tell the network what they need (data, storage, bandwidth, and so on) and the network can deliver those resources, or communicate what it has.

SDN adoption :

SDN has seen wide adoption across data centers (64%), WANs (58%), and access networks (40%).

SDN MODEL :

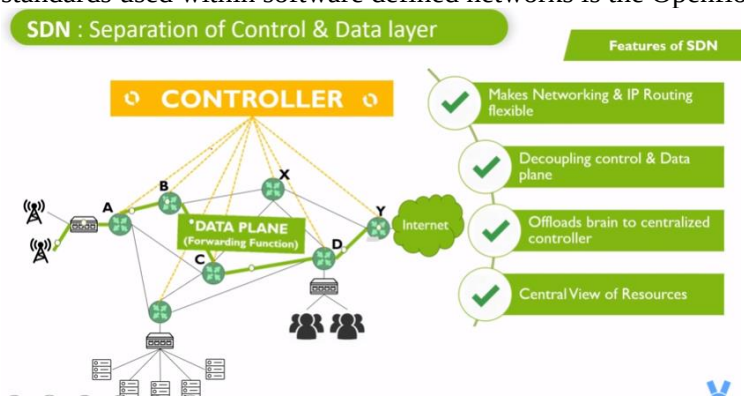


Software defined networking, SDN, is a network technology that is controlled by software functions to enable it to be adaptable, dynamic, manageable, and cost-effective. The SDN architecture decouples the network control and forwarding functions so that the network control is directly programmable and then the underlying infrastructure is abstracted for applications and network services.

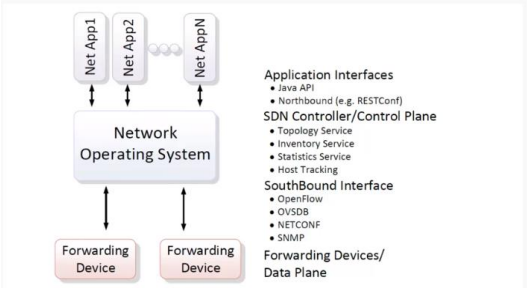
Some of the key concepts that are embedded in software defined networking include:

- ***Forwarding and control functions separated:*** By separating these functions it is possible to control the forwarding side of the network to meet the changing traffic flow requirements.
- ***Programmable:*** Not only is the network programmable, but the network control can be directly programmed because the control is decoupled from the forwarding functions.
  - ***Central management:*** One of the key concepts of software defined networking is that the network is controllable and software defined. This can only be achieved if the management is achieved using a central management core.
  - ***Programmatic configuration:*** software defined networking enables managers to configure, manage, secure, and optimize network resources. This can be achieved very quickly using automated programmes which monitor the network performance and implement the changes needed. In this way the data network can meet the ever changing demands placed upon it.
  - ***Open standards usage:*** One of the key requirements for software defined networking is that open standards are used. With data networks typically using network components from a variety of vendors it is essential that all these elements can operate together. This can only be achieved if

common open standards are used. If open standards were not used, there would be a host of different vendor specific interfaces that would not operate together. One of the key open standards used within software defined networks is the Openflow standard.



- The diagram below shows a logical view of software defined networking architecture with the three layers: application layer; control layer; infrastructure layer.

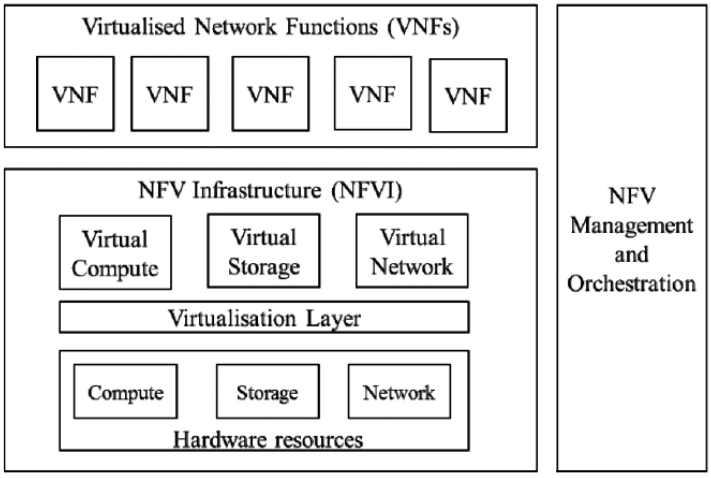


- **Software defined networking architecture**
- The intelligence for the network is typically contained within software defined networking controllers which are able to control the complete network. In this way, the whole network can be treated by the applications and policy entities as a single large logical switch.
- By adopting this software defined networking approach, the whole network can be controlled from a single point. This greatly simplifies the design, operation and updates. SDN also simplifies the network devices themselves as they only need to interface with a single control standard and not the many protocol standards they would otherwise need to process.

Network Functions Virtualization :

Network Functions Virtualization (NFV) is the decoupling of network functions from proprietary hardware appliances and running them as software in virtual machines (VMs). The different functions — such as firewalls, traffic control, and virtual routing — are called virtual network functions (VNFs).

NFV uses virtualized networking components to support an infrastructure totally independent of hardware. The standard resources of compute, storage, and network functions can all be virtualized and placed on commercial off-the-shelf (COTS) hardware like x86 servers. Having virtualized resources means that VMs can be given portions of the resources available on the x86 server. That way, multiple VMs can run on a single server and scale to consume the remaining free resources. This also means that resources are less often sitting idle and data centers with virtualized infrastructure can be more effectively used. Within the data center and the outside networks, the data plane and control plane can also be virtualized with NFV



NFV framework

As with any system, a network using NFV techniques can be broken down into a number of elements. Those for Network Functions Virtualization are:

- ***Virtualized network functions, VNF:*** The virtualised network functions comprise the software used to create the various network functions in their virtualised format. These are then deployed onto the hardware, i.e. the Network Function Virtualization Infrastructure.
- ***Network function virtualization infrastructure, NFVI:*** The NFVI consists of all the hardware and software components which are contained within the environment in which VNFs are deployed.

  One of the advantages of NFV is that the NFV-Infrastructure, NFVI can be located across several physical locations, allowing operators to typically place their centres at the most convenient locations. The network providing connectivity between these locations is part of the NFV-Infrastructure.
- ***Network functions virtualization management and orchestration architectural framework, NFV-MANO Architectural Framework:*** NFV- MANO consists of the various functional blocks in whatever form that enables the exchange information, manipulation and storage needed manage and run the NFVI and VNFs, the network to operate correctly and provide significant improvements in efficiency and performance over other forms of network.

The NFVI and the NFV-MANO areas of the network are built within the overall NFV platform. This platform implements carrier-grade features used to manage and monitor the various components, recover from failures and provide effective security. These functions are all needed to run a public carrier network.

Network functions virtualization :

Network functions virtualization (NFV) enables IT pros to modernize their networks with modular software running on standard server platforms.

Over time, NFV will deliver high-performance networks with greater scalability, elasticity, and adaptability at reduced costs compared to networks built from traditional networking equipment. NFV covers a wide range of network applications, but is driven primarily by new network requirements, including video, SD-WAN, Internet of Things and 5G.

According to ETSI, the goal of NFV is to transform the way that network operators architect networks by evolving standard IT virtualization technology to consolidate many network equipment types on to industry standard high-volume servers, switches and storage, which could be located in the data center, in the network or at end-customer premises. NFV replaces traditional, custom-designed network equipment (black boxes) that continues to dominate the installed base of networks.

Architecture

NFV provides for an open architecture with many flexible options for deploying an NFV solution. The typical architecture of NFV consists of three distinct layers:

- Network functions virtualization infrastructure (NFVi) – the hardware and infrastructure software platform required to run network applications.
- Virtual network functions (VNFs) – software applications that deliver specific network functions, such as routing, security, mobile core, IP multi-media subsystems, video, etc.
- Management, automation and network orchestration (MANO) – the framework for management and orchestration of NFVi and various VNFs.

The Benefits of NFV

Network operators who virtualize their network can save money, shorten the time-to-market for new or updated products, and better scale and adjust resources available to applications and services.

Other benefits include:

Less Vendor Lock-in: Running VNFs on COTS hardware, which means organizations aren't locked into proprietary, fixed-function boxes that take truck rolls and significant time and labor to deploy and configure.

Greater Resource Efficiency: A virtualized data center or other infrastructure is more efficient to operate because more can be done with less. Data center footprint, power consumption, and cooling requirements can all be reduced or kept the same, but with increased workload capacity. This is possible because a single server can run multiple VNFs at once, so not as many servers are needed to do the same amount of work. When network demand changes, an organization can update its infrastructure through software instead of doing another truck roll. The instances where an organization needs to physically update its network and data centers are significantly reduced.

Flexibility: Organizations can use the agility of NFV to quickly adapt to changing business requirements and new market opportunities. In other words, the time-to-market period is shortened because the network infrastructure can be changed to adequately support the organization's new products. A network that has gone through NFV is also able to adjust quickly and easily to changes in resource demand as traffic coming to the data center increases or decreases. Scaling up and down in the number of VMs and the resources provided to them can be done automatically through SDN software.

Challenges Associated with NFV

Challenges around NFV are rooted in three components of the technology: the NFV manager (NFVM), VNFs, and the NFV infrastructure (NFV infrastructure (NFVI)). The three components are so tightly joined together that instead of the theoretical simplification for network operators, in practice it adds complexity and difficulty when deploying NFV at scale.

The Lean NFV has attempted to solve these problems by developing a new approach to NFV architecture.

"The complexity currently hindering NFV arises not from how any one of the above pieces is built, but instead from how they are woven together into an overall system," the organization said in a 2019 white paper. "More specifically, the complexity arises when the NFV manager is integrated with the existing computational infrastructure, when VNFs are integrated with the NFV manager, and when the coordination is required between the various components of the NFV manager."

The white paper goes on to say that the focus needs to be on simplifying the three points of integration so other elements of NFV designs can be innovated more freely.

One reason there is complexity in the components of NFV technology is because there have been multiple organizations trying to standardize them. This led to no single approach that has worked for the whole industry and no set of standards that stood out enough to be more heavily invested in or adopted.

Difference between SDN & NFV

Software defined networking and network functions virtualization are very closely linked as they both have software as the key, but they are not the same.

The two techniques can be used together or separately. The main points of each are summarised below so that both SDN and NFV can be evaluated with their similarities and differences.

- ***SDN, Software Defined Networks:*** SDN separates the network control and forwarding planes and provides a central view for more efficient implementation and running of the network services.
- ***NFV, Network Functions Virtualisation:*** NFV focuses on optimising the network services themselves. This technique decouples the network functions from proprietary hardware, placing them on more generic servers or computers so these functions can run in software to provide more flexibility for operation, changes and updates.

Although the two are similar the difference between SDN and NFV means that they are not the same. However both techniques can be used on the same network to provide significant benefits.

Software defined networks are now well established. As the efficiency of data networks needs to improve as costs per bit fall, SDN provides one path by which the effectiveness of networks can image more adaptable, dynamic, and manageable, whilst providing significant improvements in terms of their cost effectiveness.

NFV refers to the virtualization of network components, while SDN refers to a network architecture that injects automation and programmability into the network by decoupling network control and forwarding functions. In other words, NFV virtualizes network infrastructure and SDN centralizes network control. Combined, SDN and NFV create a network that is built, operated, and managed by software.

An SDN typically has an SDN controller, northbound application program interfaces (APIs), and southbound APIs. The controller allows network administrators to view the network and dictate behaviors and policies to the underlying infrastructure. Southbound APIs take information about the state of the network from that infrastructure and send it back to the controller, which is necessary to keep the network running smoothly. Applications and services use northbound APIs to communicate their resource needs to the controller.

Software defined networking (SDN) is typically defined as the separation of the forwarding and control planes in a network element. It provides improved control/management as well as network programmability. SDN is distinct from NFV – but many NFV deployments may use SDN controllers as part of the overall NFV architecture.

Network function virtualization (**NFV**) and software-defined networks (**SDN**) are two closely related technologies that often exist together, but not always. So let's understand the basic difference between these two closely related words ---

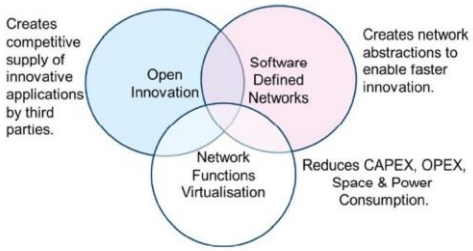
An SDN can be considered a series of network objects (such as switches, routers, firewalls) that deploy in a highly automated manner.

The automation may be achieved by using commercial or open source tools customized according to the administrator's requirements. A full SDN may only cover relatively straightforward networking requirements, such as VLAN and interface provisioning.

In many cases, SDN will also be linked to server virtualization, providing the glue that sticks virtual networks together. This may involve NFV, but not necessarily.

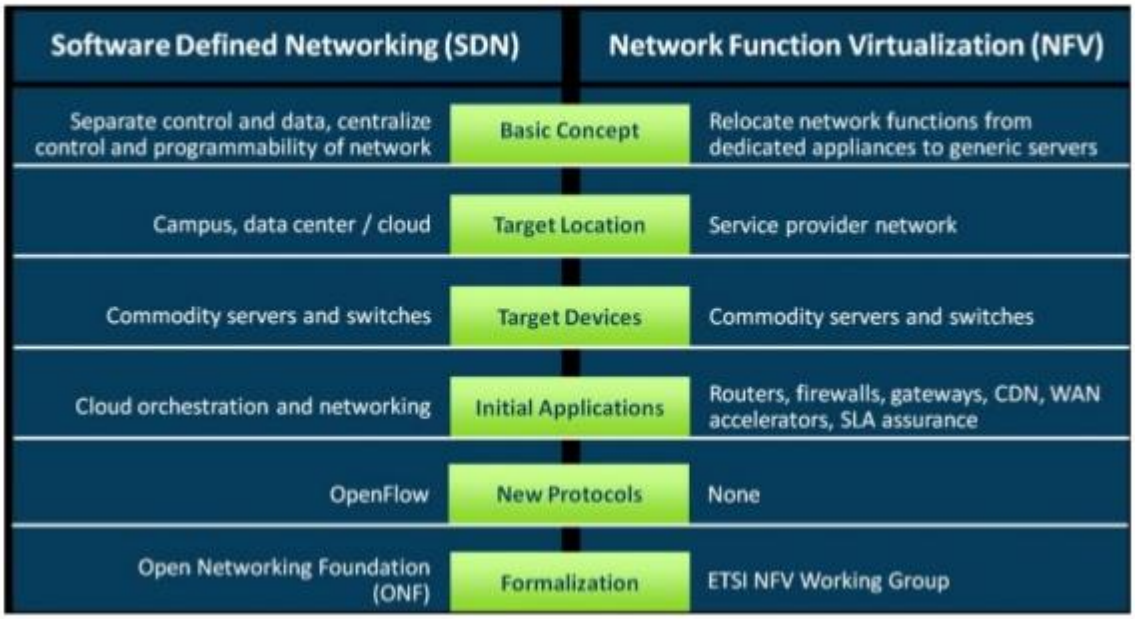
NFV is the process of moving services, such as load balancing,firewalls and IPS, away from dedicated hardware into a virtualized environment. This is, of course, part of a wider movement toward the virtualization of applications and services.

A good example is virtualized application delivery controllers (ADCs). With careful configuration, it is possible to react to the network state and spin up or down application servers as demands rise and fall. *So it's perfectly possible to have NFV without the inclusion of a full-blown SDN. The two are often deployed together, and an SDN that drives NFV is a very powerful combination.*

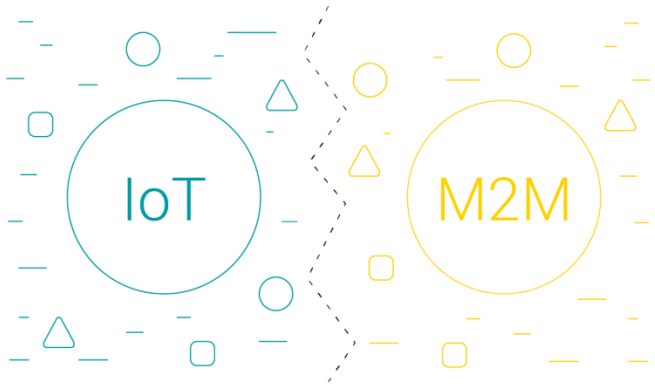


Neither NFV nor SDN were turnkey solutions earlier -- a great deal of integration and policy design still need to happen. This can become a reality for many enterprises, but the harness is not entirely in place. That said, the tools are rapidly evolving, and many vendors are bringing technologies to market that support SDN or NFV deployments. Ultimately, the implementation of either or both technologies will be driven by the business needs.

NFV vs SDN



IoT vs M2M — What is the Difference?

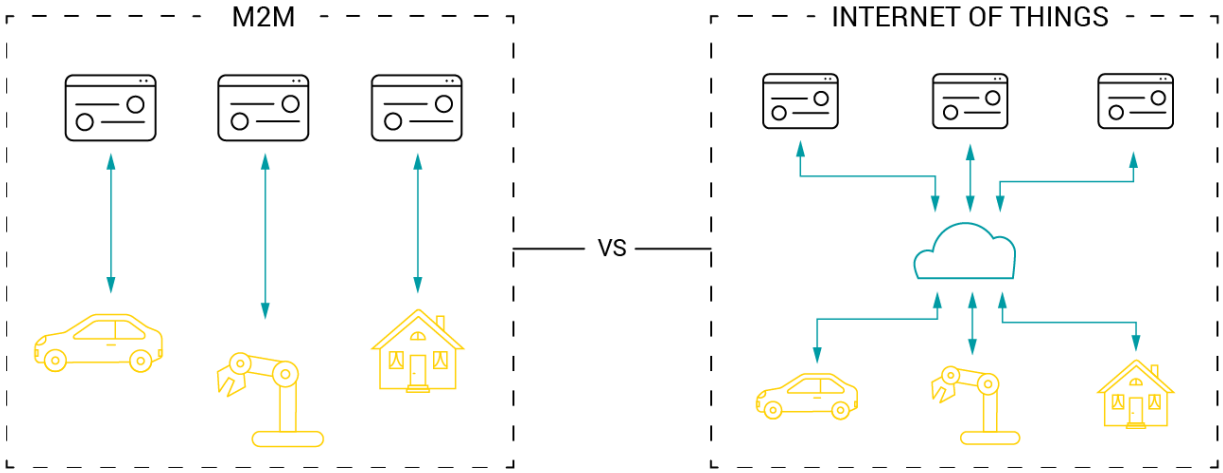


It seems that recently there has been as much hype surrounding the IoT as there is confusion in differentiating it from the M2M technology. This short overview will tackle the popular misconceptions and pinpoint some pivotal differences between these two technology buzzwords.

Understanding the differences between M2M and IoT

Whether at home, at work or in the street, we are constantly surrounded by IoT devices. The Internet of Things, probably the fastest growing phenomenon in the whole IT world, has already become a fully-fledged partner in our day-to-day activities, with smartphones, smart cars, smart coffee machines and an endless list of other smart appliances gradually making their way into our lives and starting to drive our future. Therefore, getting the hang of the differences between the Internet of Things and Machine-To-

Machine (M2M) communication, which is the underlying concept that gave rise to the IoT as we know it, seems to be an essential element in understanding what the whole IoT scene is all about nowadays.
To fully understand the differences between M2M and IoT it's nice to first know what M2M is. Luckily, the definition of M2M is (seemingly) not that complicated: M2M, or machine-to-machine communication, derives from telemetry technology and generally refers to data exchange between various devices (usually) through the Internet without human participation. While many people, some IT know-it-alls included, often treat both terms as synonyms and use them interchangeably, it doesn't necessarily take a rocket scientist to spot one basic characteristic that differentiates them. As predecessor to the IoT, M2M has been used throughout the decades as the standard technology in telemetry even before the invention of the Internet itself, as it involved an interaction between two or more machines without human intervention. The idea of the Internet of Things, on the other hand, having evolved on the foundations laid down by M2M, aims at offering much more functionality. It uses Internet connectivity not only to enable communication between a fleet of the same kind of machines, but also to unite disparate devices and systems in an effort to marry different technology stacks and deliver interactive and fully integrated networks across varying environments.



A real life example of this dissimilarity could be found in telemedicine. Let us imagine a solution that connects a sensor monitoring the heart rate of a patient to an external application which lets the doctor know the patient needs attention. Such kinds of solutions could easily be provided by the M2M technology. On the other hand, if we take a sensor and integrate it with an interactive pillbox that would advise the patient to take the medicine and, moreover, would be able to send alerts to their family members' smartphones that the medicine has not been taken from the pillbox, it would definitely involve an IoT approach. To cut the story short, IoT could be viewed as M2M, but acting in a wider context.

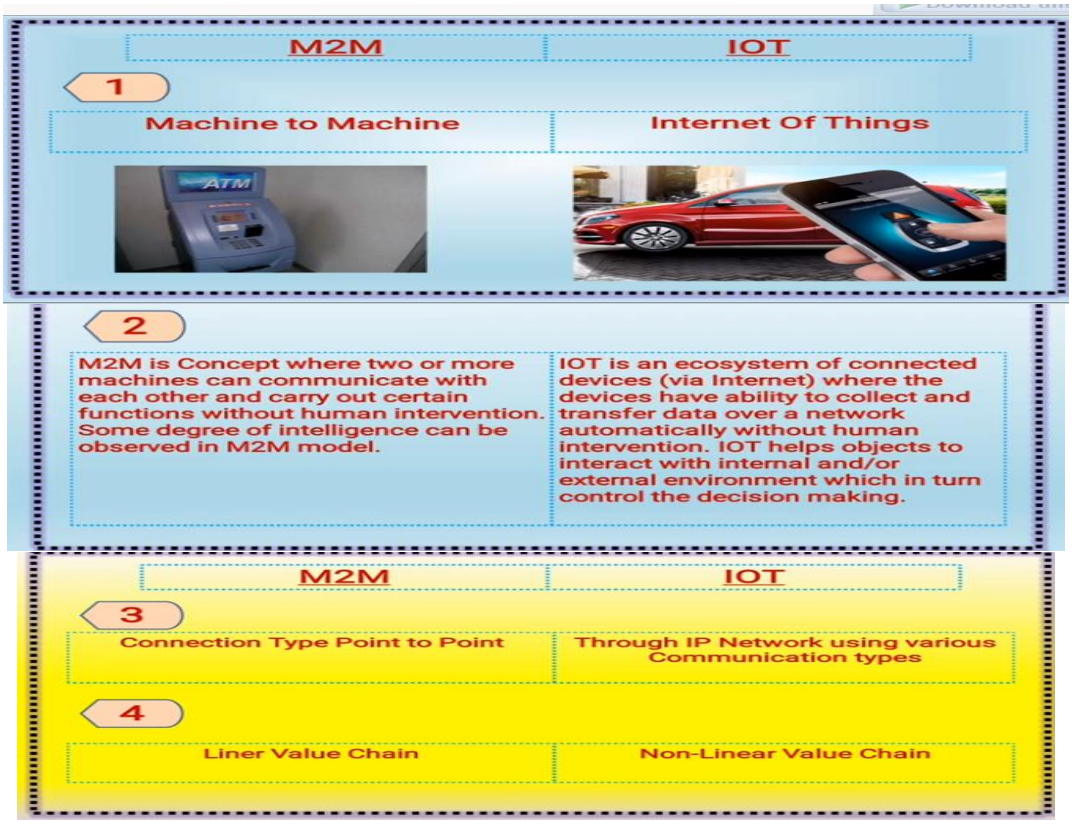
Scale: The main difference between IoT and M2M

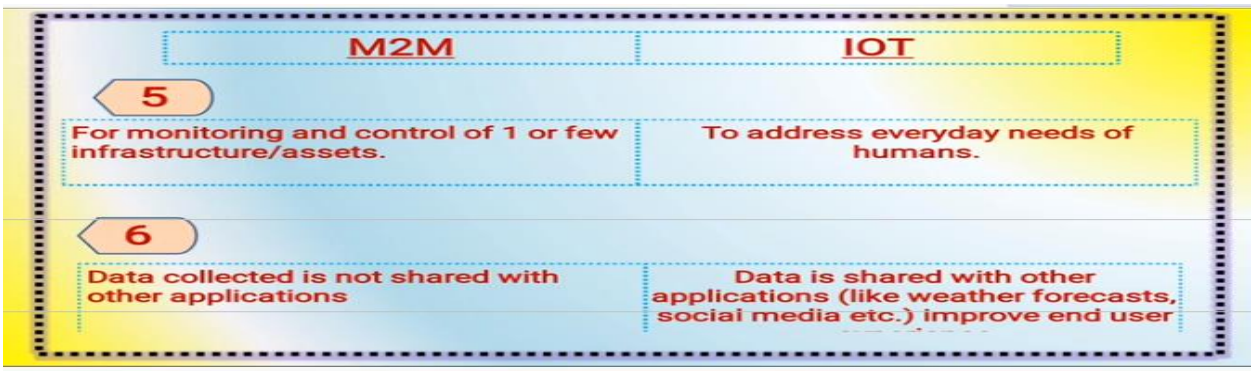
Bearing all that in mind, it seems that the whole discussion of differences could be quite conveniently boiled down to one word: scale. And very rightly so, yet there is much more to it than this.
Historically speaking, M2M technology harks back to the invention of two-way radio networks in the beginnings of the 20th century which fuelled the development of telemetry, a major industry area in which M2M has found numerous applications. With the rise of GSM data connectivity in the 1990s, M2M entered a new phase in its development, to reach its prime only in the first decade of the 21st century. The break-neck speed at which the Internet technology has been developing since the beginnings of the 2010s has recently forced M2M to give way to the IoT as the primary concept that is shaping the way we think about the future of business and our everyday life. To put matters simple, M2M typically relies on communication provided via either cellular or wired networks, which is quite restraining in

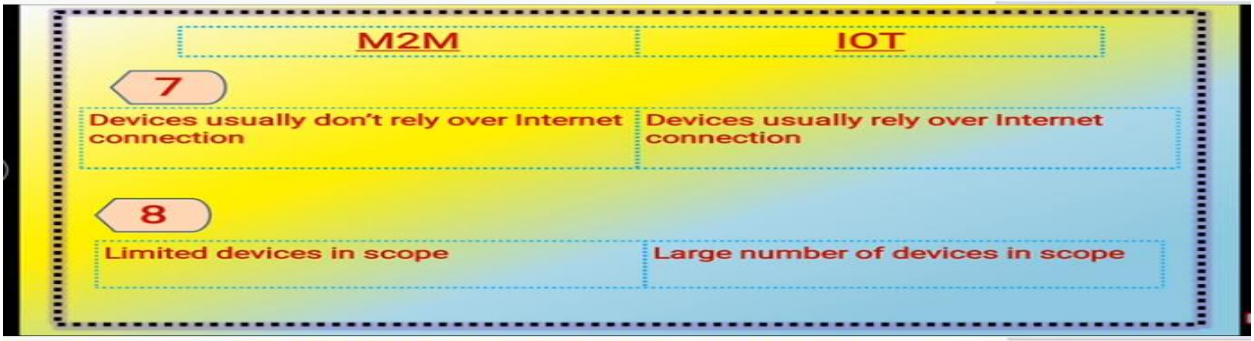
itself, while the IoT can benefit from the more effective and open-ended IP-based networks and transfer the gathered data to cloud-based environments.
In addition to the differences in how remote device access is achieved by each technology, there is also the question of scalability. Quite understandably, IoT is inherently more scalable, since it encompasses a range of cutting-edge connectivity solutions to synergise disparate stacks of technology. In turn, these are usually driven by the M2M platforms. In this respect, M2M can be seen as IoT's poor relation, offering limited integration possibilities, since the devices it connects must have a shared communication standard.

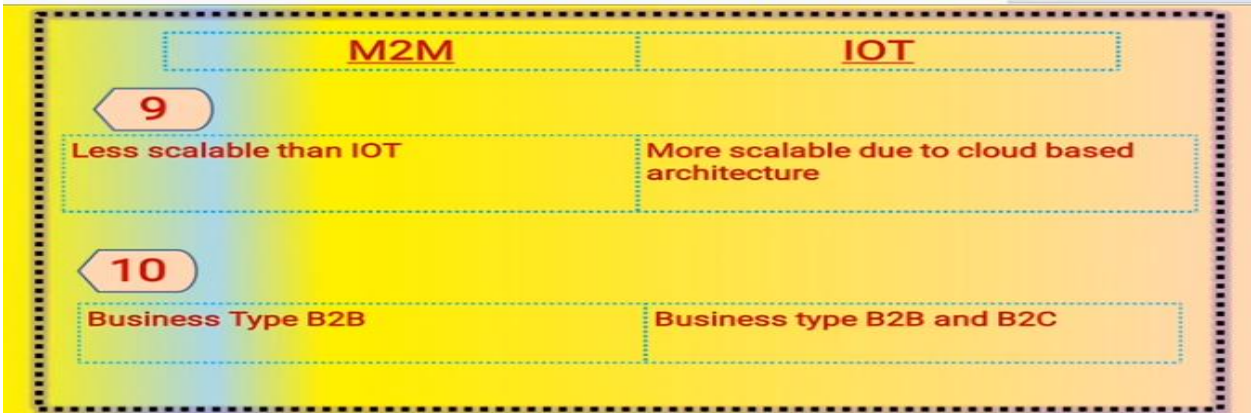
M2M	IoT
Simple device-to-device communication usually within an embedded software at client site	Grand-scale projects and want-it-all approach
Isolated systems of devices using same standards	Integrates devices, data and applications across varying standards
Limited scalability options	Inherently more scalable
Wired or cellular network used for connectivity	Usually devices require active Internet connection
Extensive background of historical applications	State-of-the-art approach with roots in M2M

Differences between M2M and IoT




M2M
IOT
5
For monitoring and control of 1 or few infrastructure/assets.
To address everyday needs of humans.
6
Data collected is not shared with other applications
Data is shared with other applications (like weather forecasts, sociai media etc.) improve end user


M2M
IOT
7
Devices usually don't rely over Internet connection
Devices usually rely over Internet connection
8
Limited devices in scope
Large number of devices in scope


M2M
IOT
9
Less scalable than IOT
More scalable due to cloud based architecture
10
Business Type B2B
Business type B2B and B2C

Key Differences

Domain	IOT	M2M
Application	IoT data can be stored over cloud hence can be used for analytics applications, remote diagnosis etc.	M2M data is more preferably stored in point solutions and can be accessed locally. Hence used for on-premise solutions and local diagnosis.
Protocols	IoT & M2M differ in communication protocols and machines communicate. In IoT, they focus on protocols above network Layer e.g. HTTP, COAP,DDS etc.	M2M uses proprietary and non ip based communication protocols e.g. Zigbee, Bluetooth, Modbus, IEEE 802.15.4
Hardware	IoT is more focused on sensors and interfacing. Various components of IoT systems are sensors, internet and Networking Infrastructure.	M2M is typically more emphasized on embedded Hardware.
Machines vs Things	The things in IoT refers to objects which are uniquely identifiable and can sense data	In contrast, M2M have homogeneous machines within network.

IoT vs M2M — synonyms or completely different technologies?

In any case, it can be concluded that the IoT and M2M technologies are similar in that they both provide solutions for collection, storage and exchange of data between devices under minimal human supervision. What also unites them is the fact that they are increasingly closer to each other in matters of device management.

While it is true that the IoT has been largely based on the foundations provided by M2M solutions, it must be added that it has been improving on them ever since it was established as one of the main sources of innovation in the lives of individuals, businesses and whole societies.

M2M& IOT VALUE CHAIN– A Basic Perspective

Definition of M2M

M2M:Machine to machine refers to technology that allowed both wireless and wired systems to communicate with other devices of the same type.

Digi: Machine to Machine (M2M) technology allows organization to gather data from the edge of the enterprise and apply it in way that positively impact the business.

Orange: exchange the information between machine that is established between central control system (server) and any type of equipment, through one or several communication networks.

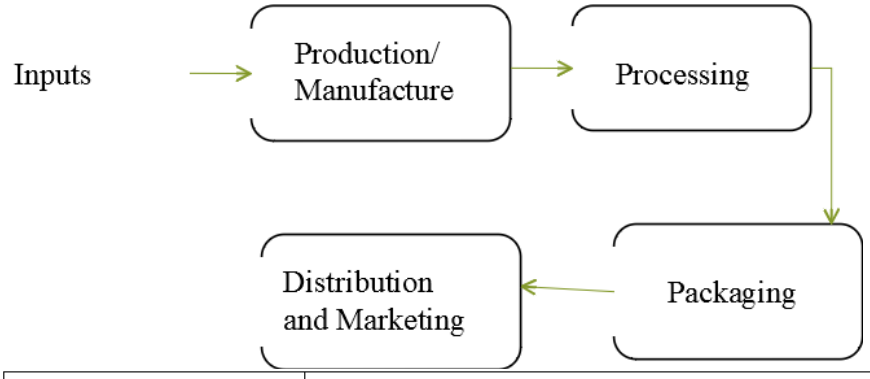
Global value chains

A value chain describes the full range of activities that firms and workers perform to bring a product from its conception to end use and beyond, including design, production, marketing, distribution, and support to the final consumer

Analyzing an industry from a global value chain (GVC) perspective permits understanding of the context

of globalization on the activities contained within them by "focusing on the sequences of tangible and intangible value-adding activities, from conception and production to end use. GVC analysis therefore provides a holistic view of global industries both from the top down and from the bottom up".

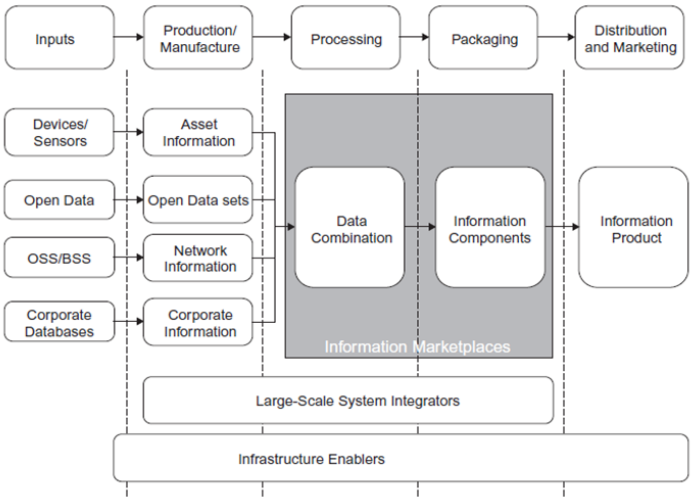
M2M value chain



Input	Inputs are the base raw ingredients that are turned into a product.
Example	cocoa beans for the manufacture of chocolate
M2M example	data from an M2M device that will be turned into a piece of information.
Production/ Manufacture:	Production/Manufacture refers to the process that the raw inputs are put through to become part of a value chain.
Example	cocoa beans may be dried and separated before being transported to overseas markets.
M2M example	Data from an M2M, needs to be verified and tagged for provenance.
Processing:	Processing refers to the process whereby a product is prepared for sale.
Example	cocoa beans may now be made into cocoa powder, ready for use in chocolate bars.
M2M example	M2M refers to the aggregation of multiple data sources to create an information component something that is ready to be combined with other data sets to make it useful for corporate decision-making.

Packaging:	Packaging refers to the process whereby a product can be branded as would be recognizable to end-user consumers.
Example	A chocolate bar would now be ready to eat and have a red wrapper with the words "KitKatt" on it.
M2M example	M2M data will have to be combined with other information from internal corporate databases, for example, to see whether the data received requires any action. This data would be recognizable to the end-users that need to use the information, either in the form of visualizations or an Excel spreadsheet.
Distribution/ Marketing:	This process refers to the channels to market for products.
Example	a chocolate bar may be sold at a supermarket, a kiosk, or even online.
M2M example	will have produced an Information Product that can be used to create new knowledge within a corporate environment examples include more detailed scheduling of maintenance based on real-world information or improved product design due to feedback from the M2M solution.

IoT value chains



Inputs:	significantly more inputs than for an M2M solution
Devices/Sensors:	data from devices and sensors is used to provides a different and much broader marketplace than M2M does.
Open Data:	A piece of data is open if anyone is free to use, reuse, and redistribute it subject only, at most, to the requirement to attribute and/or share-alike. **Example**: city maps, provided by organizations such as Ordinance Survey in the United Kingdom.
OSS/BSS:	The Operational Support Systems (OSS) and Business Support Systems (BSS) closed information marketplaces that allow operators to deliver services to enterprises. **Example**: where phone usage data is already owned by the company.
Corporate Databases:	Companies of a certain size generally have multiple corporate databases covering various functions, including supply chain management, payroll, accounting As the use of devices and sensors increases, these databases will be connected to this data to create new information sources and new knowledge.
Production/ Manufacture:	Process will need to include tagging and linking of relevant data items in order to provide provenance and traceability across the information value chain.
Asset Information:	Asset information may include data such as temperature over time of container during transit or air quality during a particular month.
Open Data Sets:	maps, rail timetables, or demographics about a certain area in a country or city.
Network Information:	GPS data, services accessed via the mobile network
Corporate information:	The current state of demand for a particular product in the supply chain at a particular moment in time.
Processing:	The data from the various inputs from the production and manufacture stage are combined together to create information.

Packaging:	The packaging section of the information value chain creates information components. These components could be produced as charts or other traditional methods of communicating information to end-users.
Distribution/ Marketing:	The final stage of the Information Value Chain is the creation of an Information Product.
Information products for improving internal decision-making:	These information products are the result of either detailed information analysis that allows better decisions to be made during various internal corporate processes, or they enable the creation of previously unavailable knowledge about a company's products, strategy, or internal processes.
Information products for resale to other economic actors:	These information products have high value for other economic actors and can be sold to them.

Industrial internet of things (IIoT)

The industrial internet of things (IIoT) is the use of smart sensors and actuators to enhance manufacturing and industrial processes. Also known as the industrial internet or Industry 4.0, IIoT leverages the power of smart machines and real-time analytics to take advantage of the data that 'dumb machines' have produced in industrial settings for years. The driving philosophy behind IIoT is that smart machines are not only better than humans at capturing and analyzing data in real time, they are better at communicating important information that can be used to drive business decisions faster and more accurately.

Connected sensors and actuators enable companies to pick up on inefficiencies and problems sooner, and save time and money in addition to supporting business intelligence (BI) efforts. In manufacturing specifically, IIoT holds great potential for quality control, sustainable and green practices, supply chain traceability and overall supply chain efficiency. In an industrial setting, IIoT is key to processes such as predictive maintenance (PdM), enhanced field service, energy management and asset tracking.

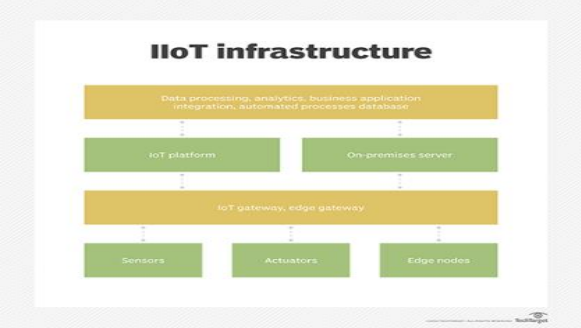
How IIoT works

IIoT is a network of intelligent devices connected to form systems that monitor, collect, exchange and analyze data. Each industrial IoT ecosystem consists of:

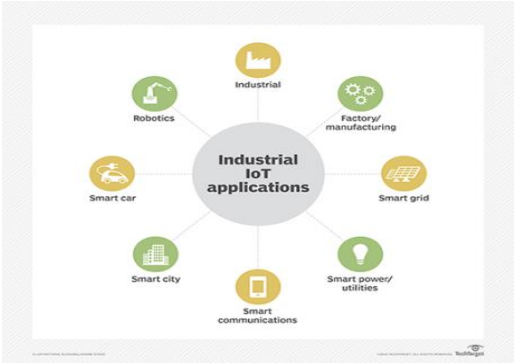
- Intelligent assets that can sense, communicate and store information about themselves;

- Public and/or private data communications infrastructure
- Analytics and applications that generate business information from raw data; and
- People.

Edge devices and intelligent assets transmit information directly to the data communications infrastructure, where it is converted into actionable information on how a certain piece of machinery is operating, for instance. This information can then be used for predictive maintenance, as well as to optimize business processes



Benefits of IIoT



Organizations can use real-time data generated from IIoT systems to predict defects in machinery, for example, before they occur, enabling companies to take action to address those issues before a part fails or a machine goes down.

Another common benefit is improved field service. IIoT technologies help field service technicians identify potential issues in customer equipment before they become major issues, enabling techs to fix the problems before they inconvenience customers.

Asset tracking is another IIoT perk. Suppliers, manufacturers and customers can use asset management systems to track the location, status and condition of products throughout the supply chain. The system will send instant alerts to stakeholders if the goods are damaged or at risk of being damaged, giving them the chance to take immediate or preventive action to remedy the situation.

IIoT also enables enhanced customer satisfaction.

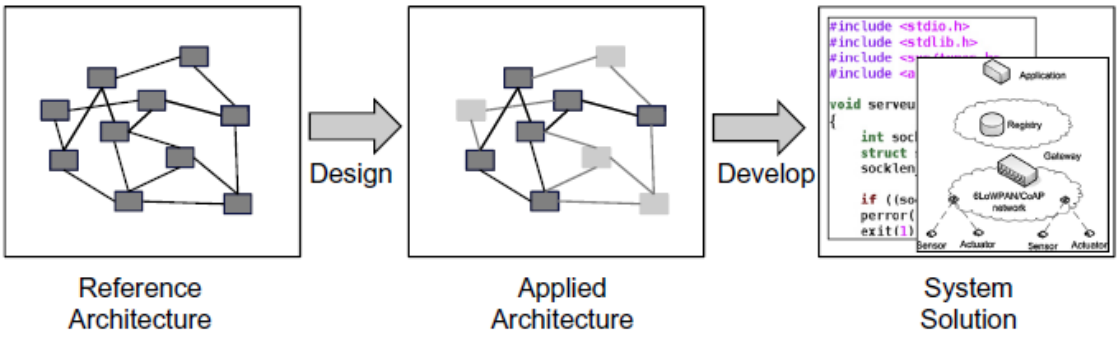
Sensors can monitor vibrations, temperature and other factors that might lead to operating conditions that are less than optimal.

UNIT-3

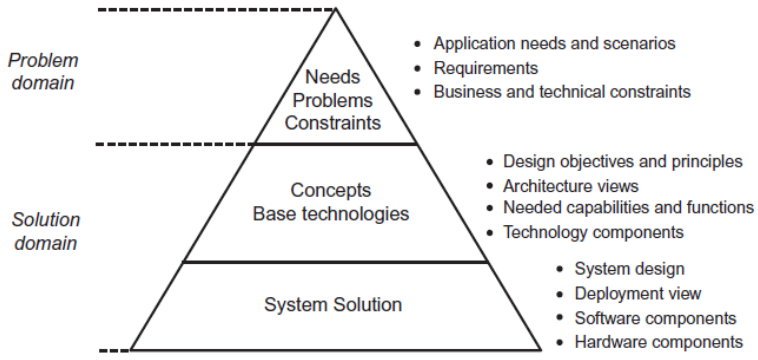
IoT Architectural and Wireless Technologies for IoT

Building An Architecture

Architecture refers to the description of the main conceptual elements, the actual elements of a target system, how they relate to each other, and principles for the design of the architecture. The applied architecture is then the blueprint used to develop the actual system solution.



From a reference architecture to a system solution.



Problem and Solution domain partitioning.

Main Design Principles And Needed Capabilities

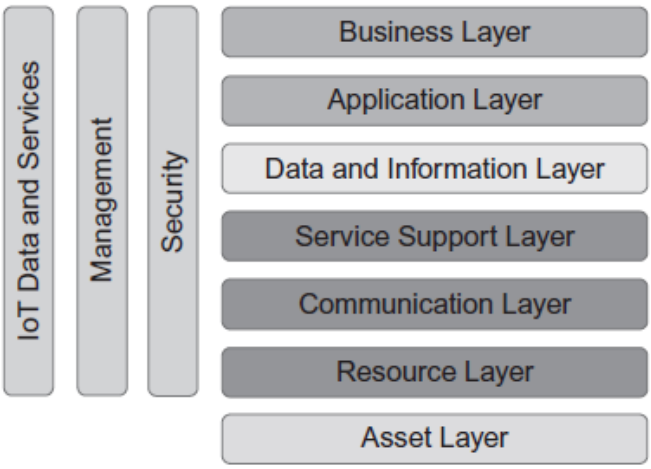
- The architecture relies on the separation of resources providing sensing and actuation from the actual devices, a set of contextual and real world entity-centric services, and the users of the services.
- SENSEI further relies on an open-ended constellation of providers and users, and also provides a reference model for different business roles.
- A number of design principles and guidelines are identified, and so is a set of requirements.
- Finally, the architecture itself contains a set of key functional capabilities.
- IoT-A refers to as the Architectural Reference Model (ARM).

- The vision of IoT-A is, via the ARM, to establish a means to achieve a high degree of interoperability between different IoT solutions at the different system levels of communication, service, and information.
- IoT-A provides a set of different architectural views, establishes a proposed terminology and a set of Unified Requirements.

1. The overall design objective of IoT architecture shall be to target a horizontal system of real-world services that are open, service-oriented, secure, and offer trust.
2. Design for reuse of deployed IoT resources across application domains.
3. Design for a set of support services that provide open service-oriented capabilities and can be used for application development and execution.
4. Design for different abstraction levels that hide underlying complexities and heterogeneities.
5. Design for sensing and actors taking on different roles of providing and using services across different business domains and value chains.
6. Design for ensuring trust, security, and privacy.
7. Design for scalability, performance, and effectiveness.
8. Design for simplicity of management.
9. Design for different service delivery models.
10. Design for lifecycle support.

An Iot Architecture Outline

Functional layers and capabilities of an IoT solution



Asset Layer

Interest	Example
real-world objects and entities that are subject to being monitored and controlled, as well as having digital representations and identities.	•Vehicles, •home, •human, •buildings, etc
Assets can also be of a more virtual character, being subjective representations of parts of the real world.	Latter → routes → trucks →logistics → Information → traffic intensity → roadwork, → road conditions →weather situation.
Assets are instrumented with embedded technologies that bridge the digital realm with the physical world, and that provide the capabilities to monitor and control the assets as well as providing identities to the assets.	•Tags: -RFID •Optical Codes:- barcodes • QR(Quick Response) code

Communication Layer

connectivity between the resources on one end and the different computing infrastructures that host and execute service support logic and application logic on the other end.

Network	Description
LAN	Local Area Network
WAN	Wide Area Network •wired • wireless technologies •Public • private
WPANs	Wireless Personal Area Networks •Fitness •Health care
HAN and BAN	Home and Buildings Area Networks •Automation and control applications
NAN	Neighbourhood Area Networks •Distribution Grid of a Smart Electricity Grid
V2V	Vehicle-to-Vehicle •Collision avoidance
ZigBee	Protocol stack for home automation •Exchanging service operations using HTTP as the means for communication

Service Support Layer

Interest	Example
support services can provide uniform handling of the underlying devices and networks, thus hiding complexities in the communications and resource layers.	support services can provide uniform handling of the underlying devices and networks, thus hiding complexities in the communications and resource layers.
Communication-related functions include selection of communication channels if different networks can be used in parallel	•reliability purposes, •publish subscribe and •message queue mechanisms
Location Based Service (LBS) and Geographic Information System (GIS) services	

Data and Information Layer

Purpose: To capture knowledge and provide advanced control logic support.
Concept: Knowledge Management Framework (KMF) as a collective term to include data, information, domain-specific knowledge, actionable services descriptions as service descriptors, rules, process or workflow descriptions, etc.
Key concepts: data and information models and knowledge representation in general, and the focus is on the organization of information.

Interest	Example
Open-ended array of different applications	smart metering in the Smart Grid, vehicle tracking, building automation,
supporting the core business or operations of any enterprise, organization, or individual that is interested in IoT applications.	•Customer Relationship Management (CRM), •Enterprise Resource Planning (ERP), or •Business Support Systems (BSS).
The business layer also provides exposure to APIs for third parties to get access to data and information, and can also contain support for direct access to applications by human users.	•city portal services for citizens in a smart city context, •providing necessary data visualizations to the human workforce in a particular enterprise.

Management

Interest	Example
deals with management of various parts of the system solution related to its operation, maintenance, administration, and provisioning.	• management of devices, • communications networks • general Information Technology (IT) infrastructure .

Security

Security is about protection of the system, its information and services, from external threats or any other harm.
Security measures are usually required across all layers, for instance, providing communication security and information security.
key capabilities : 1. Trust and identity management, 2. authentication and authorization

Data and Services

Data and Service processing can, from a topological perspective, be done in a very distributed fashion and at different levels of complexity.
Basic event filtering and simpler aggregation, such as data averaging, can take place in individual sensor nodes in WSANs, contextual metadata such as location and temporal information can be added to sensor readings, and further aggregation can take place higher up in the network topology.

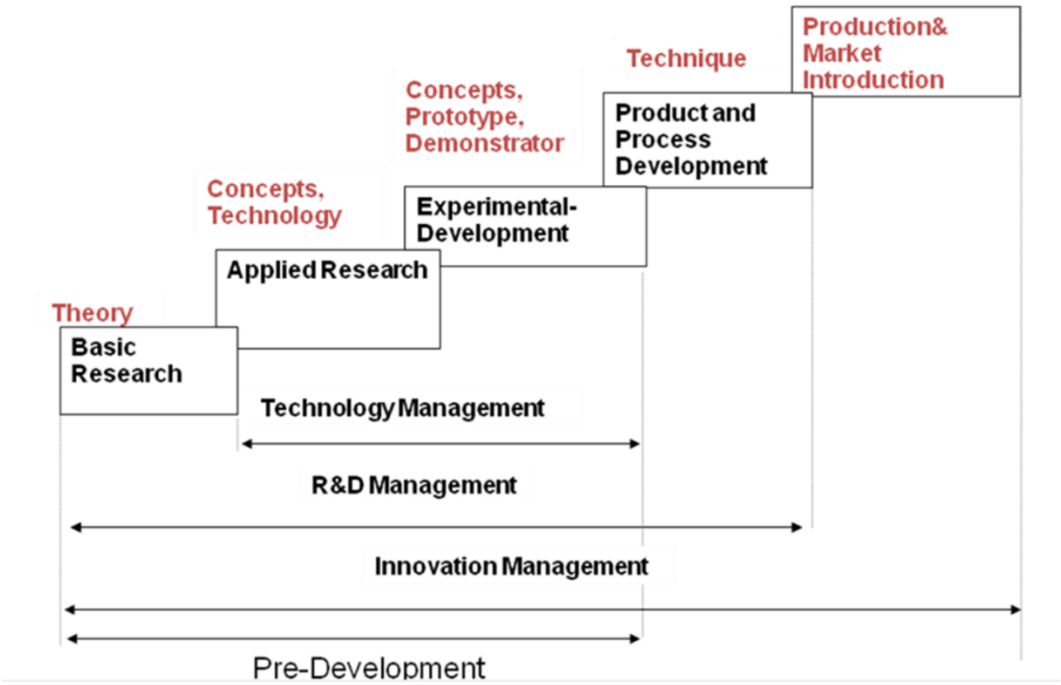
IoT Standards and Considerations:

- This section was originally created with IERC (www.internet-of-thingsresearch.eu) stakeholders to link their IoT research, development and innovation activities to international standard organisations, including ETSI, ITU-T, CEN/ISO, CENELEC/IEC, IETF, IEEE, W3C, OASIS, oneM2M and OGC.

- In 2013 the IERC IoT standard coordinators have asked contributors to focus on latest IoT standardisation issues and to recommend candidate organisations where technical specifications and standards should be developed
- **What is standardisation-** Standardisation is a voluntary cooperation among industry, consumers, public authorities and other interested parties for the development of technical specifications based on consensus.
- Standardisation complements market-based competition, typically in order to achieve objectives such as the interoperability of complementary products/services, to agree on test methods and on requirements
- For safety, health and environmental performance. Standardization also has a dimension of public interest.
- Standard makers should be close to standard users/implementers.
- What are the gaps between IoT standardization, IoT research, IoT development and IoT innovation.

There are gaps between IoT standardization and IoT Research, Development and Innovation life cycle.

How do IERC stakeholders bridge them?



- In order to fill gaps between IoT Research, Development and Innovation and standardisation life cycles (shown in Figure), IERC encourages the creation of pre standardisation groups.

- They allowed to build communities around consensus to develop standards, for example on Semantic Interoperability. Because of many options, IERC has helped to select and coordinate a lot of standards initiatives.
- IERC is also required to keep IoT Research and Development close to industry innovation and market.
- Industrial workshops have been co-organised with project and the European Commission in order to feed back IoT standardisation activities conducted by industrial stakeholders into EC funded projects.
- For example ETSI has coorganised workshops on Future Networks, M2M, Cloud, Smart Cities, ITS and RRS.
- IoT communities also welcomed the organisation of events (like Plugtests/Plugfests, Connectathon, Bake-off) focussing on interoperability testing, coexistence trials and compatibility involving applications or pilots/trial.
- Next workshops and interoperability "Plugtests/Connectathon" events should focus on IoT performance, optimization, quality (QoS, QoE), trust, safety, privacy, governance and security.
- While pre-standardisation like conducted in IRTF/ISOC, ITU-T Focus Group, IEEE-SA Industry Connection Program and ETSI Industry Specification Groups facilitates to bridge the gaps between research and standardisation, the on-time creation of Technical Committees (like ETSI TC M2M, TC NTECH) and international Partnership Projects (like ETSI 3GPP and oneM2M) helps to link the international industry with IERC research.

What are current IoT requirements?

- Without IoT standards, FI-WARE (www.fi-ware.eu)for example would not have been able to successfully provide open "Generic Enablers" for Future Internet/IoT developments in Phase 2 and 3 of FI-PPP (www.fi-ppp.eu).
- In IERC standardisation coordination meetings the most important IoT requirements for cross-domain standardisation were about cybersecurity, privacy, identification, traceability, anonimization, semantic interoperability, interoperability or coexistence testing, performance characterization and scalability, autoconfiguration, discovery, self-configuration, service robustness and resilience.
- Future standards adopters must be the standards makers.
- They know best what they need to drive their business.
- There is a risk that standards are not used if these two kinds of actors are different.

- An incentive to facilitate common early standard development is to include pre-standardisation “work packages” within research projects proposals.
- However, there could be a lack of industrial involvement.
- This is why IERC tries to be a central reference for pre-standardisation activities of EC IoT research projects to increase overall efficiency and raise mutual awareness, defragment and synergize in one unique place important information for stakeholders: Industry, Standard Development Organisations (SDOs), European Commission (EC).
- Before enforcing EC priorities using EU Regulation (Communications, Recommendations, Mandates or Directives) the EU funded programs are giving indication to proposers on EC priorities and domains, SDO / pre-standardisation activities to use, other ongoing projects, actions and deliverables to coordinate with.
- The IERC exists exactly for that, it allows exchanges between IERC, other EC clusters and projects like Future Networks, Cloud, FI-PPP, and FIA. This helps to detect standards gaps and overlaps and to link with regulation.

M2M SERVICE LAYER STANDARDISATION

M2M service layer in IoT:

- In order to be able to move from a vertical only approach to an integrated horizontal approach a standardisation of generally used service for the communication between devices and devices and applications is essential.
- Only by a world-wide standardisation on a protocol layer between transport and application a smooth integration of the diverge underlying communication technologies on the lower layers can be guaranteed.
- Already in a single vertical application domain a large variety of different communication standards exist and will exist in the future.
- It is not realistic to assume that a single standard on the lower protocol layers can be defined.
- Thus the integrating mechanism of the future horizontally integrated Internet-of-Things need to be a common cross vertical service layer.
- This service layer has to provide a set of general services to the applications at all component of the overall architecture from the devices level over the gateways to the network domain.
- A future worldwide standardised M2M service layer including definitions of interworking with existing underlying standards like 3GPP or iPv6 on the WAN side, ZigBee or KNX on the M2M area side and a clear definition of application interfaces

will open up a complete new business opportunities for existing players and more important for new players.

- The heterogeneous standards environment is depicted in Figure 1.2 below.
- As such this horizontally integrated service layer can be seen as the operational system of the future IoT providing a set of commonly required services to a broad range of applications and underlying communication technologies.

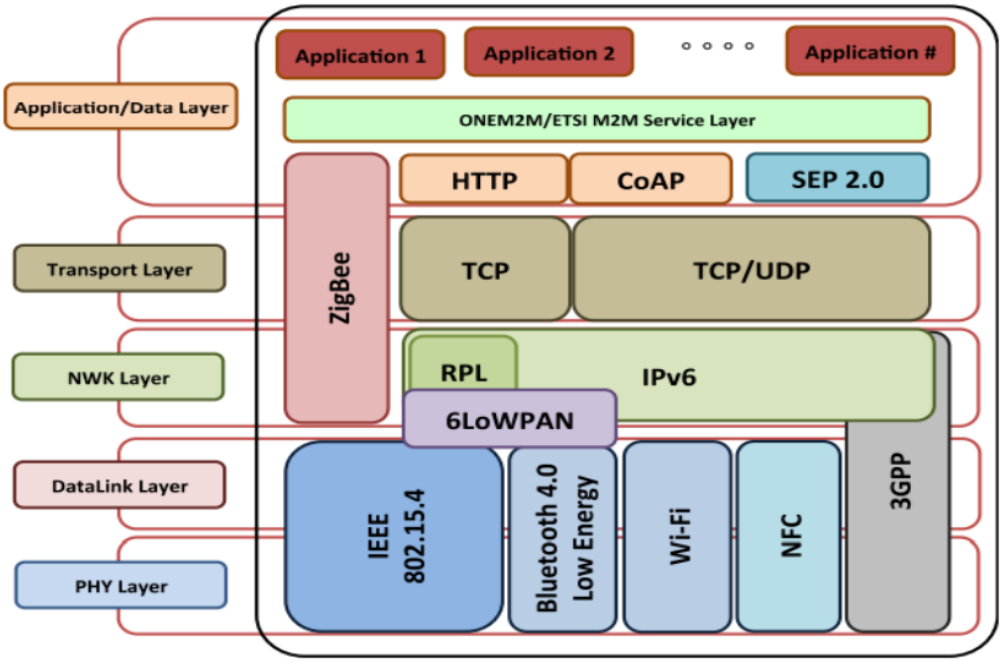


Figure 1.2 **Heterogeneous standards environment in IoT.**

Cross vertical M2M service layer standardization :

- The main tasks in the standardisation activities will be the integration of different vertical including their communication standards and the definition of clear interoperability methods.
- Here a worldwide standardised service layer for M2M type of communication will provide a framework for the integration of the different communication technologies deployed in the field of IoT.
- This M2M service layer will provide the needed services like data transport, security, devices management and device discovery [1] in a harmonized manner across a multitude of vertical domains to the application layer.
- These services will be independent from the underlying communication infrastructure and the deployed standards.

- In addition to these basic services across vertical semantic support should be included into the service layer capabilities allowing the different vertical domains to represent their semantic information in a horizontal framework.
- In recent years several standardisation activities towards a horizontal service layer approach have been started by different standardisation organizations (SDO) world-wide.
- Here the activities at TIA in the TIA-50 group (M2M Smart Device Communication) in the USA, CCSA TC10 in China and the activities in the ETSI TC M2M group in Europe should be explicitly mentioned.
- The European activities in ETSI in the scope of ETSI TC M2M can be seen as the most advanced set of horizontal M2M service layer specification with a first release of the standard at the end of 2011 and a finalization of Release 2 during 2013Figure 1.3 gives an overview over the different communication domains and objects in the ETSI M2M solution. Since the creation of oneM2M PP the ETSI technical work on the core M2M service layer will be moved to oneM2M. The scope of the ETSI M2M activity will evolve towards a broader handling and coordination of IoT related standardisation topics and the interfacing between oneM2M and the European organisations (EU Mandates, EU regulation) and EU research projects including the IERC cluster.

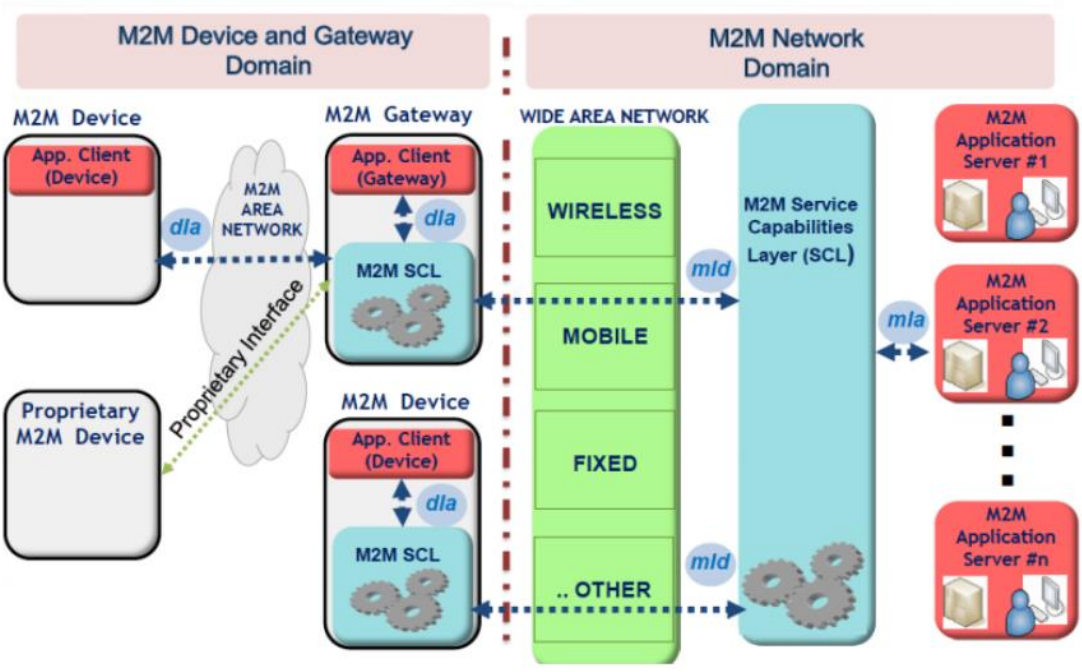


Figure 1.3 **Example: Different M2M domains and the corresponding ETSI M2M objects [source: ETSI TC M2M].**

- The CCSA (http://www.ccsa.org.cn/) in China standard defines a simple service layer with main drawback in the security and privacy domain.

- Based on these developments an operational M2M service layer called WMMP exists and is being used by China Mobile.
- This service layer has a limited capability and can be seen as light version of a service layer. The TIA-50 (http://www.tiaonline.org/) activities in USA have lead to an initial set of standards with the main focus on the devices and gateway side with a clear lack of network support.
- Just recently corresponding activities have been launched in this group.
- In 2010 the major players in the field have identified the need of a world-wide harmonized standard for the service layer for an M2M like communication.
- Based on this clear requirement the leading SDO in Europe (ETSI), USA (TIA, ATIS), China (CCSA), Korea (TTA) and Japan (TTC, ARIB) have created a world-wide partnership project called ONEM2M (http://www.onem2m.org/) which is operational and in place since September 2012.
- Participation in the partnership project is open for the individual SDO member companies and institutions.
- The participating SDOs intend to transfer all standardisation activities in the scope of M2M service layer to the ONEM2M PP and with that stop their individual activities in the domain.
- Regional tasks and adaptation of the standards toward the regional regulation will stay in the responsibility of the regional SDOs.
- In the near future the participation to the ONEM2M will be opened to other standards group and for a like the Broad Band Forum (BBF) and the Continua Health Alliance as representatives of specific vertical application domains.
- OneM2M is planning a first release of a set of standards for a service layer for the beginning of 2014.
- The requirements are based on the use cases developed in the different SDO's and will lead to a world-wide M2M service layer solution.

IoT Reference Model :

The first major contribution of the IoT Architectural Reference Model (IoT ARM) is the IoT Reference Model itself. Besides models, the IoT Reference Model provides the concepts and definitions on which IoT architectures can be built.
The IoT Reference Model aims at establishing a common grounding and a common language for IoT architectures and IoT systems. It consists of the sub-models shown in Fig. which we explain below. The yellow arrows show how concepts and aspects of one model are used as the basis for another.

IoT Domain Model :

The foundation of the IoT Reference Model is the IoT Domain Model, which introduces the main concepts of the Internet of Things like Devices, IoT Services and Virtual Entities (VE), and it also introduces relations between these concepts. The abstraction level of the IoT Domain Model has been chosen in such a way that its concepts are dependent of specific technologies and use-cases. The idea is that these concepts are not expected to change much over the next decades or longer.

IoT Information Model :

Based on the IoT Domain Model, the IoT Information Model has been developed. It defines the structure (e.g. relations, attributes) of IoT related information in an IoT system on a conceptual level without discussing how it would be represented. The information pertaining to those concepts of the IoT Domain Model is modelled, which is explicitly gathered, stored and processed in an IoT system, e.g. information about Devices, IoT Services and Virtual Entities.

IoT Functional Model :

The IoT Functional Model identifies groups of functionalities, of which most are grounded in key concepts of the IoT Domain Model. A number of these Functionality Groups (FG) build on each other, following the relations identified in the IoT Domain Model. The Functionality Groups provide the functionalities for interacting with the instances of these concepts or managing the information related to the concepts, e.g. information about Virtual Entities or descriptions of IoT Services. The functionalities of the FGs that manage information use the IoT Information Model as the basis for structuring their information.

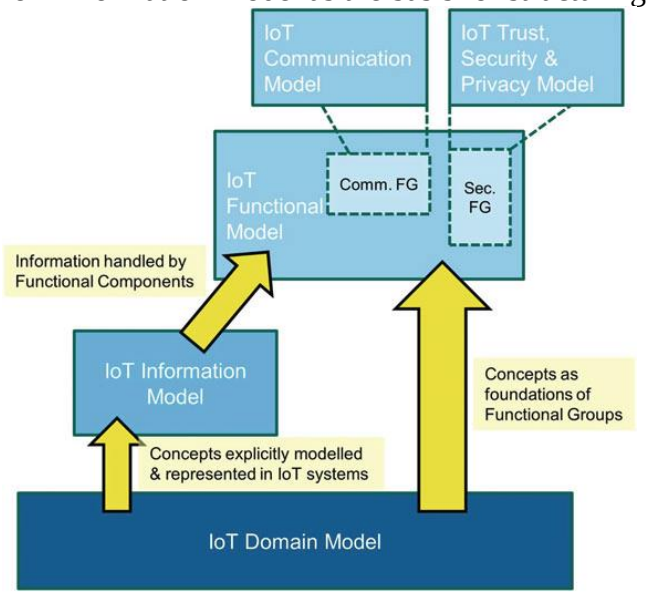


Fig.Interaction of all sub-models in the IoT Reference Model

IoT CommunicationModel :

A key functionality in any distributed computer system is the communication between the different components. One of the characteristics of IoT systems is often the heterogeneity of communication technologies employed, which often is a direct reflection of the complex needs such systems have to meet. The IoT Communication Model introduces concepts for handling the complexity of communication in heterogeneous IoT environments. Communication also constitutes one FG in the IoT Functional Model.

IoT TSP Model :

Finally, Trust, Security and Privacy (TSP) are important in typical IoT use-case

scenarios. Therefore, the relevant functionalities and their interdependencies and interactions are introduced in the IoT TSP Model. As in the case of communication, security constitutes one FG in the Functional Model.

IoT Reference Architecture:

A system architecture, and thus by default, a reference architecture, needs to answer a wide range of questions

Functional elements.
• Interactions of said elements.
• Information management.
• Operational features.
• Deployment of the system.

What the user of an architecture expects, is an architectural description, viz. “a set of artifacts that documents an architecture in a way its stakeholders can understand and that demonstrates that the architecture has met their concerns” (Rozanski 2005b). Instead of providing these artifacts in a monolithic description, one often chooses to delineate them by so-called architectural views. The idea behind so doing is to focus on system aspects that can be conceptionally isolated. Architectural views make both the derivation of the architecture and its validation easier. The above bullet-point list provides examples of such views.

Architectural Views :

Views are used during the design and implementation phase of a concrete system architecture. They are defined in the following way:
A view is a representation of one or more structural aspects of an architecture that illustrates how the architecture addresses one or more concerns held by one or more of its stakeholders (Rozanski and Woods 2011).
A view is composed of viewpoints, which aggregate several architectural concepts in order to make the work with views easier. The IEEE standard 1471 defines viewpoints as follows:
A viewpoint is a collection of patterns, templates, and conventions for constructing one type of view. It defines the stakeholders whose concerns are reflected in the viewpoint and the guidelines, principles, and template models for constructing its views
Some typical examples for viewpoints are:
• Functional view: functional-decomposition viewpoint; interaction viewpoint; interface viewpoint;
• Information view: information-hierarchy viewpoint; semantics viewpoint;information-processing viewpoint; information-flow viewpoint

Usage of Views and Perspectives in the IoT Reference

Architecture : Architecture is use-case- and application- independent and is therefore not compatible to the concept of views and viewpoints one-by-one. But the idea behind the concept is nevertheless helpful and was thus adopted for the use within the IoT Reference

Architecture. As discussed above the following views were left out from the IoT Reference Architecture.

- Physical Entity View and
- Context View.

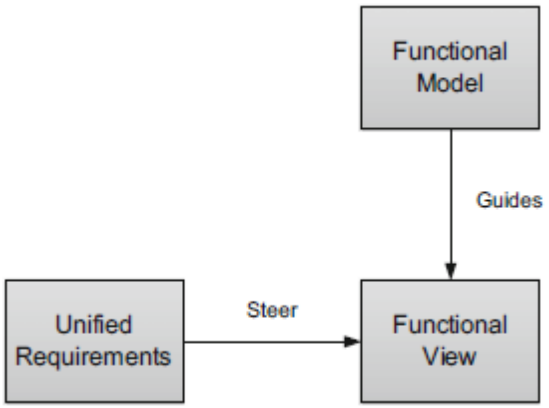
Concerning the Functional View, of the above three viewpoints, interactions are not covered in the IoT Reference Architecture, since the number of arrangements of the Functional Components and also their invocation is practically infinite. Instead, we chose to cover some typical – but yet high-level – interaction patterns.

The following sections present the IoT Functional View, IoT Information View, and the IoT Deployment and Operational view of the IoT Reference Architecture

The Reference Architecture is independent from a specific use-case or application and includes three views: (a) functional, (b) information, and (c) deployment and operation.

Functional View :

The functional view describes the function components of a system; these include components' responsibilities, default functions, interfaces, and interactions. The architecture is composed of five longitudinal functionality groups (FGs), namely service organisation, IoT process management, virtual entity, IoT services, communication, and two transversal FGs, namely management and security.

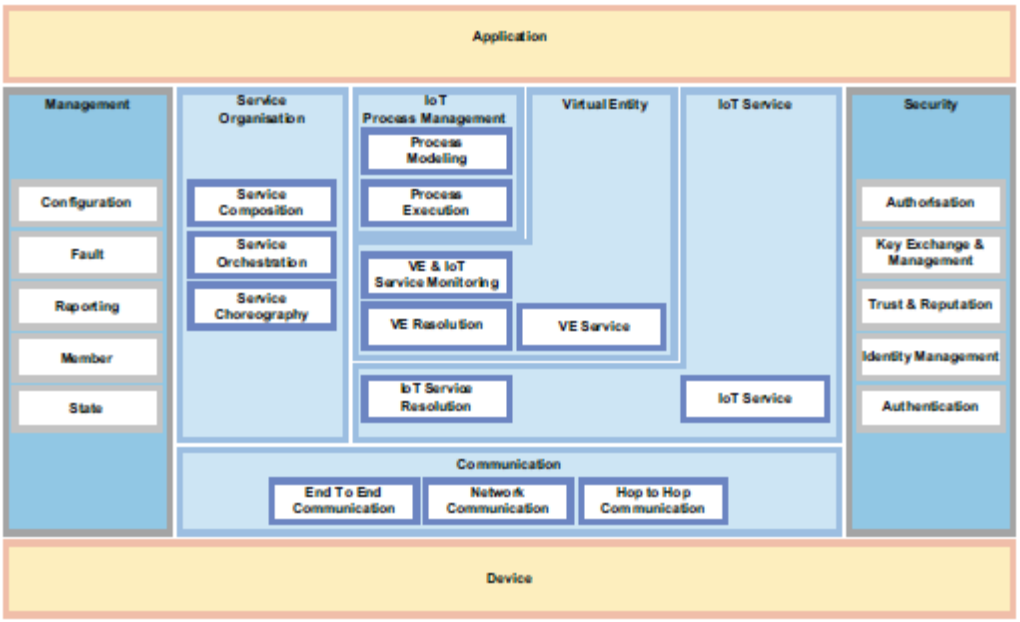


In a first step, the Unified Requirements are mapped to the different Functionality Groups of the IoT Functional Model.
Next, clusters of requirements of similar functionality are formed and a Functional Component for these requirements defined.

Functional View diagram :

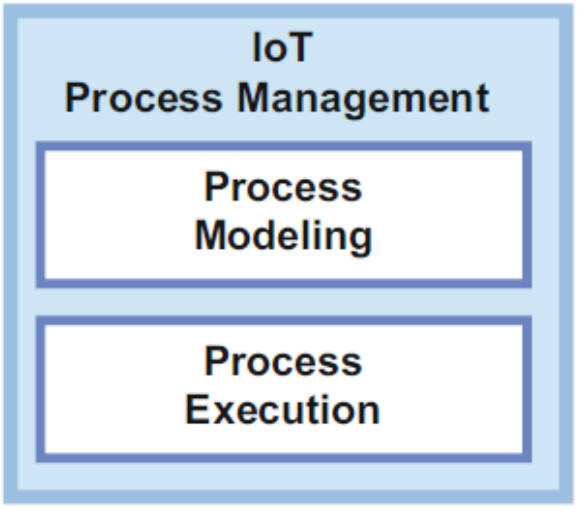
The Application FG and Device FG are out-of-scope of the IoT-A Reference Architecture and are coloured in yellow;
• Management FG and Security FG are transversal FGs and are coloured dark blue.

For each of the Functionality Groups, the Functional Components (FC) are depicted.



The Functional View presented here will give a description of the Functional Components, but will not describe the interactions taking place between the Functional Components.

IoT Process Management :



The IoT Process Management FG relates to the integration of traditional process management systems with the IoT ARM. The overall aim of the FG is to provide the functional concepts and interfaces necessary to augment traditional (business) processes with the idiosyncrasies of the IoT world.
The IoT Process Management FG consists of two Functional Components (see Figbelow):
• **Process Modelling;**

• **Process Execution**.
The Process Modelling FC provides an environment for the modelling of IoT-aware usiness processes that will be serialised and executed in the Process Execution FC. The main function of the Process Modelling FC is to provide the tools necessary for modelling processes using the standardised notation, i.e. using novel modelling concepts specifically addressing the idiosyncrasies of the IoT ecosystem .
The Process Execution FC executes IoT-aware processes that have been modelled in the Process Modelling FC described above. This execution is achieved by utilising IoT Services that are orchestrated in the Service Organisation layer. The Process Execution FC is responsible for deploying process models to the execution environments: activities of IoT-aware process models are applied to appropriate execution environments, which perform the actual process execution by finding and invoking appropriate IoT Services.

Service Organisation:

The Service Organisation FG (see Fig)is the central Functional Group that acts as a communication hub between several other Functional Groups. Since the primary concept of communication within the IoT ARM is the notion of a Service, the Service Organisation is used for composing and orchestrating Services of different levels of abstraction. The Service Organisation FG consists of three Functional Components:
• **Service Orchestration;**
• **Service Composition;**
• **Service Choreography.**

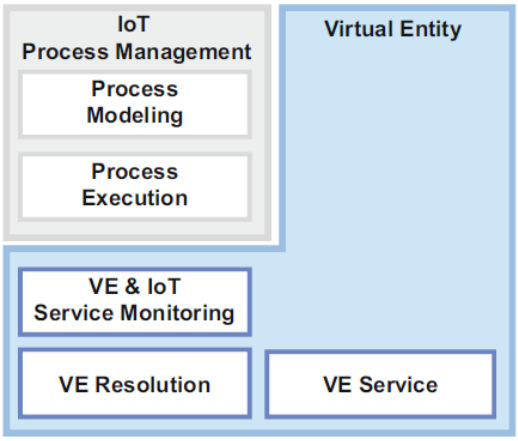
The Service Orchestration FC resolves the IoT Services that are suitable to fulfil service requests coming from the Process Execution FC or from Users. Its only function is to orchestrate IoT Services: resolve the appropriate services that are capable of handling the IoT User's request. If needed, temporary resources will be set up to store intermediate results that feed into Service Composition or complex event processing.

The Service Composition FC resolves services that are composed of IoT Services and other services in order to create services with extended functionality. The Functional Component has two main functions: (1) support flexible service compositions and (2) increase quality of information.
To support flexible service compositions, the Service Composition FC must provide dynamic resolution of complex services, composed of other services. These combinable services are chosen based on their availability and the access rights of the requesting user. Quality of information can be increased by combining information from several sources. For example,

an average value – with an intrinsically lower uncertainty – can be calculated based on the information accessed through several resources.

The Service Choreography FC offers a broker that handles Publish/Subscribe communication between services. One service can offer its capabilities at the FC and the broker function makes sure a client interested in the offer will find the service with the desired capabilities. Also service consumers can put service requests onto the Choreography FC while a suitable service is not available at the time when the request was issued. The service consumer will get notified as soon as a service became available that fulfils the service request issued before.

Virtual Entity :



The Virtual Entity FG (see Fig.) contains functions for interacting with the IoT System on the basis of VEs, as well as functionalities for discovering and looking up services that can provide information about VEs, or which allow the interaction with VEs. Furthermore, it ontains all the functionality needed for managing associations, as well as dynamically finding new associations and monitoring their validity.
The Virtual Entity FG consists of three Functional Components:
• **VE Resolution;**
• **VE & IoT Service Monitoring;**
• **VE Service.**

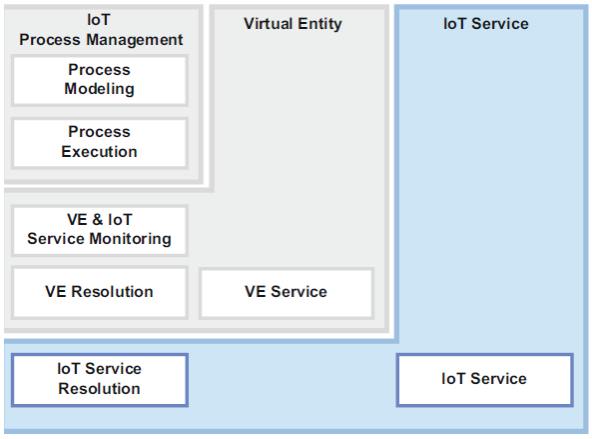
The **VE Resolution FC** is the Functional Component which provides the functionalities to the IoT User to retrieve associations between VE's and IoT Services. This includes the discovery of new and mostly dynamic associations between VE and associated services. For the discovery qualifiers, location, proximity, and other context information can be onsidered. If no association exists, the association can be created. The User can also subscribe or unsubscribe to continuous notifications about association discovery that fit a provided specification of the VE or of the Service. In case of a notification, a callback function will be called. Similar, the User can subscribe or unsubscribe to notifications about association lookup.

The **VE Resolution FC** also allows to lookup VE-related services, i.e. search for services exposing resources related to a VE. Finally, the VE Resolution FC allows managing associations: insert, delete and update associations between a VE and the IoT Services that are associated to the VE.

The **VE & IoT Service Monitoring FC** is responsible for automatically finding new associations, which are then inserted into the VE Resolution FC. New associations can be derived based on existing associations, Service Descriptions and information about VE's.

Finally, the **VE Service FC** handles with entity services. An entity service represents an overall access point to a particular entity, offering means to learn and manipulate the status of the entity. Entity services provide access to an entity via operations that enable reading and/or updating the value(s) of the entities' attributes. The type of access to a particular attribute depends on the specifics of that attribute (read only/write only or both).

IoT Service :



The IoT Service FG (see Fig. 8.6) contains IoT services as well as functionalities for discovery, look-up, and name resolution of IoT Services. It consists of two Functional

Components:
• IoT Service;
• IoT Service Resolution.

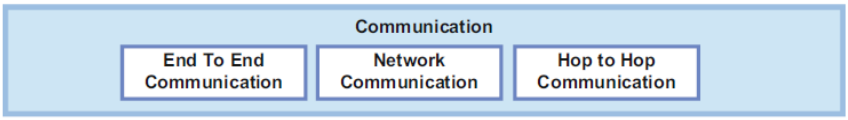
An **IoT Service** exposes one Resource to make it accessible to other parts of the IoT system. Typically, IoT Services can be used to get information provided by a resource retrieved from a sensor device or from a storage resource connected through a network. An IoT Service can also be used to deliver information to a resource in order to control actuator devices or to configure a resource. Resources can be configurable in non-functional aspects, such as dependability security (e.g. access control), resilience (e.g. availability) and performance (e.g. scalability, timeliness

A particular type of IoT Service can be the Resource history storage that provides storage capabilities for the measurements generated by resources. The main functions of the IoT Service FC are to (1) return information provided by a resource in a synchronous way, (2) accept information sent to a resource in order to store the information or to configure the resource or to control an actuator device and (3) subscribe to information, i.e. return information provided by a resource in an asynchronous way.

The **IoT Service Resolution FC** provides all the functionalities needed by the user in order to find and be able to contact IoT Services. The IoT Service Resolution also gives services

the capability to manage their service descriptions (typically stored in a database as one entry), so they can be looked up and discovered by the user. The user can be either a Human User or a software component.

Communication FG



The Communication FG (see Fig. 8.7 below) is an abstraction, modelling the variety of interaction schemes derived from the many technologies belonging to IoT systems and providing a common interface to the IoT Service FG.
The Communication FG consists of three functional components:
• **Hop To Hop Communication;**
• **Network Communication;**
• **End To End Communication**.
The Hop To Hop Communication FC provides the first layer of abstraction from the device's physical communication technology. The functional component is an abstraction to enable the usage and the configuration of any different link layer technology. Its main functions are to transmit a frame from the Network Communication FC to the Hop To Hop Communication FC and from a Device to the Hop To Hop Communication FC. The arguments for the frame transmission can be set; examples of arguments include: reliability, integrity, encryption and access control.

The Network Communication FC takes care of enabling communication between networks through Locators (addressing) and ID Resolution. The FC includes routing, which enables linking different network address spaces. Moreover different network technologies can be converged through network protocol translations. The functions of the Network Communication FC are to transmit a packet from the Hop To Hop Communication FC to the Network Communication FC and from the End To End Communication FC to the Network Communication FC. The arguments for the packet transmission can be configured and examples of arguments include: reliability, integrity, encryption, unicast/multicast addressing and access control. The Network Communication FC enables as well network protocol translation where it allows translating between different network protocols. Examples would be to translate IPv4 to IPv6 and ID to IPv4. Note that this function is necessary to implement a Gateway. Finally, the Network Communication FC can manage the packet queue and setup the size and priorities of the input and output packet queues. This function can be leveraged in order to achieve QoS

The End To End Communication FC takes care of the whole end-to-end communication abstraction, meaning that it takes care of reliable transfer, transport and, translation functionalities, proxies/gateways support and of tuning configuration parameters when the communication crosses different networking environments. The End To End Communication FC is responsible to transmit a message from the Network Communication FC to the End To End Communication FC and from (IoT) Service to the End To End Communication FC. The arguments for the message can be configured and examples include: reliability, integrity, encryption, access control and multiplexing. A second function of the End To End Communication FC is to cache and proxy. The Cache and Proxy function allows to buffer messages in the End To End Communication FC.

Security :

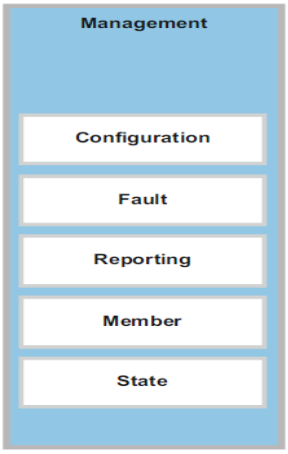
The Security FG is responsible for ensuring the security and privacy of IoT-A-compliant systems.

It consists of five functional components:
• Authorisation;
• Key Exchange & Management;
• Trust & Reputation;
• Identity Management;
• Authentication.

The **Authorization FC** is a front end for managing policies and performing access control decisions based on access control policies. This access control decision can be called whenever access to a restricted resource is requested. For example, this function is called inside the IoT Service Resolution FC, to check if a user is allowed to perform a lookup on the requested resource. This is an important part of the privacy protection mechanisms The two default functionalities offered by the Authorization FC are firstly, to determine whether an action is authorized or not. The decision is made based on the information provided from the assertion, service description and action type. Second functionality is to manage policies, such as adding, updating or deleting an access policy.
The **Authentication FC** is involved in user and service authentication. It checks the credentials provided by a user, and, if valid, it returns an assertion as result, which is required to use the IoT Service Client. Upon checking the correctness of the credentials supplied by a newly joining node, it establishes secured contexts between this node and various entities in its local environment. The two functionalities provided by the Authentication FC are (1) to authenticate a user based on provided credential and (2) to verify whether an assertion provided by a user is valid or invalid.
The **Identity Management FC** addresses privacy questions by issuing and managing pseudonyms and accessory information to trusted subjects so that they can operate (use or provide services) anonymously. Only one default function is attributed to this FC: to create a fictional identity (root identity, secondary identity, pseudonym or group identity) along with the related security credentials for users and services to use during the authentication process.
The **Key Exchange and Management (KEM) FC** is involved to enable secure communications between two or more IoT-A peers that do not have initial knowledge

of each other or whose interoperability is not guaranteed, ensuring integrity and confidentiality.
The **Trust and Reputation Architecture FC** collects user reputation scores and calculates service trust levels.

Management :



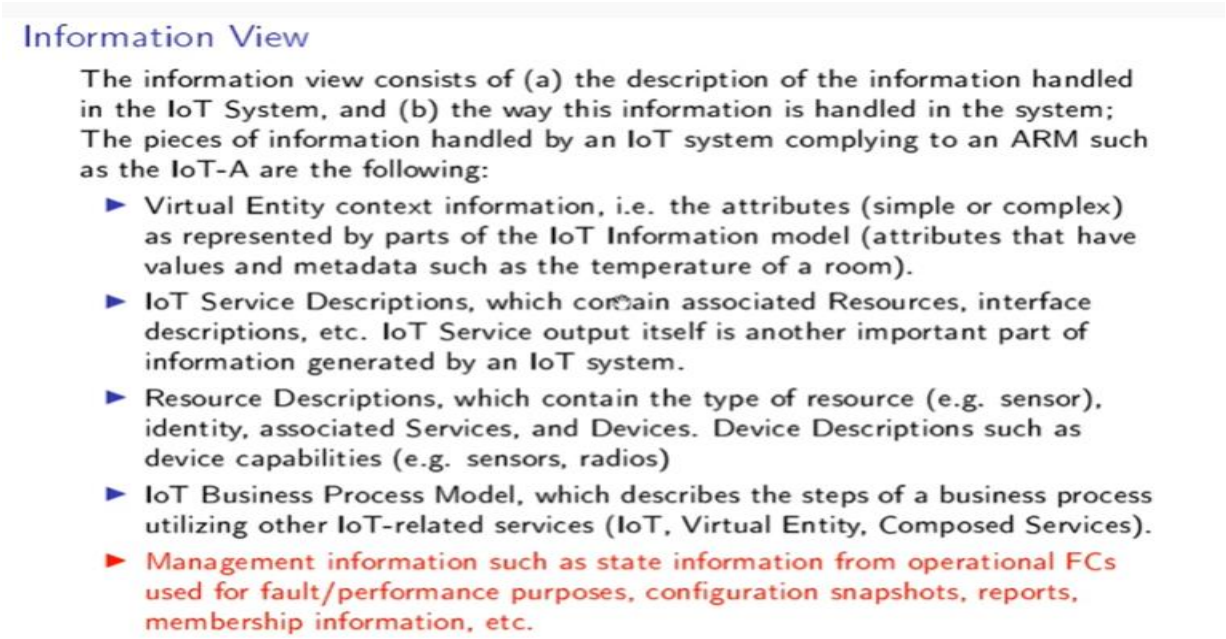
The Management FG (see Fig. 8.9) consists of five Functional Components:
• Configuration;
• Fault;
• Reporting;
• Member;
• State

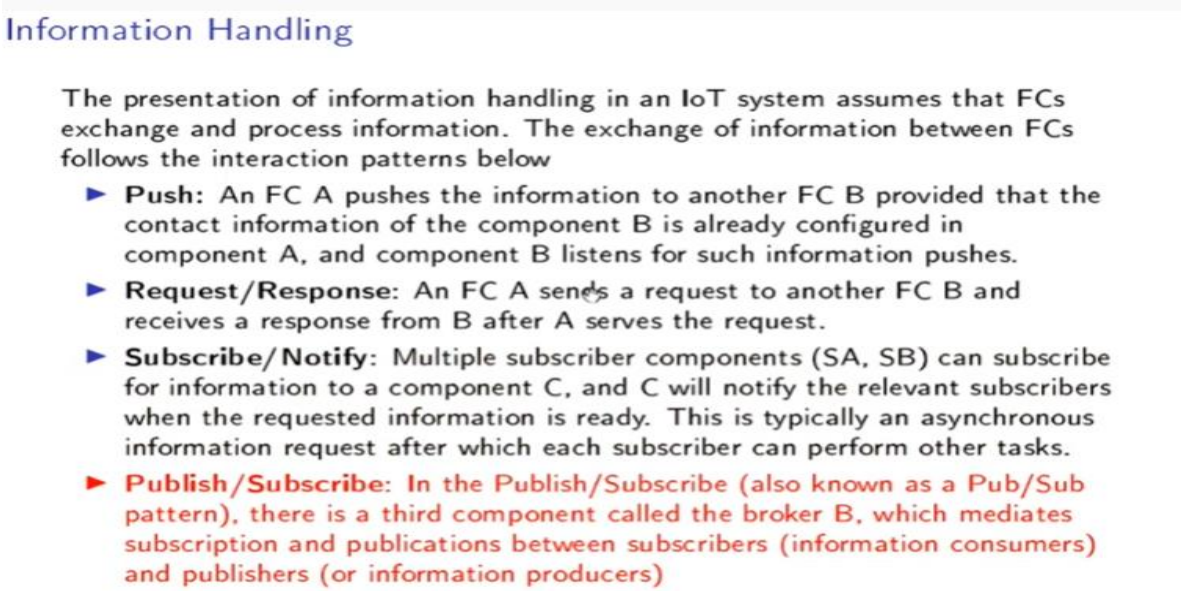
The **Configuration FC** is responsible for initialising the system configuration such as gathering and storing configuration from FC's and Devices. It is also responsible for tracking configuration changes and planning for future extension of the system.
The goal of the **Fault FC** is to identify, isolate, correct and log faults that occur in the IoT system. When a fault occurs, the respective functional component notifies the Fault FC. Such notification triggers, for instance, are the gathering of more data in order to identify the nature and severity of the problem.
The **Member FC** is responsible for the management of the membership and associated information of any relevant entity (FG, FC, VE, IoT Service, Device, Application, User) to an IoT system
The **Reporting FC** can be seen as an overlay for the other Management FCs. It distils information provided by them. One of many conceivable reporting goals is to determine the efficiency of the current system. This is important since by "collecting and analysing performance data, the [system] health can be monitored.
The **State FC** monitors and predicts state of the IoT system. For a ready diagnostic of the system, as required by Fault FC, the past, current and predicted (future) state of the system are provided. This functionality can also support billing.

Information View :

The information view covers the information life cycle in the IoT system, providing an overview of the information structures and flows (i.e. how information is defined,

structured, exchanged, processed, and stored), and the list of the components involved in the process.





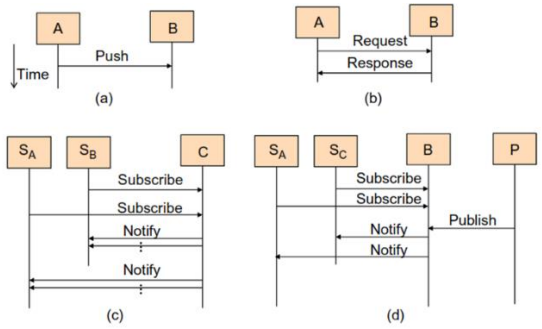
Information Description :

The pieces of information handled by an IoT system

- Virtual Entity context information, i.e. the attributes (simple or complex) as represented by parts of the IoT Information model.

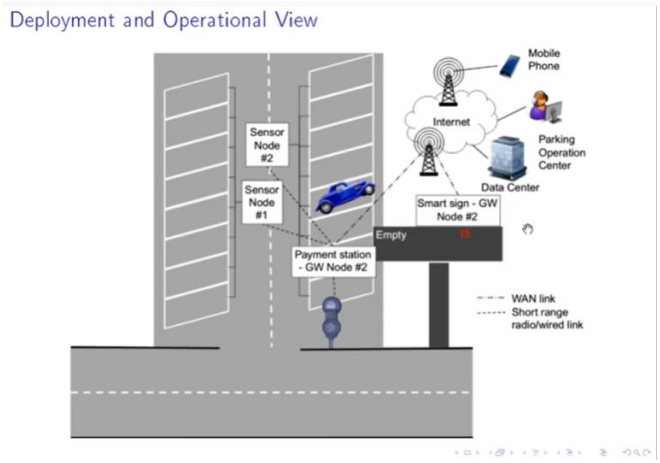
- IoT Service output itself is another important part of information generated by an IoT system. For example, Sensor or a Tag Service.
- Virtual Entity descriptions in general, which contain not only the attributes coming from IoT Devices (e.g. ownership information).
- Virtual Entity Associations with other Virtual Entities (e.g. Room #12 is on floor#7)
- **Resource Descriptions –>** type of resource (e.g. sensor), identity, associated Services, and Devices.
- **Device Descriptions –>** device capabilities (e.g. sensors, radios).
- **Descriptions of Composed Services –>** the model of how a complex service is composed of simpler services.
- **IoT Business Process Model describes –>** the steps of a business process utilizing other IoT-related services.
- Management information such as state information from operational FCs used for fault/performance purposes, configuration snapshots, reports, membership information, etc.

IOT Information handling:



Deployment and Operational view:

Lastly, the deployment and operation view has an important role in the realisation of IoT systems as they are bringing together a number of devices, each of which has different resources and connection interfaces, which can be interconnected in numerous ways. The deployment and operation view provides a set of guidelines for system design, covering different aspects of technologies, communication protocols, services, resources, and information storage.



Deployment and the operational view are very important in addressing how the actual system can be realized by selecting technologies and making them communicate and operate in a comprehensive.

Let's take an example of the parking lot system

As we can see in the figure, there are two sensor nodes #1 and #2, each of which is connected to eight car presence sensors.

- They are also connected to the payment stations through wireless or wired communication.

- The payment station acts both as a user interface for the device to pay and get a payment receipt as well as a communication gateway that connects the two sensor nodes and payment interface physical devices with the internet through WAN.

- The occupation sign also acts as a communication gateway for the actuator node, and we assume that because of the deployment, a direct connection to the payment station is not feasible.

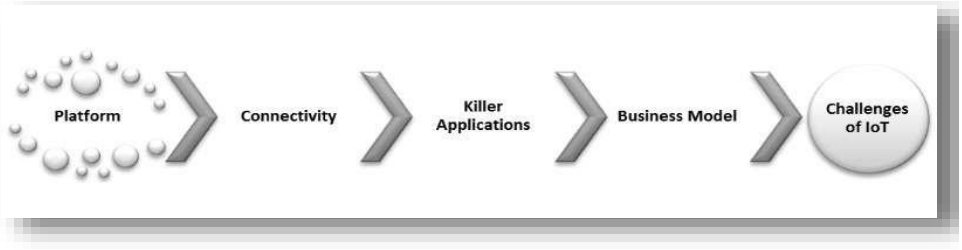
- The physical gateway devices connect through WAN to the internet and towards a data center where the parking lot management system software is hosted as one of the virtual machines on a platform as service configurations.

- The two main applications connected to this management system are human user mobile phone applications and parking operation center applications.

Wireless Technologies for IOT:
Protocol Standardization for IOT, M2M, RFID
Standardization is one of the most critical hurdles of the IoT evolution. ... The IoT promises billions of connected things which in turn require common standards in order to operate with an acceptable, manageable and scalable level of complexity

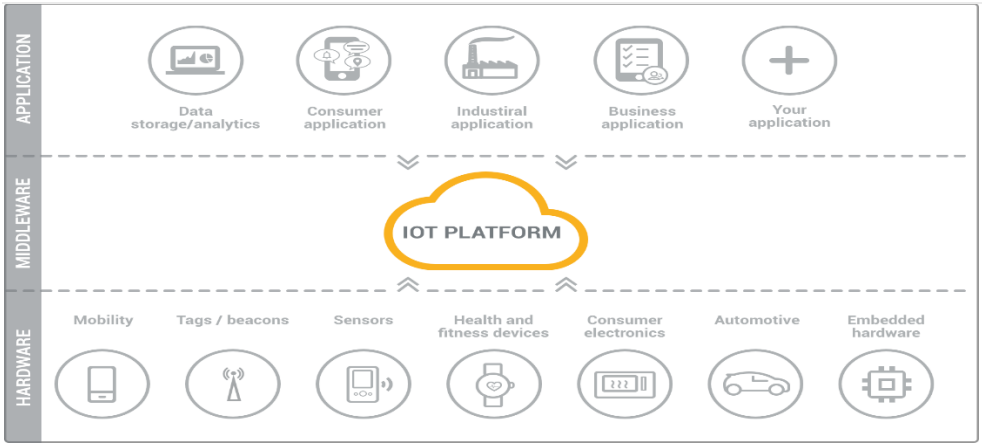
The hurdles facing IoT standardization can be divided into 4 categories;

- Platform
- Connectivity
- Business Model
- Killer Applications



IOT PLATFORM : This part includes the form and design of the products (UI/UX), analytics tools used to deal with the massive data streaming from all products in a secure way, and scalability which means wide adoption of protocols like IPv6 in all vertical and horizontal markets is needed.

IoT platform	Services
Particle	Hardware, Connectivity, Device **Cloud**, and Apps.
Salesforce IoT Cloud	Data from customers, partners, devices, and sensors.
ThingWorx	End-to-end Industrial IoT platform.
IBM Watson IoT	Connection Service, Analytics Service, Blockchain Service.



Connectivity :

IoT connectivity is a term defining connection between all the points in the **IoT** ecosystem, such as sensors, gateways, routers, applications, platforms and other systems. It usually refers to different types of network solutions based on their power consumption, range and bandwidth consumption.

This phase includes all parts of the consumer's day and night routine, from using wearables, smart cars, smart homes, and in the big scheme, smart cities. From the business prospective we have connectivity using IIoT (Industrial Internet of Things) where M2M communications dominating the field.

Business Model :
The bottom line is a big motivation for starting, investing in, and operating any business, without a sound and solid business models for IoT we will have another bubble , this model must satisfied all the requirements for all kinds of e-commerce; vertical markets, horizontal markets and consumer markets. But this category is always a victim of regulatory and legal scrutiny

Killer Applications :

In this category there are three functions needed to have killer applications: control "things", collect "data", and analyze "data". IoT needs killer applications to drive the business model using a unified platform.

IoT Business Model #1: Subscription Model
Now instead of having a one-time sale, you can offer a subscription model in which your customer pays a fee in return for continuous value.
IoT Business Model #2: Outcome-Based Model
The outcome-based IoT business model is an example of an innovative approach enabled by IoT products.
IoT Business Model #3: Asset-Sharing Model
IoT has the potential to solve this problem, and we are already starting to see solutions with self-driving cars, virtual power plants, shared drones
IoT Business Model #4: Monetize Your IoT Data:facebook,twitter

All four categories are inter-related, you need all them to make all them work. Missing one will break that model and stall the standardization process. A lot of work needed in this process, and many companies are involved in each of one of the categories, bringing them to the table to agree on a unifying model will be daunting task.

Major Differences Between WoT Over IoT :

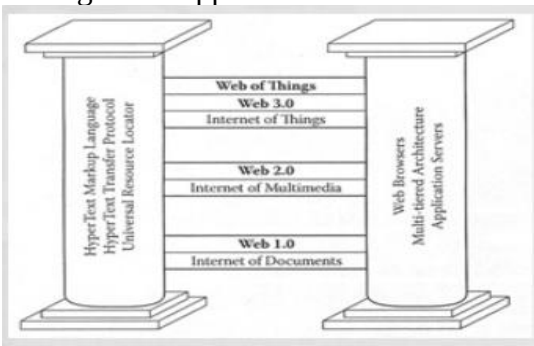
- aggregation of already available technologies.
- It is simply concept of basically controlling different devices by establishing connection and communicating from mobile app or web browser.
- In Iot it is very difficult to establish a effective communication between different machines of different vendors.
- And due to this limitations in **IoT**, **WoT** comes in pictures.
- The **Web of Things** (WoT) is a term used to describe approaches, software architectural styles and programming patterns that allow real-world objects to be part of the World Wide Web.
- From developers perspective , the WoT enables access and control over IoT resources and applications using mainstream web technologies (such as HTML 5.0 , JavaScript , Ajax , PHP , Ruby n Rails , etc)
- The approach to building WoT is therefore based on RESTful principles and REST API s, which enable s both developers and deployers to benefit from the popularity and maturity of web technologies.
- building the WoT has various scalability security etc challenges especially as part of a roadmap towards a global WoT.

- While IoT is about creating a network of objects, things , people, system and applications ,WoT tries to integrate them to Web.

- Technically speaking WoT can be thought as flavor/Option of an application layer added over the IoT‘s network layer .

Two pillars of web :

- The application server became the oundation that helped build widely spreading web-based applications
- An application server acts as a set of components accessible to the software developer through an API defined by the middle ware itself
- The application server is based on the three tiered or multi tired software architecture
- As the teo pillars for web applications and internet revolution ,the protocols HTML/HTTP/URL and software will continue to be the two pillars of play an important role in building WOT applications**.**



First pillar : Protocols
Second pillar : Software
Protocols include : HTML,HTTP,URL
Software include : Web browsers, multi tired architecture and application servers

Protocol Standardization for IoT :

IoT-Architecture one of the few efforts targeting a holistic architecture for all IoT sectorsThis consortium consists of 17 European organizations from nine countries.Summarized current status of IoT standardization as

- Fragmented architectures,no coherent unifying concepts,solutions exist only for applications silos
- No holistic approach to implement IoT has yet been proposed
- Many island solutions do exist (RFID, sensor nets, etc.)
- Little cross-sector reuse of technology and exchange of knowledge

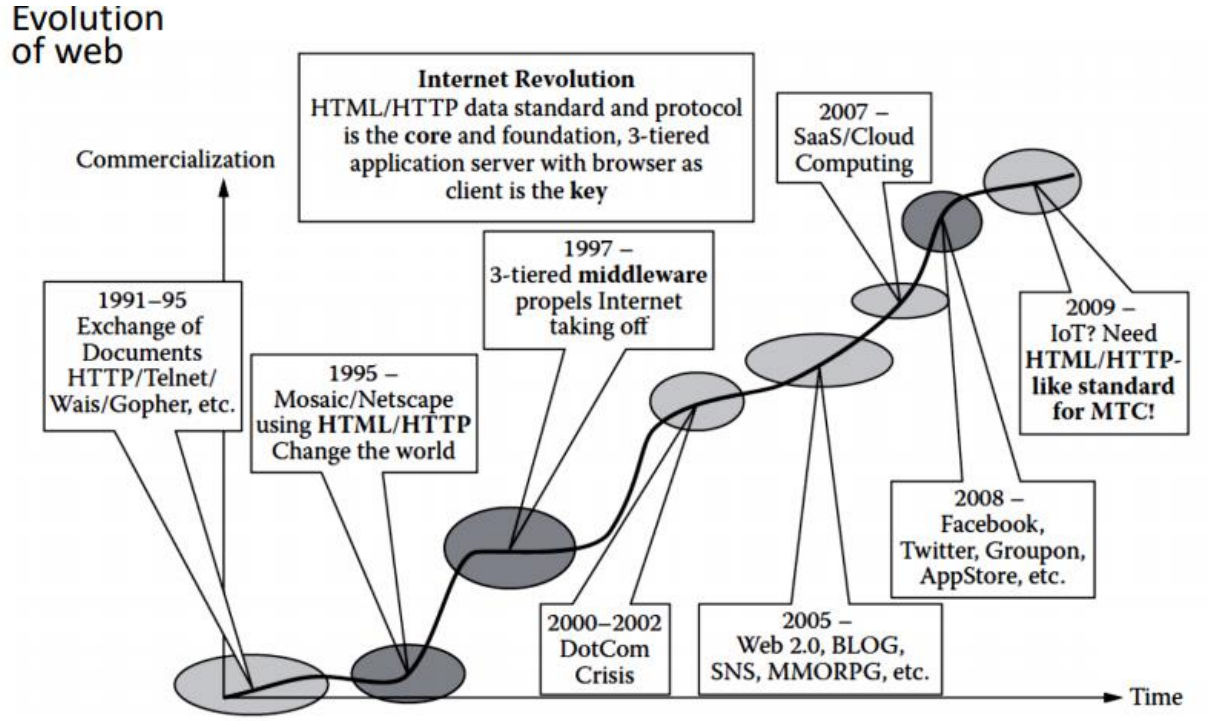
The key objectives of IOT-A consortium are as follows ;

- Create the architectural foundations of an interoperable internet of things as a key dimension of the larger future internet
- Try to use alreafy existing technologies
- Demonstrating the applicability in a set of use cases

- Removing the barriers of deployment and wide –scale acceptance of the IOT by establishing a strongly involved stakeholder group
- Federating heterogeneous IOT technologies into an interoperable IOT fabric

Unified Data Standards :

- Already discussed about two pillars of the Internet
- HTML/HTTP combination of data format and exchange protocol is the foundation pillar of WWW
- Described great number of data standards and protocols proposed for four pillar domains of IoT
- Many issues still impede the development of IoT and especially WoT vision



Unified data standards in IOT for data Exchange :

- **Common set of standards** for internet of things.
- A message translation to be replaced with **a unified message structure** to communicate among embedded, gateway, and cloud machines.

- IoT landscape is **very vast** in nature. It comprises industrial, energy, agriculture, healthcare, supply chain, automotive, smart city and many more.
- So, it would be **difficult to create a unified standard for the industry** as a whole.

Protocols –IEEE 802.15.4 :

- Defines operation of low-rate wireless personal area networks (LR-WPANs)
- Specifies physical layer and media access control for LR-WPANs
- Maintained by IEEE 802.15 working group, which defined the standard in 2003
- Basic framework conceives a 10m communications range with a transfer rate of 250 kbit/s

M2M & WSN Protocol :

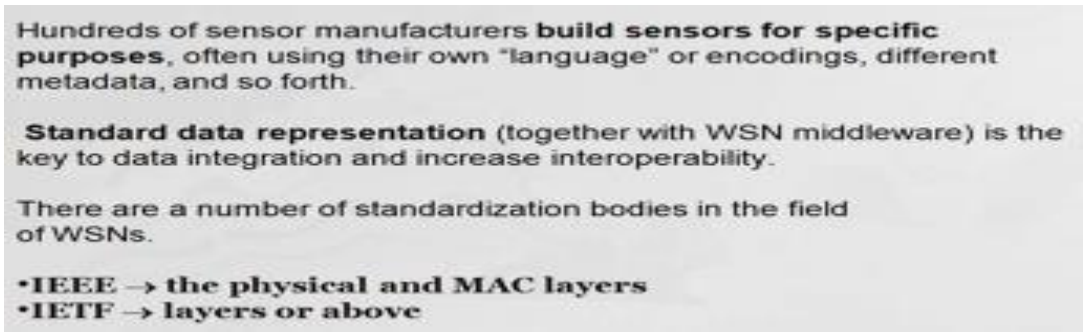
- Most M2M applications are developed today in a highly customized fashion
- High-level M2M architecture from M2M Standardization Task Force (MSTF) does include fixed & other non cellular wireless networks
- Means it's generic, holistic IoT architecture even though it is M2M architecture
- M2M and IoT sometimes are used interchangeably in the United States
- Other M2M standards activities include:
    - Data transport protocol standards - M2MXML, JavaScript Object Notation (JSON), BiTXML, WMMP, MDMP
    - Extend OMA DM to support M2M devices protocol management objects
    - M2M device management, standardize M2M gateway
    - M2M security and fraud detection
    - Network API's M2M service capabilities
    - Remote management of device behind gateway/firewall
    - Open REST-based API for M2M applications

M2M Protocol :

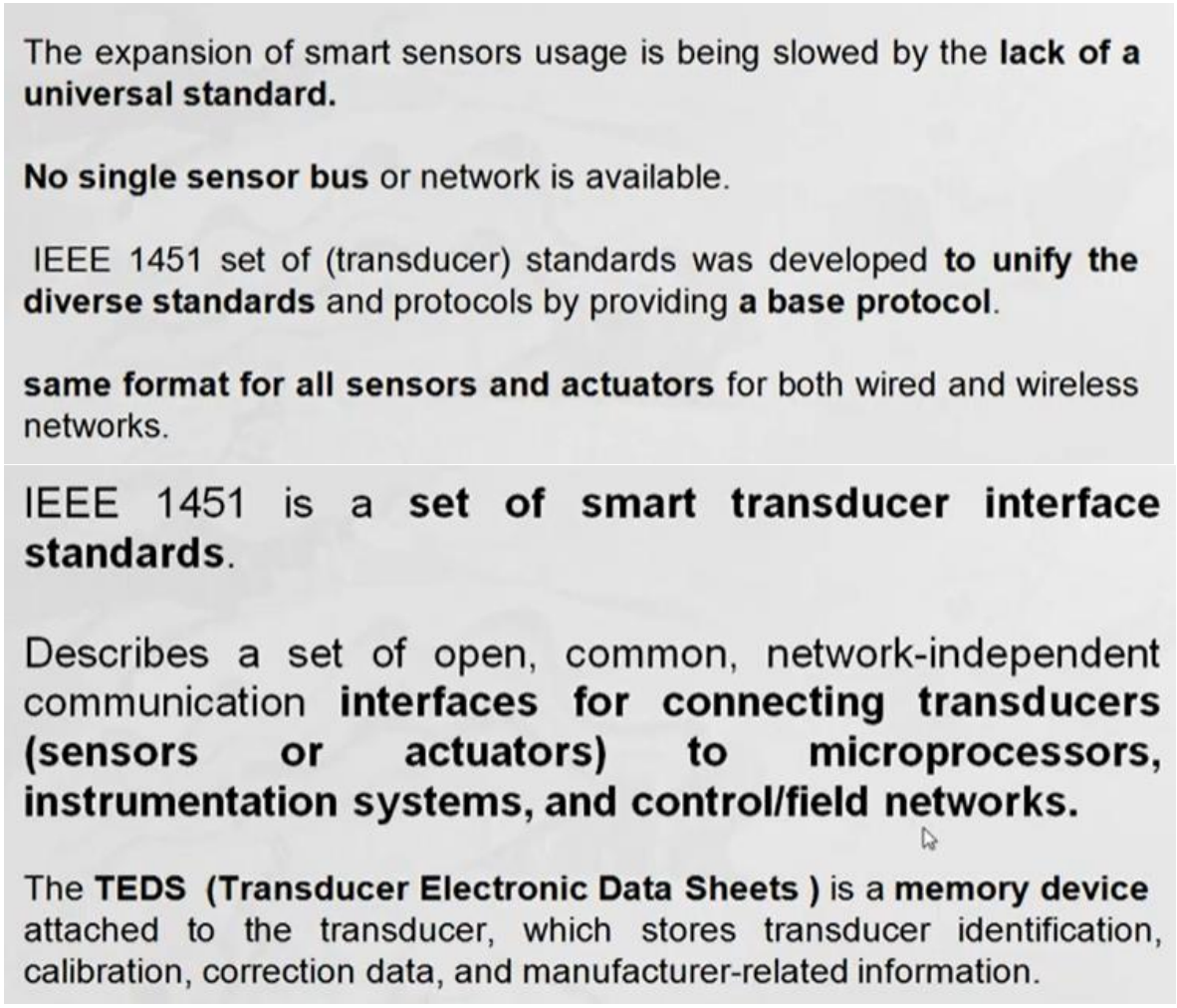
Most M2M applications are developed today in a highly customozed fashion and vertical specific industry.

A broad horizontal standard is a key requirement for the M2M industry to move from its current state of applications based on its own technology to a truly interconnected internet of things.
M2M standardization Task Force (MSTF) has been created for the same.

WSN Protocol standards :



WSN :IEEE 1451 Standard :



The IEEE 1451 family of standards includes :

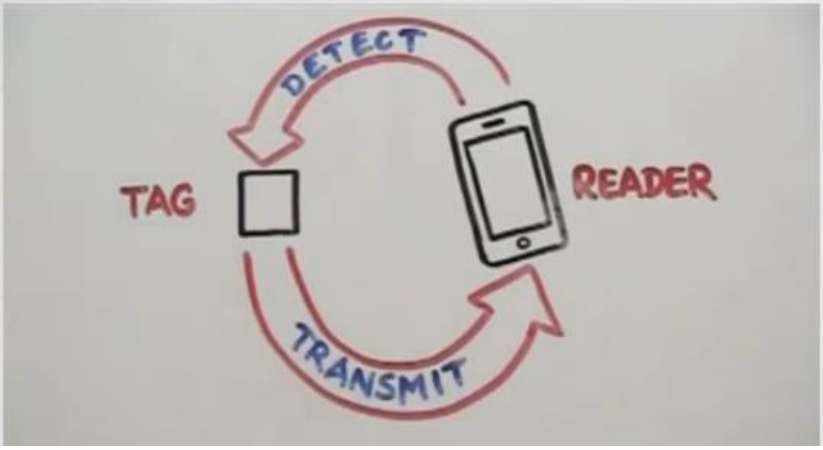
- **1451.0-2007 Common Functions**, Communication Protocols, and TEDS Formats
- **1451.1-1999 Network Capable** Application Processor information Model
- **1451.2-1997 Transducer to Microprocessor** Communication Protocols & TEDS Formats
- **1451.3-2003** Digital Communication & TEDS Formats for **Distributed Multi-drop Systems**
- **1451.4-2004 Mixed-mode** Communication Protocols & TEDS Formats
- **1451.5-2007 Wireless Communication** Protocols & TEDS Formats
- **1451.7-2010 Transducers to** Radio Frequency Identification (**RFID**) Systems.

RFID Protocol :

6.2.2 SCADA and RFID Protocols

- The Object Linking and Embedding for Process Control (OPC) Foundation is an industry consortium that creates and maintains standards for open connectivity of industrial automation devices and systems
- The OPC standards specify the communication of industrial process data, alarms and events, historical data and batch process data between sensors, instruments, controllers, software systems, and notification devices.

- The smart cards with contactless interfaces (RFID is a subset) are becoming increasingly popular for payment and ticketing applications.
- The standard for contactless smart card communications is ISO/IEC 14443.



The processing is now distributed
Functions that used to be done at control center can now be done by IED i.e. M2M between devices

Due to restructuring of electric industry, traditional vertically integrated electric utilities are replaced by many entities such as

- GENCO (Generation Company),
- TRANSCO (Transmission Company),
- DISCO (Distribution Company),
- ISO (Independent System Operator), etc.

NFC Standadization :

NFC technology also in comparison to other wireless technologies. NFC is a short range wireless technology with only a few centimeters of communication distance. It's based on existing 13.56 megahertz RFID contactless card standards, which are established for several years and are used for payment, ticketing, electronic passport, access control, among many other applications.

Data rates start at 106 kilobits per second and go up to 424 kilobits per second. A few NFC devices are already capable of supporting up to 848 kilobits per second, which is now being considered for inclusion in the NFC Forum specifications. Compared to other wireless communication technologies, NFC is designed for proximity or short range communication, which provides a dedicated read zone and some inherent security. The frequency band is 13.56 megahertz, therefore, it falls into the ISM band, which is available worldwide.

Speaking further to the technology, it is a bidirectional communication, meaning you can exchange data in both directions with a typical range of four to 10 centimeters depending on the antenna geometry and the o utput power. With a smartphone, you normally will get less range than with a dedicated reader device due t o a lower output power.

The NFC specifications are defined by an industry organization called the NFC Forum, which has nearly 2 00 member companies. It has been formed in 2004 to advance the use of near field communication technol ogy by developing specifications and sharing interoperability among devices and services, and educating t he market about NFC technology.

- Short range wireless technology, only a few centimeters
- Based on existing contactless cards standards (RFID)
- Data-Rates: 106-424Kbps (848Kbps)
- Frequency-Band: 13.56MHz (ISM Band)

Short-range radio technology that enables **bi-directional short range** communication between devices.

Communication distance is **4 cm (max. 10 cm)** depending on the antenna geometry and output power.

Being defined in the **NFC Forum** group http://www.nfc-forum.org

Based on the existing 13.56 MHz RFID standards.
ISO 14443 A/B,
JIS-X 6319-4
(ISO15693 – integration under development)

NFC Forum specifications for **extended functionality** and **interoperability** are standardized by ECMA, ETSI and ISO/IEC groups
NFCIP-1 Standard (ECMA-340, ISO/IEC 18092 , ETSI TS 102 190)
NFCIP-2 Standard (ECMA-352, ISO/IEC 21481, ETSI TS 102 312)

Smartphones in the market are already supporting the ISO 15693 protocol. These NFC specifications, and especially the specifications for the extended NFC functionalities are again, standardized by the internation al standard organizations like ISO/IEC, ECMA, and ETSI.

Coming now to the operating modes of NFC, the NFC Forum has extended the functionality of a traditiona l RFID system, and has defined three operating modes for NFC devices. Reader/writer, card emulation, and peer-to-
peer. This is a new definition of the functionality compared to legacy RFID standards. With a legacy RFID system, you always have a reader/writer device on one side and a transponder on the other. The reader/wri ter generates the RF field and the transponder communicates back to the reader/writer using load modulati on.

In brief there are three operating modes as shown :

The requirement for NFC is that the NFC device in the smartphone can change the operating mode dependi ng on the use case and application requirements and also, that the one smartphone can communicate directl y with another smartphone. So the peer-to-
peer mode has been added where two NFC devices can talk to each other directly, and either device can ini tiate the communication.

In the reader/writer mode, smartphones can read or write the memory of NFC transponders and can also co mmunicate with smartphones which are working in card emulation. And in card emulation mode, the NFC device is able to behave like a contactless card or an NFC transponder.

UNIT IV

IOT Physical Devices

Introduction to different IoT tools

Eclipse IoT

Eclipse IoT is an open-source platform that allows IoT developers and IoT development companies to develop applications in Java. With the help of Eclipse IoT, you can build IoT Devices, Cloud Platforms, and Gateways. This tool focuses on the development, adoption, and promotion of open-source IoT technologies.

These IoT protocols, application frameworks and services, and tools are promoted as the best-suited programming language for IoT using Lua programming language.

2. Node-RED

Node-RED is a simple and open-source visualization tool built on Node.JS which is used to connect the devices, services, and APIs together for the Internet of Things. Node-RED is a user-friendly interface, developed by IBM's Emerging Technology department, allowing you to connect, hardware, an API or an online service with tight integration. It helps you connect the devices easily and quickly. It helps deal with the flow of the data, integrates with APIs, services and any devices.

3.Tessel 2

Tessel 2 is a robust IoT platform that is used to build basic IoT solutions and prototypes. It integrates additional sensors and modules. This board has a capacity to hold up to a dozen modules including RFID, GPS, camera, and accelerometer.

This Tessel is very easier for those developers who are familiar with Node.JS programming. This way, Tessel can be used to host several servers and hardware firmware IoT solutions. You can leverage all the libraries of Node.JS to unveil a host of devices in minutes with Tessel.

4. Arduino

Arduino is an open-source prototyping platform offering both IoT hardware and software. Arduino is a hardware specification that can be applied to interactive electronics and a set of software which includes the Integrate Development Environment (IDE) and the Arduino programming language. It's one of the most preferable IDEs in all IoT development tools which is easy and simple to use.

Arduino can be your first choice if you are planning to build a computer that can sense and control more of the physical world when compared to your normal stand-alone computing device.

5. Kinoma Create

Kinoma Create is a device that allows establishing a connection between two devices without having too high programming knowledge in JavaScript. Kinoma Create consists of many features

that are required for developing small IoT applications like connecting light, temperature or movement sensors for a specific purpose with mobile notifications in case of any alterations.

Kinoma Create has some fundamental components which are as follows:

- A touch screen.
- An ARM SoC 800 MHz processor.
- Bluetooth and integrated WiFi.
- Several ports to connect peripherals, including a USB 2.0 port.
- A memory of 128 MB and flash memory of 16 MB.
- MicroSD card slot.
- Speaker and microphone features.
- Linux distribution.

6. Device Hive

Based on Data Art's AllJyone, Device Hive is a free open-source Machine to Machine (M2M) communication framework. Launched in 2012, it's considered as one of the most preferred platforms for IoT app development. Since Device Hive is a cloud-based API, you can control remotely without having network configuration. A similar thing applies to the libraries, portals and management protocols.

7. Home Assistant

Home Assistant is an open-source tool that is used for home automation and functions with a Python-based coding system. Mobile or desktop browser can easily have their control on the IoT system developed with this tool. It's very easy to set up and is trusted for operations, security, and privacy. The software supports any systems which are running on Python 3.

IOT Physical Devices

What is an IoT Device

- A "Thing" in Internet of Things (IoT) can be any object that has a unique identifier and which can send/receive data (including user data) over a network (e.g., smart phone, smart TV, computer, refrigerator, car, etc.).
- IoT devices are connected to the Internet and send information about themselves or about their surroundings (e.g. information sensed by the connected sensors) over a network (to other devices or servers/storage) or allow actuation upon the physical entities/environment around them remotely.

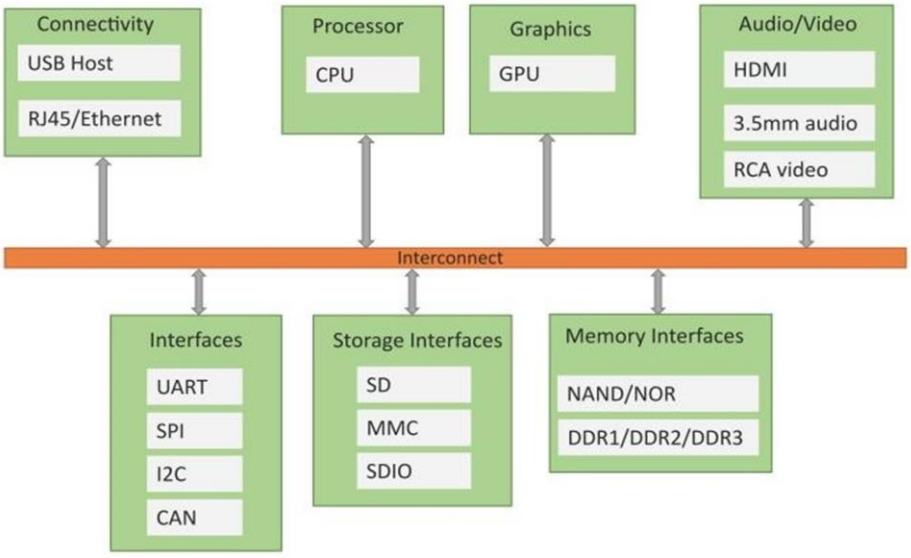
IoT Device Examples

- A home automation device that allows remotely monitoring the status of appliances and controlling the appliances.
- An industrial machine which sends information abouts its operation and health monitoring data to a server.
- A car which sends information about its location to a cloud-based service.

- A wireless-enabled wearable device that measures data about a person such as the number of steps walked and sends the data to a cloud-based service.

Basic building blocks of an IoT Device

- Sensing
    - Sensors can be either on-board the IoT device or attached to the device.
- Actuation
    - IoT devices can have various types of actuators attached that allow taking
    - actions upon the physical entities in the vicinity of the device.
- Communication
    - Communication modules are responsible for sending collected data to other devices or cloud-based servers/storage and receiving data from other devices and commands from remote applications.
- Analysis & Processing
    - Analysis and processing modules are responsible for making sense of the collected data.



Block diagram of an IoT Device

Stages of IoT Solutions Architecture

- Sensors/Actuators: Sensors or Actuators are the devices that are able to emit, accept and process data over the network. These sensors or actuators may be connected either through wired or wireless. This contains GPS, Electrochemical, Gyroscope, RFID, etc. Most of the sensors need connectivity through sensors gateways. The connection of sensors or actuators can be through a Local Area Network (LAN) or Personal Area Network.

- Gateways and Data Acquisition: As the large numbers of data are produced by this sensors and actuators need the high-speed Gateways and Networks to transfer the data. This network can be of type Local Area Network (LAN such as WiFi, Ethernet, etc.), Wide Area Network (WAN such as GSM, 5G, etc.).

- Edge IT: Edge in the IoT Architecture is the hardware and software gateways that analyze and pre-process the data before transferring it to the cloud. If the data read from the sensors and gateways are not changed from its previous reading value then it does not transfer over the cloud, this saves the data used.

- Data center/ Cloud: The Data Center or Cloud comes under the Management Services which process the information through analytics, management of device and security controls. Beside this security controls and device management the cloud transfer the data to the end users application such as Retail, Healthcare, Emergency, Environment, and Energy, etc.

End Points

- Endpoints as edge computers. IoT endpoints extend the computing capacity of the network and limit the amount of data that must be communicated from a host device to the backend network for computation requirements. Endpoint functionality. Endpoint devices are designed to perform specific, limited functions.

- An Endpoint, from an IoT perspective, is a physical computing device that performs a function or task as a part of an Internet connected product or service. An Endpoint, for example, could be a wearable fitness device, an industrial control system, an automotive telematics unit or even a personal drone unit.

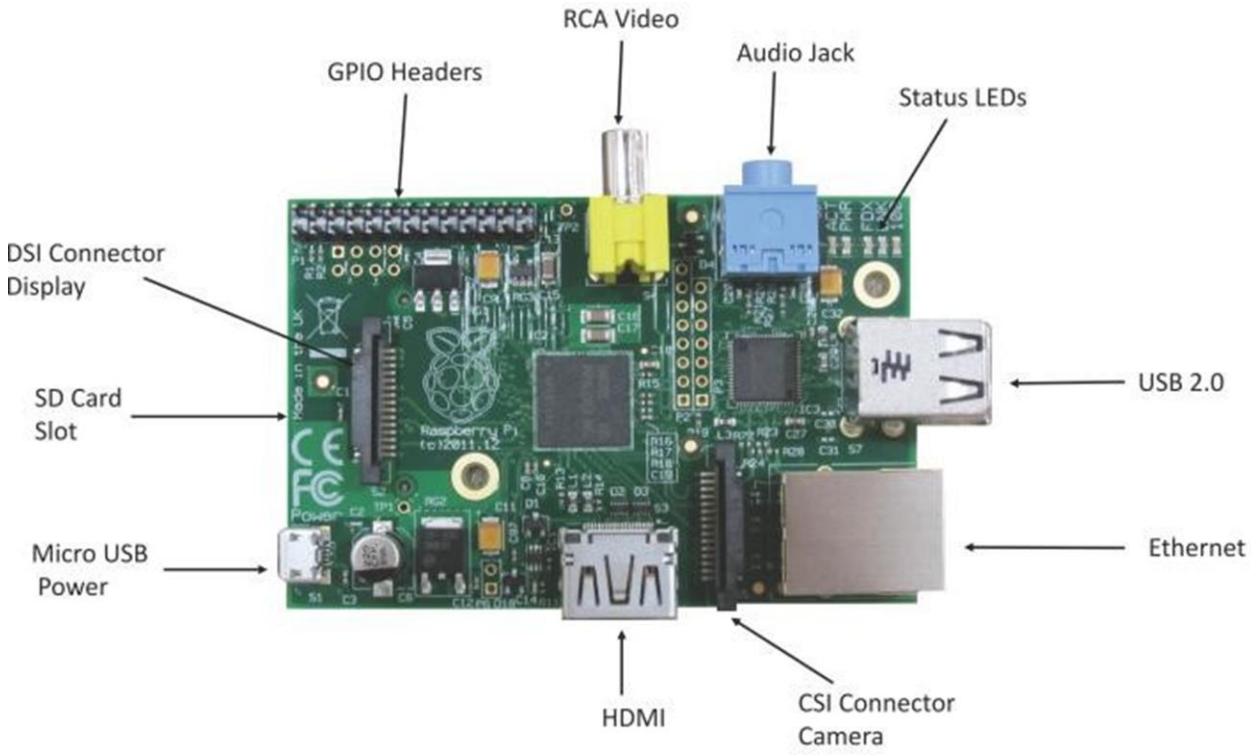
- Laptops, desktops, mobile phones, tablets, servers, and virtual environments can all be considered endpoints.

Introduction to Raspberry PI, Interfaces

Exemplary Device: Raspberry Pi

Raspberry Pi is a low-cost mini-computer with the physical size of a credit card. Raspberry Pi runs various flavors of Linux and can perform almost all tasks that a normal desktop computer can do. Raspberry Pi also allows interfacing sensors and actuators through the general purpose

I/O pins. Since Raspberry Pi runs Linux operating system, it supports Python "out of the box". Raspberry Pi is a low-cost mini-computer with the physical size of a credit card. Raspberry Pi runs various flavors of Linux and can perform almost all tasks that a normal desktop computer can do. Raspberry Pi also allows interfacing sensors and actuators through the general purpose I/O pins. Since Raspberry Pi runs Linux operating system, it supports Python "out of the box".



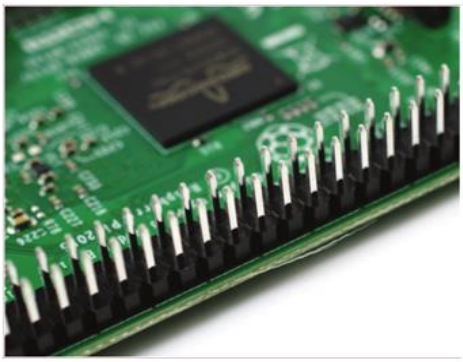
Linux on Raspberry Pi

1. Raspbian: Raspbian Linux is a Debian Wheezy port optimized for RaspberryPi.
2. Arch: Arch is an Arch Linux port for AMDdevices.
3. Pidora: Pidora Linux is a Fedora Linux optimized for RaspberryPi.
4. RaspBMC: RaspBMC is an XBMC media-center distribution for RaspberryPi.
5. OpenELEC: OpenELEC is a fast and user-friendly XBMC media-centerdistribution.
6. RISC OS: RISC OS is a very fast and compact operatingsystem.

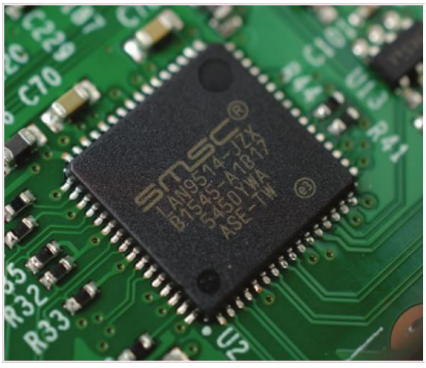
BCM: the Broadcom BCM2837 system-on-chip (SoC) includes four high-performance ARM Cortex-A53 processing cores running at 1.2GHz with 32kB Level 1 and 512kB Level 2 cache

memory, a VideoCore IV graphics processor, and is linked to a 1GB LPDDR2 memory module on the rear of the board.

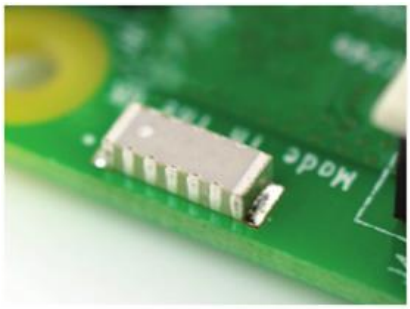
GPIO: The Raspberry Pi 3 features the same 40-pin general-purpose input-output (GPIO) header as all the Pis going back to the Model B+ and Model A+. Any existing GPIO hardware will work without modification; the only change is a switch to which UART is exposed on the GPIO's pins, but that's handled internally by the operating system



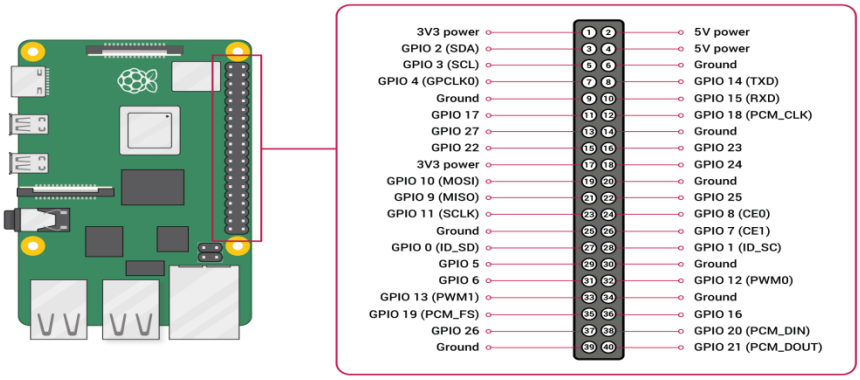
USB chip: The Raspberry Pi 3 shares the same SMSC LAN9514 chip as its predecessor, the Raspberry Pi 2, adding 10/100 Ethernet connectivity and four USB channels to the board. As before, the SMSC chip connects to the SoC via a single USB channel, acting as a USB-to-Ethernet adaptor and USB hub



Antenna: There's no need to connect an external antenna to the Raspberry Pi 3. Its radios are connected to this chip antenna soldered directly to the board, in order to keep the size of the device to a minimum. Despite its diminutive stature, this antenna should be more than capable of picking up wireless LAN and Bluetooth signals – even through walls.



Raspberry Pi GPIO



- The Raspberry Pi Model A and B boards have a 26-pin 2.54 mm (100 mil) expansion header, marked as P1, arranged in a 2x13 strip. They provide 8 GPIO pins plus access to I²C, SPI, UART), as well as +3.3 V, +5 V and GND supply lines.
- A 40-pin GPIO header is found on all current Raspberry Pi boards (unpopulated on Pi Zero and Pi Zero W). Prior to the Pi 1 Model B+ (2014), boards comprised a shorter 26-pin header. Any of the GPIO pins can be designated (in software) as an input or output pin and used for a wide range of purposes.

Raspberry Pi Interfaces

1. **Serial**: The serial interface on Raspberry Pi has receive (Rx) and transmit (Tx) pins for communication with serialperipherals.
2. **SPI:** Serial Peripheral Interface (SPI) is a synchronous serial data protocol used for communicating with one or more peripheraldevices.
3. **I2C:** The I2C interface pins on Raspberry Pi allow you to connect hardware modules.

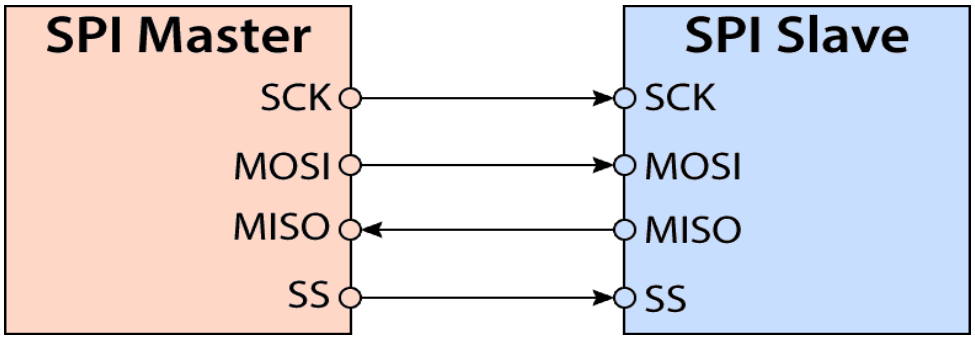
I2C interface allows synchronous data transfer with just two pins - SDA (data line) and SCL (clockline).

Serial Interfaces

- The serial interface on Raspberry Pi has receive (Rx) and transmit (Tx) pins for communication with serial peripherals.
- The Raspberry Pi 3 Model B, B+, 4 and Raspberry Pi Zero W contain two UART controllers which can be used for serial communication, the mini UART and PL011 UART.
- By default, the mini UART is mapped to the TXD (GPIO 14) and RXD (GPIO 15) on the 40 pin GPIO header and the PL011 UART is used for the Bluetooth/Wireless module but either module can be mapped to the GPIO port.

What is Serial Peripheral Interface?

- **Serial Peripheral Interface (SPI)** is an synchronous serial bus commonly used to send data between micro controllers and small peripherals such as shift registers, sensors, and SD cards.
- It uses separate clock (SCK) and data lines (MISO, MOSI), along with a select line(SS) to choose among the multiple slave devices



With SPI, we can interface multiple peripheral devices as SPI slaves and control them with the SPI master. In general, each slave will need a separate SS line. To talk to a particular slave, the SPI master makes that slave's SS line low and keeps the rest of them high

I2C Interface

- The I2C interface pins on Raspberry Pi allow you to connect hardware modules. I2C interface allows synchronous data transfer with just two pins - SDA (data line) and SCL (clock line).

- GPIO 2 and GPIO 3 - the Raspberry Pi's I2C1 pins - allow for two-wire communication with a variety of external sensors and devices. ... I2C is a multi-drop bus, multiple

devices can be connected to these same two pins. Each device has its own unique I2C address.

- I2C is a multi-device bus used to connect low-speed peripherals to computers and embedded systems. The Raspberry Pi supports this interface on its GPIO header and it is a great way to connect sensors and devices. Once configured you can connect more than one device without using up additional pins on the header.

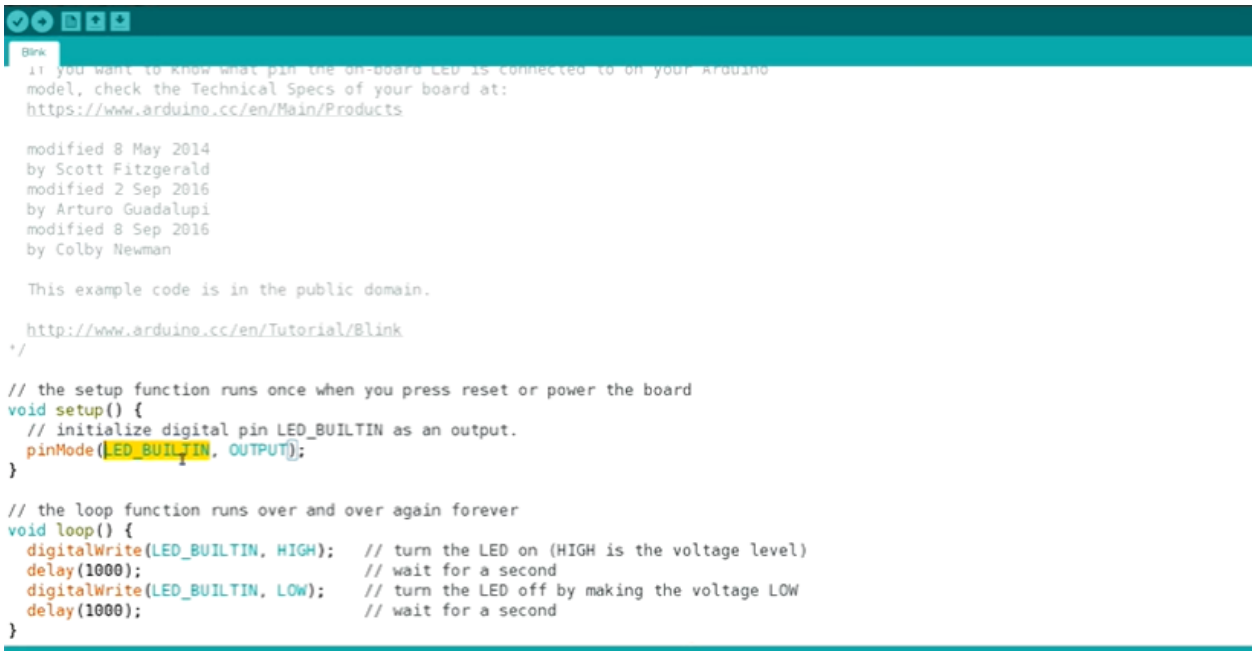
Python program with Raspberry PI

Recall -python program

```
Sum=0,i=1                Initialisation/declaration
  while i<=5:              Repetition i.e.,1<=5
   sum=sum+1                0+1
   i=i+1                      i=2
  print (sum)
```

This is a small example of python program

Arduino interface (program to interface with arduino)



```
  If you want to know what pin the on-board LED is connected to on your Arduino
  model, check the Technical Specs of your board at:
  https://www.arduino.cc/en/Main/Products

  modified 8 May 2014
  by Scott Fitzgerald
  modified 2 Sep 2016
  by Arturo Guadalupi
  modified 8 Sep 2016
  by Colby Newman

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Arduino interface

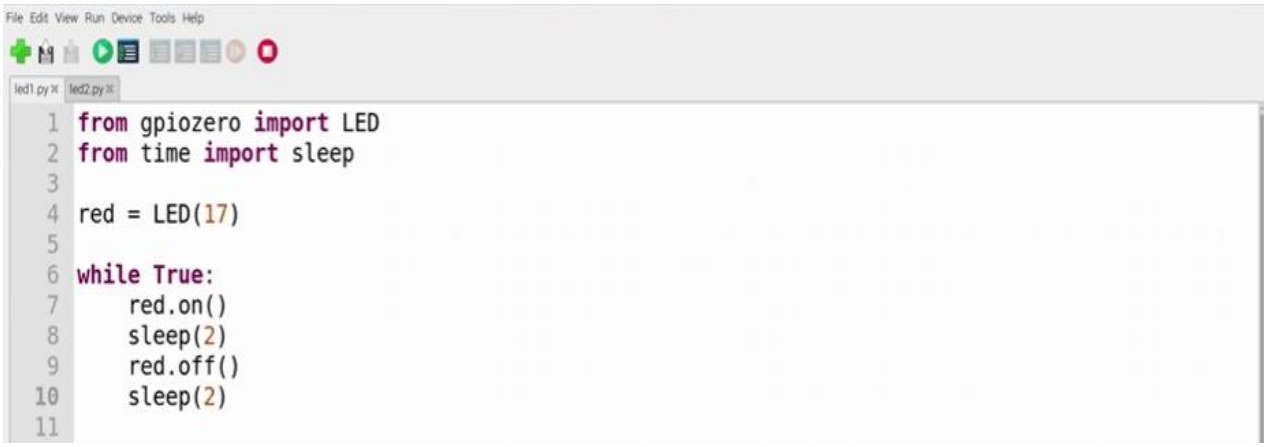
GPIO Zero

- GPIO Zero started out as a friendly API on top of the RPi.GPIO library, but later we extended it to allow other pin libraries to be used.
- With GPIO Zero, you import the name of the interfaces you're using, for example: from

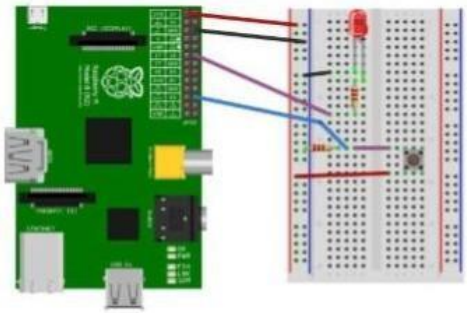
gpiozero import LED

- Also you must correctly wire up any components you're using and connect them to the GPIO pins

Raspberry Pi interface



Raspberry Pi Example: Interfacing LED and switch with Raspberry Pi



from time import sleeP

import RPi.GPIO asGPIO

GPIO.setmode(GPIO.BC

M)

#Switch Pin

GPIO.setup(25,GPIO.IN) #LEDPin

GPIO.setup(18,GPIO.OUT)

state=false

deftoggleLED(pin):

state = not state

GPIO.output(pin,sta

te)

whileTrue:
try:

if (GPIO.input(25) ==True):

toggleLED(pin) sleep(.01)

except Keyboard Interrupt:

exit()

Code for Blinking an LED with Raspberry Pi

```
#!/usr/bin/env python

import RPi.GPIO as GPIO # RPi.GPIO can be referred as GPIO from now
import time

ledPin = 22    # pin22

def setup():
        GPIO.setmode(GPIO.BOARD)       # GPIO Numbering of Pins
        GPIO.setup(ledPin, GPIO.OUT)   # Set ledPin as output
        GPIO.output(ledPin, GPIO.LOW)  # Set ledPin to LOW to turn Off the
LED

def loop():
        while True:
                print 'LED on'
                GPIO.output(ledPin, GPIO.HIGH)   # LED On
                time.sleep(1.0)                   # wait 1 sec
                print 'LED off'
                GPIO.output(ledPin, GPIO.LOW)   # LED Off
                time.sleep(1.0)                  # wait 1 sec
def endprogram():

        GPIO.output(ledPin, GPIO.LOW)     # LED Off
        GPIO.cleanup()                    # Release resources
```

```
if __name__ == '__main__':         # Program starts from here
        setup()
        try:
                loop()
        except KeyboardInterrupt:  # When 'Ctrl+C' is pressed, the destroy()
will be  executed.
                endprogram()
```

Other Devices :

- pcDuino
- BeagleBone Black
- Cubieboard

UNIT – V
DOMAIN SPECIFIC APPLICTION

IoT Applications for :

- Home
- Cities
- Environment
- Energy Systems
- Retail
- Logistics
- Industry
- Agriculture
- Health & Lifestyle

IoT applications for smart homes:

- Smart Lighting
- Smart Appliances
- Intrusion Detection
- Smoke / Gas Detectors

Home Automation Smart Lighting

- Smart lighting achieves energy savings by sensing the human movements and their environments and controlling the lights accordingly.
- Key enabling technologies for smart lighting include :
  - Solid state lighting (such as LED lights)
  - IP-enabled lights
- Wireless-enabled and Internet connected lights can be controlled remotely from IoT applications such as a mobile or web application.
- Paper:
  - Energy-aware wireless sensor network with ambient intelligence for smart LED lighting system control [IECON, 2011]-> presented controllable LED lighting system that is embedded with ambient intelligence gathered from a distributed smart WSN to optimize and control the lighting system to be more efficient and user-oriented.

Home Automation Smart Appliances

- Smart appliances make the management easier and provide status information of appliances to the users remotely. E.g: smart washer/dryer that can be controlled remotely and notify when the washing/drying cycle is complete.
- Open Remote is an open source automation platform for smart home and building that can control various appliances using mobile and web applications.
- It comprises of three components:
  - a Controller-> manages scheduling and runtime integration between devices.
  - a Designer -> allows to create both configuration for the controller and user interface designs.
  - Control Panel -> allows to interact with devices and control them.

- Paper: - An IoT-based Appliance Control System for Smart Home [ICICIP, 2013] implemented an IoT based appliance control system for smart homes that uses a smart-central controller to set up a wireless sensor and actuator network and control modules for appliances

Home Automation Intrusion Detection

- Home intrusion detection systems use security cameras and sensors to detect intrusions and raise alerts.
- The form of the alerts can be in form: - SMS - Email - Image grab or a short video clip as an email attachment

- Papers :
  - Could controlled intrusion detection and burglary prevention stratagems in home automation systems [BCFIC, 2012] ->present a controlled intrusion detection system that uses location-aware services, where the geo-location of each node of a home automation system is independently detected and stored in the cloud

  - An Intelligent Intrusion Detection System Based on UPnP Technology for Smart Living [ISDA, 2008] -> implement an intrusion detection system that uses image processing to recognize the intrusion and extract the intrusion subject and generate Universal-Plug-and-Play (UPnP-based) instant messaging for alerts.

Home Automation Smoke / Gas Detectors

- Smoke detectors are installed in homes and buildings to detect smoke that is typically an early sign of fire.
- It uses optical detection, ionization or air sampling techniques to detect smoke
- The form of the alert can be in form :
  Signals that send to a fire alarm system
- Gas detector can detect the presence of harmful gases such as carbon monoxide (CO), liquid petroleum gas (LPG), etc.
- Paper :
  - Development of Multipurpose Gas Leakage and Fire Detector with Alarm System [TIIEC, 2013]-> designed a system that can detects gas leakage and smoke and gives visual level indication.

Cities IoT applications for smart cities:

1. Smart Parking
2. Smart Lighting for Road
3. Smart Road
4. Structural Health Monitoring
5. Surveillance
6. Emergency Response

Cities Smart Parking

- Finding the parking space in the crowded city can be time consuming and frustrating
- Smart parking makes the search for parking space easier and convenient for driver.
- It can detect the number of empty parking slots and send the information over the Internet to the smart parking applications which can be accessed by the drivers using their smart phones, tablets, and in car navigation systems.
- Sensors are used for each parking slot to detect whether the slot is empty or not, and this information is aggregated by local controller and then sent over the Internet to database.
- Paper :
  • Design and implementation of a prototype Smart Parking (SPARK) system using WSN [International Conference on Advanced Information Networking and Applications Workshop, 2009]-> designed and implemented a prototype smart parking system based on wireless sensor network technology with features like remote parking monitoring, automate guidance, and parking reservation mechanism.

Cities Smart Lighting for Roads

- It can help in saving energy
- Smart lighting for roads allows lighting to be dynamically controlled and also adaptive to ambient conditions.
- Smart light connected to the Internet can be controlled remotely to configure lighting schedules and lighting intensity.
- Custom lighting configurations can be set for different situations such as a foggy day, a festival, etc.
- Paper :
  • Smart Lighting solutions for Smart Cities [International Conference on Advance Information

Networking and Applications Workshop, 2013]-> described the need for smart lighting system in smart cities, smart lighting features and how to develop interoperable smart lighting solutions.

Cities Smart Roads

- Smart Roads provides information on driving conditions, travel time estimates and alerts in case of poor driving conditions, traffic congestions and accidents.
- Such information can help in making the roads safer and help in reducing traffic jams
- Information sensed from the roads can be communicated via internet to cloud-based applications and social media and disseminated to the drivers who subscribe to such applications.
- Paper:
  • Sensor networks for smart roads [PerCom Workshop, 2006]-> proposed a distributed and autonomous system of sensor network nodes for improving driving safety on public roads, the system can provide the driver and passengers with a consistent view of the road situation a few hundred metes ahead of them or a few dozen miles away, so that they can react to potential dangers early enough.

Cities Structural Health Monitoring

- It uses a network of sensors to monitor the vibration levels in the structures such as bridges and buildings.
- The data collected from these sensors is analyzed to assess the health of the structures.
- By analyzing the data it is possible to detect cracks and mechanical breakdowns, locate the damages to a structure and also calculate the remaining life of the structure.
- Using such systems, advance warnings can be given in the case of imminent failure of the structure.
- Paper:
  - Environmental Effect Removal Based Structural Health Monitoring in the Internet of Things [International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, 2013]-> proposed an environmental effect removal based structural health monitoring scheme in an IoT environment.

  - Energy harvesting technologies for structural health monitoring applications [IEEE Conference on Technologies for Sustainability, 2013] -> Explored energy harvesting technologies of harvesting ambient energy, such as mechanical vibrations, sunlight, and wind.◊

Cities Surveillance

- Surveillance of infrastructure, public transport and events in cities is required to ensure safety and security.
- City wide surveillance infrastructure comprising of large number of distributed and Internet connected video surveillance cameras can be created.
- The video feeds from surveillance cameras can be aggregated in cloud-based scalable storage solutions.
- Cloud-based video analytics applications can be developed to search for patterns of specific events from the video feeds.

Cities Emergency Response

- IoT systems can be used for monitoring the critical infrastructure cities such as buildings, gas, and water pipelines, public transport and power substations.
- IoT systems for critical infrastructure monitoring enable aggregation and sharing of information collected from lager number of sensors.
- Using cloud-based architectures, multi-modal information such as sensor data, audio, video feeds can be analyzed I near real-time to detect adverse events.
- The alert can be in the form :
  Alerts sent to the
  public Re-rerouting
  of traffic
  Evacuations of the affected areas

Environment IoT applications for smart environments:

1. Weather Monitoring
2. Air Pollution Monitoring
3. Noise Pollution Monitoring
4. Forest Fire Detection
5. River Flood Detection

Environment Weather Monitoring

- It collects data from a number of sensor attached such as temperature, humidity, pressure, etc and send the data to cloud-based applications and store back-ends.
- The data collected in the cloud can then be analyzed and visualized by cloud-based applications.
- Weather alert can be sent to the subscribed users from such applications.
- AirPi is a weather and air quality monitoring kit capable of recording and uploading information about temperature, humidity, air pressure, light levels, UV levels, carbon monoxide, nitrogen dioxide and smoke level to the Internet.
- Paper:
  - PeWeMoS – Pervasive Weather Monitoring System [ICPCA, 2008]-> Presented a pervasive weather monitoring system that is integrated with buses to measure weather variables like humidity, temperature, and air quality during the bus path

Environment Air Pollution Monitoring

- IoT based air pollution monitoring system can monitor emission of harmful gases by factories and automobiles using gaseous and meteorological sensors.
- The collected data can be analyzed to make informed decisions on pollutions control approaches.
- Paper: - Wireless sensor network for real-time air pollution monitorings [ICCSPA, 2013]-> Presented a real time air quality monitoring system that comprises of several distributed monitoring stations that communicate via wireless with a back- end server using machine-to machine communication.

Environment Noise Pollution Monitoring

- Noise pollution monitoring can help in generating noise maps for cities.
- It can help the policy maker in making policies to control noise levels near residential areas, school and parks.
- It uses a number of noise monitoring stations that are deployed at different places in a city.
- The data on noise levels from the stations is collected on servers or in the cloud and then the collected data is aggregate to generate noise maps.
- Papers :
  - Noise mapping in urban environments : Applications at Suez city center [ICCIE, 2009]Presented a noise mapping study for a city which revealed that the city suffered from serious noise pollution.

  - SoundOfCity – Continuous noise monitoring for a health city [PerComW,2013]-> Designed a smartphone application that allows the users to continuously measure noise levels and send to a central server here all generated information is aggregated and mapped to a meaningful noise visualization map.

Environment Forest Fire Detection

- IoT based forest fire detection system use a number of monitoring nodes deployed at different location in a forest.
- Each monitoring node collects measurements on ambient condition including temperature, humidity, light levels, etc.
- Early detection of forest fires can help in minimizing the damage.
- Papers:
  • A novel accurate forest fire detection system using wireless sensor networks [International Conference on Mobile Ad- hoc and Sensor Networks, 2011]-> Presented a forest fire detection system based on wireless sensor network. The system uses multi-criteria detection which is

implemented by the artificial neural network. The ANN fuses sensing data corresponding to ,multiple attributes of a forest fire such as temperature, humidity, infrared and visible light to detect forest fires.

Environment River Flood Detection

- IoT based river flood monitoring system uses a number of sensor nodes that monitor the water level using ultrasonic sensors and flow rate using velocity sensors.
- Data from these sensors is aggregated in a server or in the cloud, monitoring applications raise alerts when rapid increase in water level and flow rate is detected.
- Papers:

  • RFMS : Real time flood monitoring system with wireless sensor networks [MASS, 2008]-> Described a river flood monitoring system that measures river and weather conditions through wireless sensor nodes equipped with different sensors

  • Urban Flash Flood Monitoring, Mapping and Forecasting via a Tailored Sensor Network System [ICNSC, 2006] -> Described a motes-based sensor network for river flood monitoring that includes a water level monitoring module, network video recorder module, and data processing module that provides floods information n the form of raw data, predict data, and video feed.

Energy IoT applications for smart energy systems:

1. Smart Grid
2. Renewable Energy Systems
3. Prognostics

Energy Smart Grids

- Smart grid technology provides predictive information and recommendation s to utilize, their suppliers, and their customers on how best to manage power.
- Smart grid collect the data regarding :
  - Electricity generation
  - Electricity consumption
  - Storage
  - Distribution and equipment health data
- By analyzing the data on power generation, transmission and consumption of smart grids can improve efficiency throughout the electric system.
- Storage collection and analysis of smarts grids data in the cloud can help in dynamic optimization of system operations, maintenance, and planning.
- Cloud-based monitoring of smart grids data can improve energy usage usage levels via energy feedback to users coupled with real-time pricing information.
- Condition monitoring data collected from power generation and transmission systems can help in detecting faults and predicting outages.

Energy Renewable Energy System

- Due to the variability in the output from renewable energy sources (such as solar and wind), integrating them into the grid can cause grid stability and reliability problems.
- IoT based systems integrated with the transformer at the point of interconnection measure the electrical variables and how much power is fed into the grid
- To ensure the grid stability, one solution is to simply cut off the overproductions.
- Paper:

  - Communication systems for grid integration of renewable energy resources [IEEE Network, 2011]-> Provided the closed-loop controls for wind energy system that can be used to regulate the voltage at point of interconnection which coordinate wind turbine outputs and provides reactive power support.

Energy Prognostics

- IoT based prognostic real-time health management systems can predict performance of machines of energy systems by analyzing the extent of deviation of a system from its normal operating profiles.
- In the system such as power grids, real time information is collected using specialized electrical sensors called Phasor Measurement Units (PMU)
- Analyzing massive amounts of maintenance data collected from sensors in energy systems and equipment can provide predictions for impending failures.
- OpenPDC is a set of applications for processing of streaming time-series data collected from Phasor Measurements Units (PMUs) in real-time.

Retail IoT applications in smart retail systems:

1. Inventory Management
2. Smart Payments
3. Smart Vending Machines

Retail Inventory Management

- IoT system using Radio Frequency Identification (RFID) tags can help inventory management and maintaining the right inventory levels.
- RFID tags attached to the products allow them to be tracked in the real-time so that the inventory levels can be determined accurately and products which are low on stock can be replenished.
- Tracking can be done using RFID readers attached to the retail store shelves or in the warehouse.
- Paper:
  - RFID data-based inventory management of time-sensitive materials [IECON, 2005]-> described an RFID data-based inventory management system for time-sensitive materials

Retail Smart Payments

- Smart payments solutions such as contact-less payments powered technologies such as Near field communication (NFC) and Bluetooth.
- NFC is a set of standards for smart-phones and other devices to communicate with each other by bringing them into proximity or by touching them
- Customer can store the credit card information in their NFC-enabled smart-phones and make payments by bringing the smart-phone near the point of sale terminals.
- NFC maybe used in combination with Bluetooth, where NFC initiates initial pairing of devices to establish a Bluetooth connection while the actual data transfer takes place over Bluetooth.

Retail Smart Vending Machines

- Smart vending machines connected to the Internet allow remote monitoring of inventory levels, elastic pricing of products, promotions, and contact-less payments using NFC.
- Smart-phone applications that communicate with smart vending machines allow user preferences to be remembered and learned with time. E.g: when a user moves from one vending machine to the other and pair the smart-phone, the user preference and favourite product will be saved and then that data is used for predictive maintenance.
- Smart vending machines can communicated each others, so if a product out of stock in a machine, the user can be routed to nearest machine
- For perishable items, the smart vending machines can reduce the price as the expiry date nears.

Logistic IoT applications for smart logistic systems:

1. Fleet Tracking
2. Shipment Monitoring
3. Remote Vehicle Diagnostics

Logistics Fleet Tracking

- Vehicle fleet tracking systems use GPS technology to track the locations of the vehicles in the real-time.
- Cloud-based fleet tracking systems can be scaled up on demand to handle large number of vehicles,

- The vehicle locations and routers data can be aggregated and analyzed for detecting bottlenecks I the supply chain such as traffic congestions on routes, assignments and generation of alternative routes, and supply chain optimization

- Paper:
  - A Fleet Monitoring System for Advanced Tracking of commercial Vehicles [IEEE International Conference in Systems, Man and Cybernetics, 2006]-> provided a system that can analyze messages sent from the vehicles to identify unexpected incidents and discrepancies between actual and planned data, so that remedial actions can be taken.

Logistics Shipment Monitoring

- Shipment monitoring solutions for transportation systems allow monitoring the conditions inside containers.
- E.g : Containers carrying fresh food produce can be monitored to prevent spoilage of food. IoT based shipment monitoring systems use sensors such as temperature, pressure, humidity, for instance, to monitor the conditions inside the containers and send the data to the cloud, where it can be analyzed to detect food spoilage.
- Paper:
  - On a Cloud-Based Information Technology Framework for Data Driven Intelligent Transportation System [Journal of Transportation Technologies, 2013]-> proposed a cloud based framework for real time fresh food supply tracking and monitoring

Logistics Remote Vehicle Diagnostics

- It can detect faults in the vehicles or warn of impending faults.
- These diagnostic systems use on-board IoT devices for collecting data on vehicle operation such as speed, engine RPM, coolent temperature, fault code number and status of various vehicle sub-system.
- Modern commercial vehicles support on-board diagnostic (OBD) standard such as OBD-II
- OBD systems provide real-time data on the status of vehicle sub-systems and diagnostic trouble codes which allow rapidly identifying the faults in the vehicle.
- IoT based vehicle diagnostic systems can send the vehicle data to centralized servers or the cloud where it can be analyzed to generate alerts and suggest remedial actions.

Agriculture IoT applications for smart agriculture:

1. Smart Irrigation
2. Green House Control

Agriculture Smart Irrigation

- Smart irrigation system can improve crop yields while saving water.
- Smart irrigation systems use IoT devices with soil moisture sensors to determined the amount of moisture on the soil and release the flow of the water through the irrigation pipes only when the moisture levels go below a predefined threshold.
- It also collect moisture level measurements on the server on in the cloud where the collected data can be analyzed to plan watering schedules.
- Cultivar's RainCould is a device for smart irrigation that uses water valves, soil sensors, and a WiFi enabled programmable computer. [http://ecultivar.com/rain-cloud-product-project/]

Agriculture Green House Control

- It controls temperature, humidity, soil, moisture, light, and carbon dioxide level that are monitored by sensors and climatological conditions that are controlled automatically using actuation devices.

- IoT systems play an importance role in green house control and help in improving productivity.
- The data collected from various sensors is stored on centralized servers or in the cloud where analysis is performed to optimize the control strategies and also correlate the productivity with different control strategies.
- Paper: - Wireless sensing and control for precision Green house management [ICST, 2012]-> Provided a system that uses wireless sensor network to monitor and control the agricultural parameters like temperature and humidity in the real time for better management and maintenance of agricultural production.

Industry IoT applications in smart industry:

1. Machine Diagnosis & Prognosis
2. Indoor Air Quality Monitoring

Industry Machine Diagnosis & Prognosis

- Machine prognosis refers to predicting the performance of machine by analyzing the data on the current operating conditions and how much deviations exist from the normal operating condition.
- Machine diagnosis refers to determining the cause of a machine fault.
- Sensors in machine can monitor the operating conditions such as temperature and vibration levels, sensor data measurements are done on timescales of few milliseconds to few seconds which leads to generation of massive amount of data.
- Case-based reasoning (CBR) is a commonly used method that finds solutions to new problems based on past experience.
- CBR is an effective technique for problem solving in the fields in which it is hard to establish a quantitative mathematical model, such as machine diagnosis and prognosis.

Industry Indoor Air Quality Monitoring

- Harmful and toxic gases such as carbon monoxide (CO), nitrogen monoxide (NO), Nitrogen Dioxide, etc can cause serious health problem of the workers.
- IoT based gas monitoring systems can help in monitoring the indoor air quality using various gas sensors. - The indoor air quality can be placed for different locations
- Wireless sensor networks based IoT devices can identify the hazardous zones, so that corrective measures can be taken to ensure proper ventilation.
- Papers:
  - A hybrid sensor system for indoor air quality monitoring [IEEE International Conference on Distributed Computing in Sensor System, 2013]-> presented a hybrid sensor system for indoor air quality monitoring which contains both stationary sensor and mobile sensors.

Health & Lifestyle IoT applications in smart health & lifestyle:

1. Health & Fitness Monitoring
2. Wearable Electronics

Fitness Monitoring

- Wearable IoT devices allow to continuous monitoring of physiological parameters such as blood pressure, heart rate, body temperature, etc than can help in continuous health and fitness monitoring.
- It can analyze the collected health-care data to determine any health conditions or anomalies.
- The wearable devices may can be in various form such as:
  - Belts
  - Wrist-bands
- Papers:
  - Toward ubiquitous mobility solutions for body sensor network health care [IEEE Communications Magazine, 2012]-> Proposed an ubiquitous mobility approach for body sensor network in health-care

Health & Lifestyle Wearable Electronics

- Wearable electronics such as wearable gadgets (smart watch, smart glasses, wristbands, etc) provide various functions and features to assist us in our daily activities and making us lead healthy lifestyles.
- Using the smart watch, the users can search the internet, play audio/video files, make calls, play games, etc.
- Smart glasses allows users to tae photos and record videos, get map directions, check flight status or search internet using voice commands
- Smart shoes can monitor the walking or running speeds and jumps with the help of embedded sensors and be paired with smart-phone to visualize the data.
- Smart wristbands can tract the daily exercise and calories burnt.

IoT Levels and Deployment Templates

The IoT system consists of several systems which includes:

- Database: Database can be either local or in the cloud and stores the data generated by the IoT device.
- Web Service: Web services serve as a link between the IoT device, application, database and analysis components.
- Analysis Component: This is responsible for analyzing the IoT data and generating results in a form that is easy for the user to understand.
- Application: IoT applications provide an interface that the users can use to control and monitor various aspects of the IoT system. Applications also allow users to view the system status and the processed data.

➢ **IoT level-1**

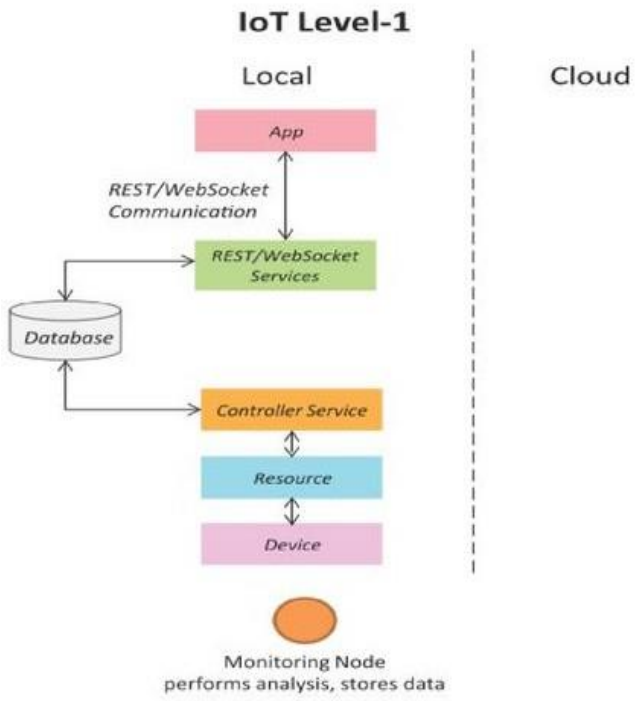
A level - 1 IoT system has a single node/device that performs sensing and/or actuation, stores data, performs analysis and hosts the application. Level - 1 IoT systems are suitable for modelling low cost and low complexity solutions where the data involved is not big and the analysis requirements are not computationally intensive.

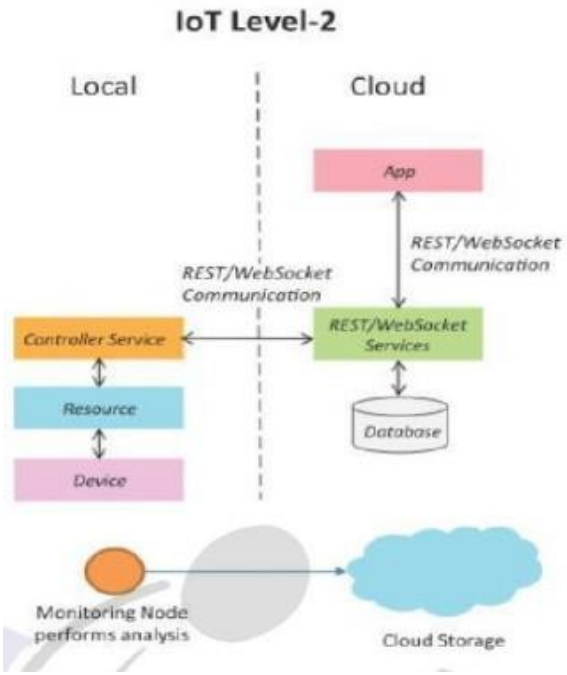
➢ **IoT level-2:**

- A level-2 IoT system has a single node that performs sensing and/or actuation and local analysis. Data is stored in the cloud and the application is usually cloud-based.
- Level-2 IoT systems are suitable for solutions where the data involved is big; however, the primary analysis requirement is not computationally intensive and can be done locally.

➢ **IoT level-3:**

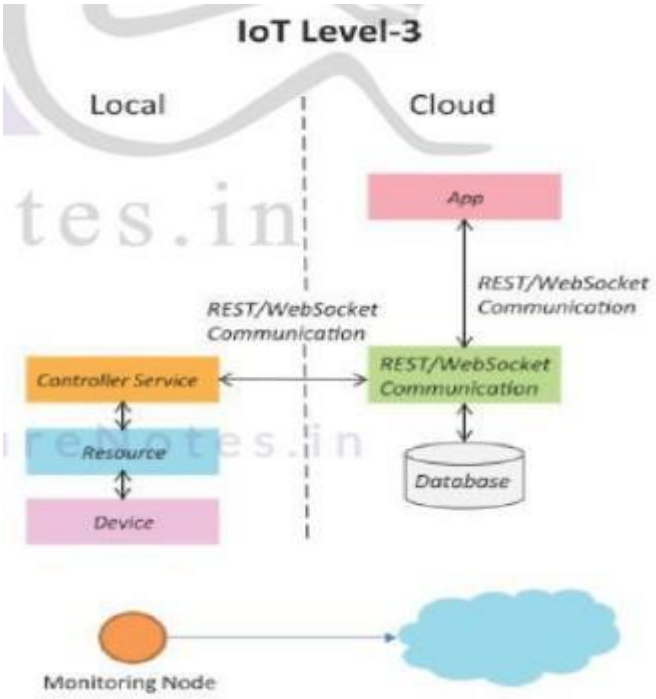
- A level-3 IoT system has a single node. Data is stored and analyzed in the cloud and the application is cloud-based.
- Level-3 IoT systems are suitable for solutions where the data involved is big and the analysis requirements are computationally intensive.

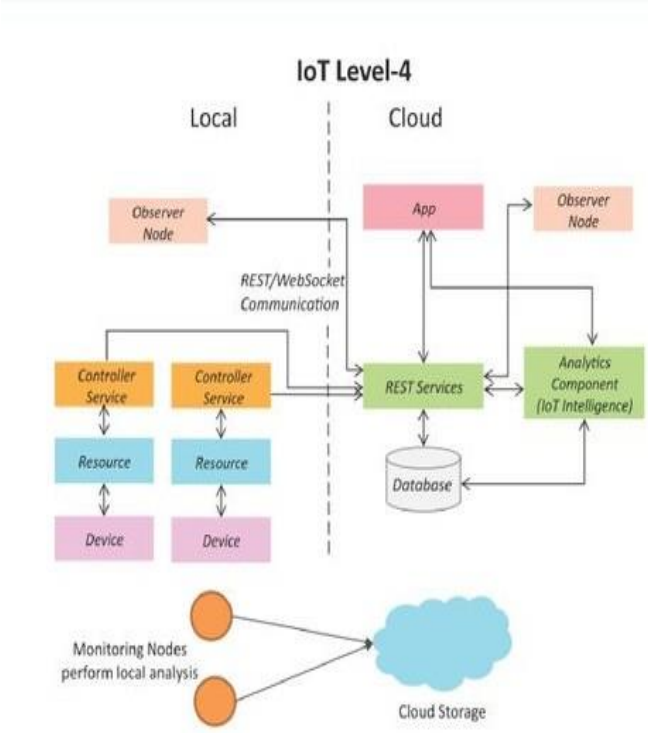


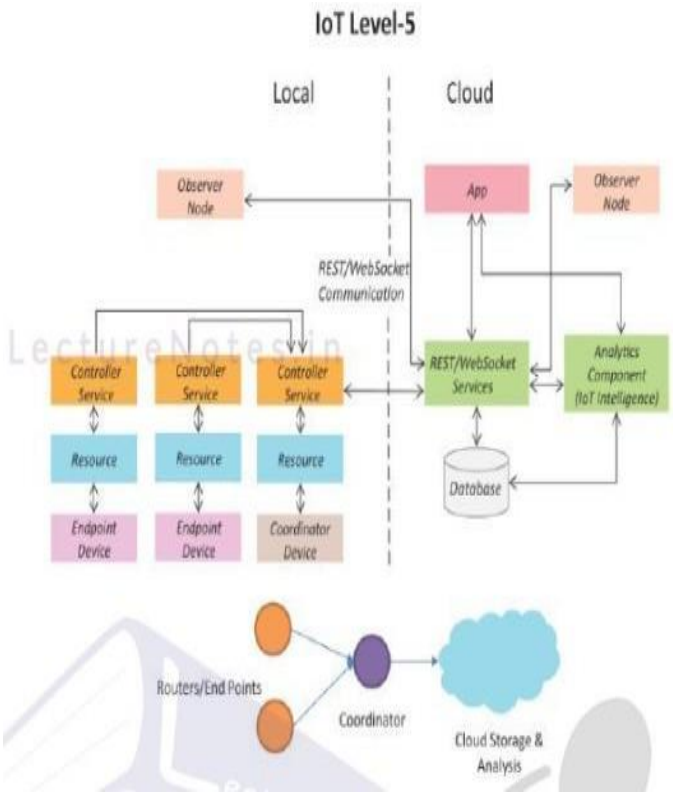


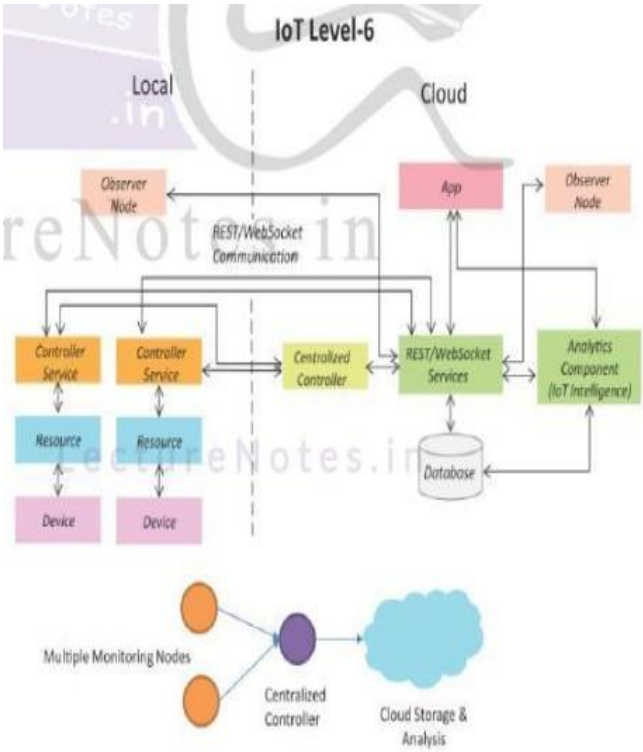
- **IoT level-4:**
  - A level-4 IoT system has multiple nodes that perform local analysis. Data is stored in the cloud and the application is cloud-based.
  - Level-4 contains local and cloud-based observer nodes which can subscribe to and receive information collected in the cloud from IoT devices.
  - Level-4 IoT systems are suitable for solutions where multiple nodes are required, the data involved is big and the analysis requirements are computationally intensive.

- **IoT level-5:**
  - A level-5 IoT system has multiple end nodes and one coordinator node. The end nodes perform sensing and/or actuation. The coordinator node collects data from the end nodes and sends it to the cloud. Data is stored and analyzed in the cloud and the application is cloud-based.
  - Level-5 IoT systems are suitable for solutions based on wireless sensor networks, in which the data involved is big and the analysis requirements are computationally intensive.









➢ **IoT level-6:**

- A level-6 IoT system has multiple independent end nodes that perform sensing

and/or actuation and send data to the cloud. Data is stored in the cloud and the application is cloud-based. The analytics component analyzes the data and stores the results in the cloud database.

- The results are visualized with the cloud-based application. The centralized controller is aware of the status of all the end nodes and sends control commands to the nodes.