SUBJECT CODE : 3171108

As per New Syllabus of

GUJARAT TECHNOLOGICAL UNIVERSITY

Semester - VII(Electrical / IT) Open Elective - III

Semester - VII(ECE) Professional Elective - V

INTERNET OF THINGS

Iresh A. Dhotre

M.E. (Information Technology)

Ex-Faculty, Sinhgad College of Engineering,

Pune.

TECHNICAL PUBLICATIONS

SINCE 1993

An Up-Thrust for Knowledge

SYLLABUS

Internet of Things - (3171108)

Credits	Examination Marks				Total Marks
C	Theory Marks		Practical Marks		
	ESE (E)	PA(M)	ESE (V)	PA (I)	
3	70	30	30	20	150

1. Introduction to Internet of Things

IoT Definition, IoT characteristics, M2M and IoT, End to End IoT Architecture, Physical design of IoT, Logical Design of IoT, Overview of IoT protocols, IoT levels and deployment templates, Challenges for IoT, Interdependencies of IoT and cloud computing, Web of things. **(Chapter - 1)**

2. Embedded IoT Devices

Sensors and actuators for IoT applications, IoT components and implementation, Programming of NodeMCU and Raspberry PI, Implementation of IoT with Edge devices, Reading sensor data and transmit to cloud, Controlling devices through cloud using mobile application and web application, Types and configurations of gatways, Specifications of IoT gateways (Practical aspects of this chapter should be covered during lab sessions). **(Chapter - 2)**

3. IoT Protocols

Link layer protocols, Network/internet layer protocols, Transport layer protocols, Application layer protocols: Hypertext transfer protocol (HTTP), Systematic HTTP access methodology, Web Socket, Constrained application protocol CoAP), Message Queue Telemetry Transport Protocol (MQTT), XMPP, DDS, AMQP. **(Chapter - 3)**

4. IoT Security and Challenges

IOT Security, Dangers, Assigning values to Information, Security Components, Key Management, Update Management, Challenges in IoT security. **(Chapter - 4)**

5. IoT Applications and Case Study

Broad categories of IoT applications: Consumer IoT, Commercial IoT, Industrial IoT, Infrastructure IoT, Military Things (IoMT)

IoT Case studies :

Home automation with IoT, River water pollution monitoring, Smart city street light control and monitoring, Health care monitoring, Voice Apps on IoT device. **(Chapter - 5)**

TABLE OF CONTENTS

Chapter - 1 Introduction to Internet of Things (1 - 1) to (1 - 62)

1.1 Evolution of IoT ..... 1 - 2
1.1.1 Definition of IoT ..... 1 - 3
1.1.2 IoT Characteristics ..... 1 - 5
1.1.3 Component of IoT ..... 1 - 5
1.1.4 Working of IoT ..... 1 - 6
1.1.5 Advantages and Disadvantages ..... 1 - 7
1.1.6 Applications of IoT ..... 1 - 8
1.2 M2M and IoT ..... 1 - 9
1.2.1 Architecture and Components of M2M ..... 1 - 11
1.2.2 Difference between M2M and IoT ..... 1 - 13
1.3 End to End IoT Architecture ..... 1 - 13
1.3.1 OneM2M Architecture ..... 1 - 13
1.3.2 IoT World Forum Standardized Architecture ..... 1 - 14
1.3.3 Simplified IoT Architecture ..... 1 - 16
1.4 Physical Design of IoT ..... 1 - 17
1.4.1 Things in IoT ..... 1 - 17
1.4.2 IoT Protocol ..... 1 - 18
1.5 Logical Design of IoT ..... 1 - 25
1.5.1 IoT Functional Blocks ..... 1 - 25
1.5.2 IoT Communication Model ..... 1 - 27
1.5.3 IoT Communication API's ..... 1 - 29
1.6 Overview of IoT Protocols ..... 1 - 30
1.6.1 Cloud Computing ..... 1 - 31
1.6.2 Big Data Analytic ..... 1 - 31
1.6.3 Wireless Sensor Networks ..... 1 - 32
1.6.4 Communication Protocols ..... 1 - 32
1.6.5 Embedded System ..... 1 - 33

1.7 IoT Levels and Deployment Templates ..... 1 - 33
1.7.1 IoT Level 1 ..... 1 - 34
1.7.2 IoT Level 2 ..... 1 - 35
1.7.3 IoT Level 3 ..... 1 - 35
1.7.4 IoT Level 4 ..... 1 - 36
1.7.5 IoT Level 5 ..... 1 - 36
1.7.6 IoT Level 6 ..... 1 - 37
1.8 Challenges for IoT ..... 1 - 37
1.9 Interdependencies of IoT and Cloud Computing ..... 1 - 40
1.9.1 Cloud Middleware ..... 1 - 41
1.9.2 Cloud Standards ..... 1 - 43
1.9.3 The Cloud of Things Architecture ..... 1 - 46
1.10 Web of Things ..... 1 - 48
1.10.1 Two Pillars of The Web ..... 1 - 49
1.10.2 Architecture Standardization for WoT ..... 1 - 51
1.10.3 Platform Middleware for IoT ..... 1 - 52
1.11 Fill in the Blanks with Answers ..... 1 - 57
1.12 Multiple Choice Questions with Answers ..... 1 - 57

Chapter - 2 Embedded IoT Devices (2 - 1) to (2 - 44)

2.1 Sensors ..... 2 - 2
2.1.1 Types of Sensors ..... 2 - 3
2.1.2 Sensor Data Communication Protocols ..... 2 - 4
2.1.3 Actuators ..... 2 - 4
2.2 Smart Objects ..... 2 - 5
2.2.1 Communication Patterns used for Smart Objects ..... 2 - 6
2.2.2 Connecting Smart Objects ..... 2 - 7
2.3 IoT System Building Blocks ..... 2 - 11
2.4 Arduino ..... 2 - 13
2.5 Raspberry Pi ..... 2 - 16
2.5.1 About the Board ..... 2 - 17

2.5.2 Linux on Raspberry Pi ..... 2 - 21
2.5.3 Difference between Raspberry Pi is and Desktop Computers ..... 2 - 23
2.6 Raspberry Pi Interface ..... 2 - 24
2.7 Raspberry Pi with Python ..... 2 - 27
2.7.1 Controlling LED with Raspberry Pi ..... 2 - 28
2.7.2 Interfacing an LED and Switch with Raspberry Pi ..... 2 - 31
2.7.3 Interfacing Light Sensor ..... 2 - 31
2.8 Implementation of IoT with Edge Devices ..... 2 - 34
2.9 Reading Sensor Data and Transmit to Cloud ..... 2 - 36
2.10 Controlling Devices through Cloud using Mobile Application and Web Application ..... 2 - 39
2.11 IoT Gateways ..... 2 - 40
2.12 Fill in the Blanks with Answers ..... 2 - 41
2.13 Multiple Choice Questions with Answers ..... 2 - 42

Chapter - 3 IoT Protocols (3 - 1) to (3 - 54)

3.1 Introduction of IoT Protocol ..... 3 - 2
3.1.1 Link Layer Protocols ..... 3 - 2
3.1.2 Network / Internet Layer Protocols ..... 3 - 2
3.1.3 Transport Layer ..... 3 - 2
3.1.4 Application Layer ..... 3 - 3
3.2 IEEE 802.15.4 ..... 3 - 3
3.2.1 802.15.4 Physical Layer ..... 3 - 4
3.2.2 MAC Layer ..... 3 - 5
3.2.3 FHSS and DHSS ..... 3 - 8
3.2.4 Choice of 802.15.4 Communication channel ..... 3 - 10
3.2.5 Beacon Enabled Mode ..... 3 - 10
3.2.6 Non-Beacon Enabled Mode ..... 3 - 11
3.3 LoRaWAN ..... 3 - 12
3.4 IPv4 ..... 3 - 16
3.4.1 Packet Format ..... 3 - 16

3.4.2 Classes of IP Address ..... 3 - 18
3.4.3 Public and Private Addresses ..... 3 - 20
3.5 IPv6 ..... 3 - 20
3.5.1 Packet Format ..... 3 - 21
3.5.2 Difference between IPv4 and IPv6 ..... 3 - 23
3.5.3 Advantages of IPv6 ..... 3 - 23
3.6 Transport Protocol ..... 3 - 23
3.6.1 Bluetooth Low Energy ..... 3 - 23
3.6.2 Components of BLE ..... 3 - 25
3.6.3 BLE Topology ..... 3 - 27
3.6.4 Comparison between Classic Bluetooth and Bluetooth Low Energy ..... 3 - 28
3.6.5 Light Fidelity ..... 3 - 28
3.6.6 Difference between Li-Fi and Wi-Fi ..... 3 - 30
3.7 Application Layer Protocols ..... 3 - 30
3.7.1 CoAP ..... 3 - 30
3.7.2 MQTT ..... 3 - 33
3.7.3 Difference between CoAP and MQTT ..... 3 - 37
3.7.4 Hypertext Transfer Protocol ..... 3 - 37
3.7.4.1 HTTP Methods ..... 3 - 40
3.7.4.2 Difference between Persistent and Non-persistent HTTP ..... 3 - 42
3.7.5 Systematic HTTP Access Methodology : REST API ..... 3 - 42
3.7.6 Web Socket ..... 3 - 44
3.7.7 XMPP ..... 3 - 45
3.7.8 DDS ..... 3 - 47
3.7.9 AMQP ..... 3 - 49
3.8 Fill in the Blanks with Answers ..... 3 - 50
3.9 Multiple Choice Questions with Answers ..... 3 - 50

Chapter - 4 IoT Security and Challenges (4 - 1) to (4 - 18)

4.1 IoT Security Issues and Need ..... 4 - 2
4.1.1 Security Architecture ..... 4 - 3
4.1.2 Security Requirement ..... 4 - 4

4.2 Assigning Values to Information ..... 4 - 5
4.2.1 Risk Assessment ..... 4 - 6
4.3 Security Components ..... 4 - 8
4.3.1 Components of an Information System ..... 4 - 11
4.4 Key Management ..... 4 - 12
4.5 Update Management ..... 4 - 13
4.6 Challenges in IoT Security ..... 4 - 14
4.7 Fill in the Blanks with Answers ..... 4 - 14
4.8 Multiple Choice Questions with Answers ..... 4 - 15

Chapter - 5 IoT Applications and Case Study (5 - 1) to (5 - 24)

5.1 Broad Categories of IoT Applications ..... 5 - 2
5.1.1 Consumer IoT ..... 5 - 2
5.1.2 Commercial IoT ..... 5 - 4
5.1.3 Industrial IoT ..... 5 - 4
5.1.4 Infrastructure IoT ..... 5 - 7
5.1.5 Military Things (IoMT) ..... 5 - 8
5.2 Home Automation with IoT ..... 5 - 9
5.2.1 Smart Lighting ..... 5 - 9
5.2.2 Smart Appliances ..... 5 - 11
5.2.3 Intrusion Detection ..... 5 - 12
5.2.4 Smoke for Gas Detection ..... 5 - 13
5.3 River Water Pollution Monitoring ..... 5 - 14
5.4 Smart City Street Light Control and Monitoring ..... 5 - 15
5.4.1 Smart Parking ..... 5 - 16
5.4.2 Smart Lighting ..... 5 - 19
5.4.3 Smart Roads ..... 5 - 20
5.5 Health Care Monitoring ..... 5 - 20
5.6 Voice Apps on IoT Device ..... 5 - 23

Solved Model Question Paper (M - 1) to (M - 2)

1 Introduction to Internet of Things

Syllabus

IoT Definition, IoT characteristics, M2M and IoT, End to End IoT Architecture, Physical design of IoT, Logical Design of IoT, Overview of IoT protocols, IoT levels and deployment templates, Challenges for IoT, Interdependencies of IoT and cloud computing, Web of things

Contents

1.1 Evolution of IoT

1.2 M2M and IoT

1.3 End to End IoT Architecture

1.4 Physical Design of IoT

1.5 Logical Design of IoT

1.6 Overview of IoT Protocols

1.7 IoT Levels and Deployment Templates

1.8 Challenges for IoT

1.9 Interdependencies of IoT and Cloud Computing

1.10 Web of Things

1.11 Fill in the Blanks

1.12 Multiple Choice Questions

1.1 Evolution of IoT

- The Internet of Things (IoT) refers to the capability of everyday devices to connect to other devices and people through the existing Internet infrastructure. Devices connect and communicate in many ways.
- Examples of this are smartphones that interact with other smartphones, vehicle-to-vehicle communication, connected video cameras, and connected medical devices.
- They are able to communicate with consumers, collect and transmit data to companies, and compile large amounts of data for third parties.
- Things are objects of the physical world (physical things) or of the information world (virtual world) which are capable of being identified and integrated into communication networks. Things have associated information, which can be static and dynamic.
- Physical things exist in the physical world and are capable of being sensed, actuated and connected. Examples of physical things include the surrounding environment, industrial robots, goods and electrical equipment.
- Virtual things exist in the information world and are capable of being stored, processed and accessed. Examples of virtual things include multimedia content and application software.
- A device is a piece of equipment with the mandatory capabilities of communication and optional capabilities of sensing, actuation, data capture, data storage and data processing.
- The devices collect various kinds of information and provide it to the information and communication networks for further processing. Some devices also execute operations based on information received from the information and communication networks.
- Fig. 1.1.1 shows evolutionary phase of internet.

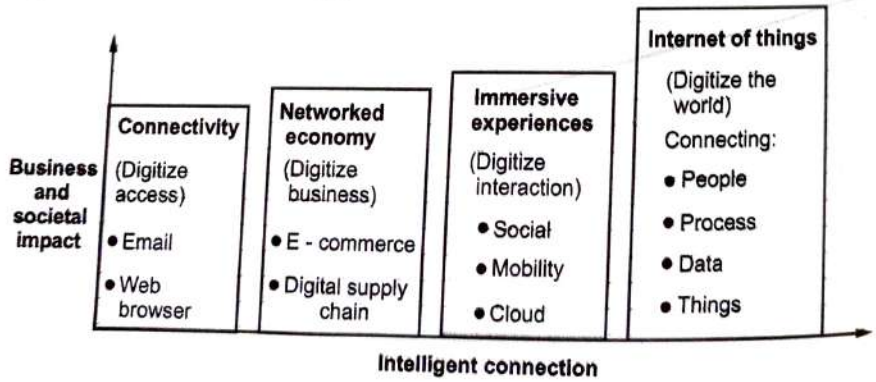
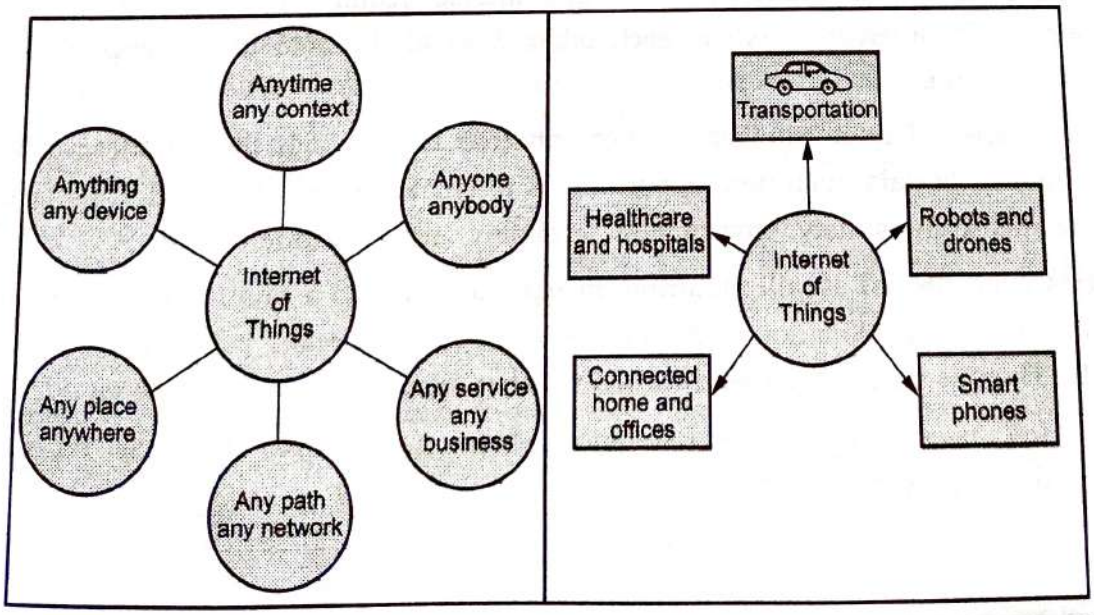


Fig. 1.1.1 : Evolutionary phase of internet

- Evolutionary phase of internet is Connectivity, Networked Economy, Immersive Experiences and IoT.
  1. Connectivity : in the phase, peoples are connected to email, web services and searches the information.
  2. Networked Economy : this phase support e-commerce and supply chain enhancements along with collaborative engagement to drive increased efficiency in business processes.
  3. Immersive Experiences: this phase extended the Internet experience to encompass widespread video and social media while always being connected through mobility.
  4. Internet of Things: it adds connectivity to objects and machines in the world around us to enable new services and experiences.

1.1.1 Definition of IoT

- The Internet of Things (IoT) is the network of physical objects i.e. devices, vehicles, buildings and other items embedded with electronics, software, sensors, and network connectivity that enables these objects to collect and exchange data.
- Wikipedia definition : The Internet of Things, also called The Internet of Objects, refers to a wireless network between objects, usually the network will be wireless and self-configuring, such as household appliances.
- WSIS 2005 Definition : By embedding short-range mobile transceivers into a wide array of additional gadgets and everyday items, enabling new forms of communication between people and things, and between things.
- The Internet of Things refers to the capability of everyday devices to connect to other devices and people through the existing Internet infrastructure.



- Devices connect and communicate in many ways. Examples of this are smart phones that interact with other smart phones, vehicle-to-vehicle communication, connected video cameras, and connected medical devices. They are able to communicate with consumers, collect and transmit data to companies, and compile large amounts of data for third parties.
- IoT data differs from traditional computing. The data can be small in size and frequent in transmission. The number of devices, or nodes, that are connecting to the network are also greater in IoT than in traditional PC computing.
- Machine-to-Machine communications and intelligence drawn from the devices and the network will allow businesses to automate certain basic tasks without depending on central or cloud based applications and services.
- IoT impacts every business. Mobile and the Internet of Things will change the types of devices that connect into a company's systems. These newly connected devices will produce new types of data.
- The Internet of Things will help a business gain efficiency, harness intelligence from a wide range of equipment, improve operations and increase customer satisfaction.
- Ubiquitous computing, pervasive computing, Internet Protocol, sensing technologies, communication technologies, and embedded devices are merged together in order to form a system where the real and digital worlds meet and are continuously in symbiotic interaction.
- The smart object is the building block of the IoT vision. By putting intelligence into everyday objects, they are turned into smart objects able not only to collect information from the environment and interact /control the physical world, but also to be interconnected, to each other, through Internet to exchange data and information.
- The expected huge number of interconnected devices and the significant amount of available data open new opportunities to create services that will bring tangible benefits to the society, environment, economy and individual citizens.
- However, the IoT is still maturing, in particular due to a number of factors, which limit the full exploitation of the IoT. Some of the factors are listed below :
  1. There is no unique identification number system for object in the world.
  2. IoT uses Architecture Reference Model (ARM) but there is no further development in ARM.

3. Missing large-scale testing and learning environments
4. Difficulties in exchanging of sensor information in heterogeneous environments.
5. Difficulties in developing business which embraces the full support of the Internet of Things.

1.1.2 IoT Characteristics

1. Interconnectivity : Everything can be connected to the global information and communication infrastructure.
2. Heterogeneity : Devices within IoT have different hardware and use different networks but they can still interact with other devices through different networks.
3. Things-related services : Provides things-related services within the constraints of things, such as privacy and semantic consistency between physical and virtual thing.
4. Dynamic changes : The state of a device can change dynamically, thus the number of devices can vary.
5. Integrated into information network : IoT devices are integrated with information network for communication purpose. It will exchange data with other devices.
6. Self-adapting: Self-Adaptive is a system that can automatically modify itself in the face of a changing context, to best answer a set of requirements.
7. Self-configuration primarily consists of the actions of neighbour and service discovery, network organization, and resource provisioning.

1.1.3 Component of IoT

- The hardware utilized in IoT systems includes devices for a remote dashboard, devices for control, servers, a routing or bridge device, and sensors. These devices manage key tasks and functions such as system activation, action specifications, security, communication, and detection to support-specific goals and actions.
- Major components of IoT devices are as follows :

1. **Control units** : A small computer on a single integrated circuit containing processor core, memory and a programmable I/O peripheral. It is responsible for the main operation.
2. **Sensor** : Devices that can measure a physical quantity and convert it into a signal, which can be read and interpreted by the microcontroller unit. These devices consist of energy modules, power management modules, RF modules, and sensing modules. Most sensors fall into 2 categories : Digital or analog. An analog data is converted to digital value that can be transmitted to the Internet.

a. Temperature sensors : accelerometers
b. Image sensors : gyroscopes
c. Light sensors : acoustic sensors
d. Micro flow sensors : humidity sensors
e. Gas RFID sensors : pressure sensors

3. **Communication modules** : These are the part of devices and responsible for communication with rest of IoT platform. They provide connectivity according to wireless or wired communication protocol they are designed. The communication between IoT devices and the Internet is performed in two ways :

a) There is an Internet-enable intermediate node acting as a gateway;

b) The IoT Device has direct communication with the Internet.

- The communication between the main control unit and the communication module uses serial protocol in most cases.

4. **Power Sources** : In small devices the current is usually produced by sources like batteries, thermocouples and solar cells. Mobile devices are mostly powered by lightweight batteries that can be recharged for longer life duration.

- **Communication Technology and Protocol** : IoT primarily exploits standard protocols and networking technologies. However, the major enabling technologies and protocols of IoT are RFID, NFC, low-energy Bluetooth, low-energy wireless, low-energy radio protocols, LTE-A, and WiFi-Direct. These technologies support the specific networking functionality needed in an IoT system in contrast to a standard uniform network of common systems.

1.1.4 Working of IoT

- Fig 1.1.2 shows working of IoT.

1. **Collect and transmit data** : The device can sense the environment and collect information related to it and transmit it to a different device or to the Internet.
2. **Actuate device based on triggers** : It can be programmed to actuate other devices based on conditions set by user.
3. **Receive information** : Device can also receive information from the network.
4. **Communication assistance** : It provides communication between two devices of same network or different network.

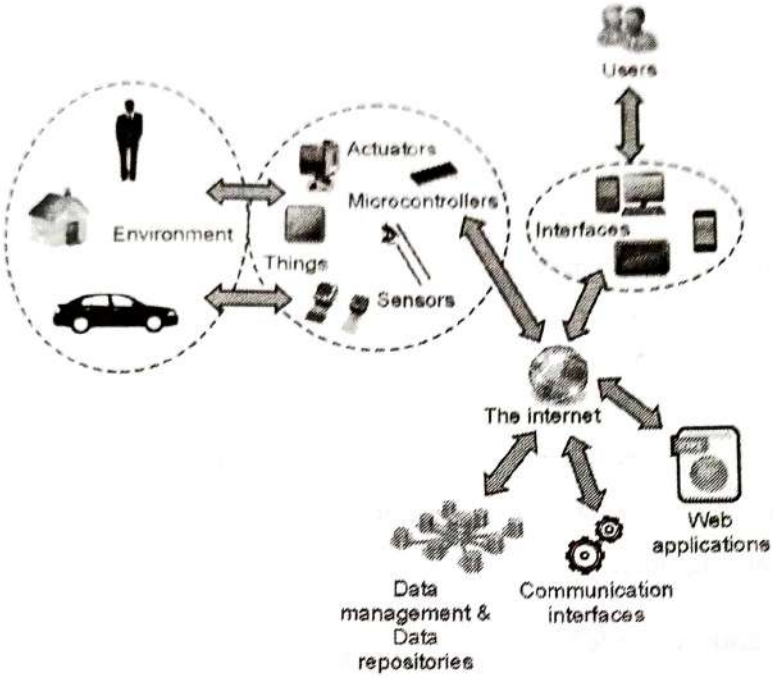


Fig. 1.1.2 : Working of IoT

- Sensors for various applications are used in different IoT devices as per different applications such as temperature, power, humidity, proximity, force etc.
- Gateway takes care of various wireless standard interfaces and hence one gateway can handle multiple technologies and multiple sensors.
- The typical wireless technologies used widely are 6LoWPAN, Zigbee, Zwave, RFID, NFC etc. Gateway interfaces with cloud using backbone wireless or wired technologies such as WiFi, Mobile , DSL or Fibre.

1.1.5 Advantages and Disadvantages

Advantages of IoT

1. Improved customer engagement and communication.
2. Support for technology optimization
3. Support wide range of data collection
4. Reduced waste

Disadvantages of IoT

1. **Loss of privacy and security** : As all the household appliances, industrial machinery, public sector services like water supply and transport, and many other devices all are connected to the Internet, a lot of information is available on it. This information is prone to attack by hackers.
2. **Flexibility** : Many are concerned about the flexibility of an IoT system to integrate easily with another.
3. **Complexity** : The IoT is a diverse and complex network. Any failure or bugs in the software or hardware will have serious consequences. Even power failure can cause a lot of inconvenience.
4. **Compatibility** : Currently, there is no international standard of compatibility for the tagging and monitoring equipment.
5. Save time and money.

1.1.6 Applications of IoT

1. **Home** : Buildings where people live. It controls home and security systems.
2. **Offices** : Energy management and security in office buildings; improved productivity, including for mobile employees.
3. **Factories** : Places with repetitive work routines, including hospitals and farms; operating efficiencies, optimizing equipment use and inventory.
4. **Vehicles** : Vehicles including cars, trucks, ships, aircraft, and trains; condition-based maintenance, usage-based design, pre-sales analytics
5. **Cities** : Public spaces and infrastructure in urban settings; adaptive traffic control, smart meters, environmental monitoring, resource management.
6. **Worksites** : It is custom production environments like mining, oil and gas, construction; operating efficiencies, predictive maintenance, health and safety.

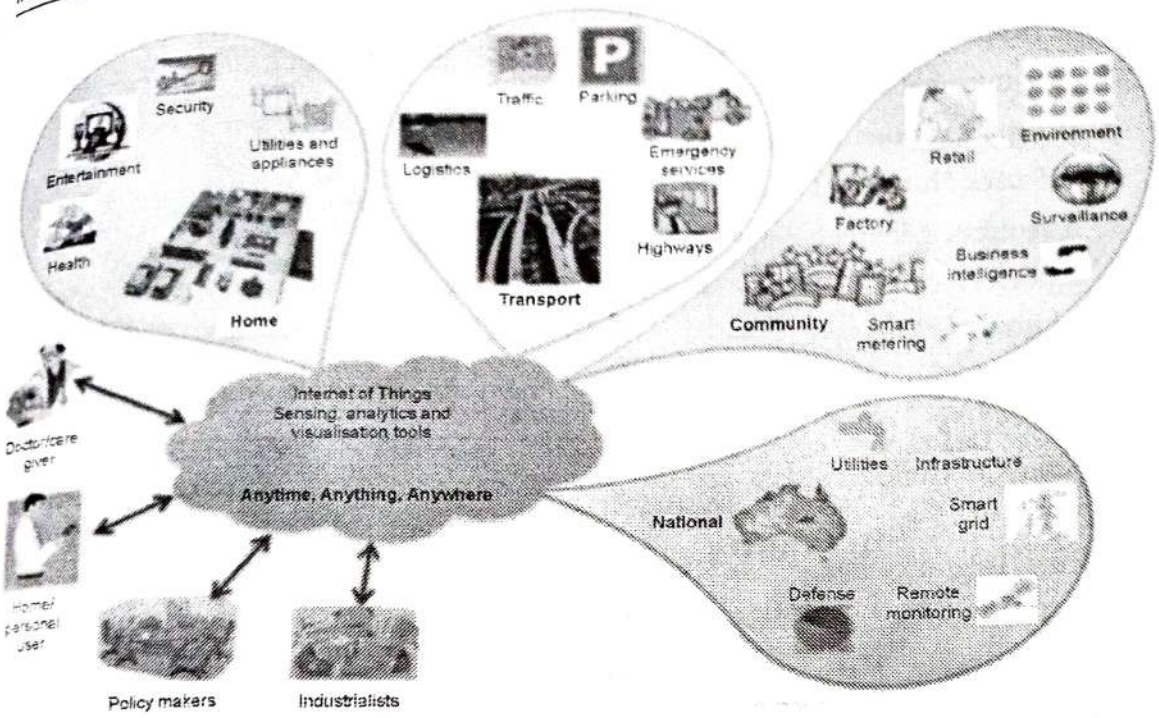


Fig. 1.1.3

1.2 M2M and IoT

- Machine to Machine (M2M) communication is the communication among the physical things which do not need human intervention.
- M2M communication is a form of data communication that involves one or more entities that do not necessarily require human interaction or intervention in the process of communication. M2M is also named as Machine Type Communication (MTC) in 3GPP.
- M2M communication could be carried over mobile networks (e.g. GSM-GPRS, CDMA EVDO networks). In the M2M communication, the role of mobile network is largely confined to serve as a transport network.
- M2M is only a subset of IoT. IoT is a more encompassing phenomenon because it also includes Human-to-Machine communication (H2M).
- Radio Frequency Identification (RFID), Location-Based Services (LBS), Lab-on-a-Chip (LOC), sensors, Augmented Reality (AR), robotics and vehicle telematics, which are some of the technology innovations that employ both M2M and H2M communications.

- Reasons for shifting from M2M to IoT :

1. It supports multiple application with multiple device.
2. It is information and service centric.
3. It supports open market place.
4. IoT uses Horizontal enabler approach.
5. It requires generic commodity devices.
6. Used in B2B and B2C.

Key features of M2M :

1. Low Mobility : M2M Devices do not move and if moves only within a certain area.
2. Time Controlled : data can be sent or receive only at certain pre-defined time periods.
3. Time Tolerant : Sometimes data transfer can be delayed.
4. Packet Switched : Network operator to provide packet switched service
5. Online small Data Transmissions : Devices frequently send or receive small amounts of data.
6. Low Power Consumption : To improve the ability of the system to efficiently service M2M applications.
7. Location Specific Trigger : Intending to trigger M2M device in a particular area e.g. wake up the device.

Six Pillars of M2M :

- The six pillars of M2M are as follows :
  1. Remote monitoring is a generic term most often representing supervisory control, data acquisition and automation of industrial assets.
  2. RFID is a data-collection technology that uses electronic tags for storing data.
  3. A sensor network monitors physical or environmental conditions, with sensor nodes acting cooperatively to form/maintain the network.
  4. The term smart service refers to the process of networking equipment and monitoring it at a customer's site so that it can be maintained and serviced more effectively.
  5. Telematics to the integration of telecommunications and infomatics, but most often it refers to tracking, navigation and entertainment applications in vehicles.
  6. Telemetry is usually associated with industrial, medical and wildlife-tracking applications that transmit small amounts of vehicles data.

1.2.1 Architecture and Components of M2M

- Fig. 1.2.1 shows M2M architecture.

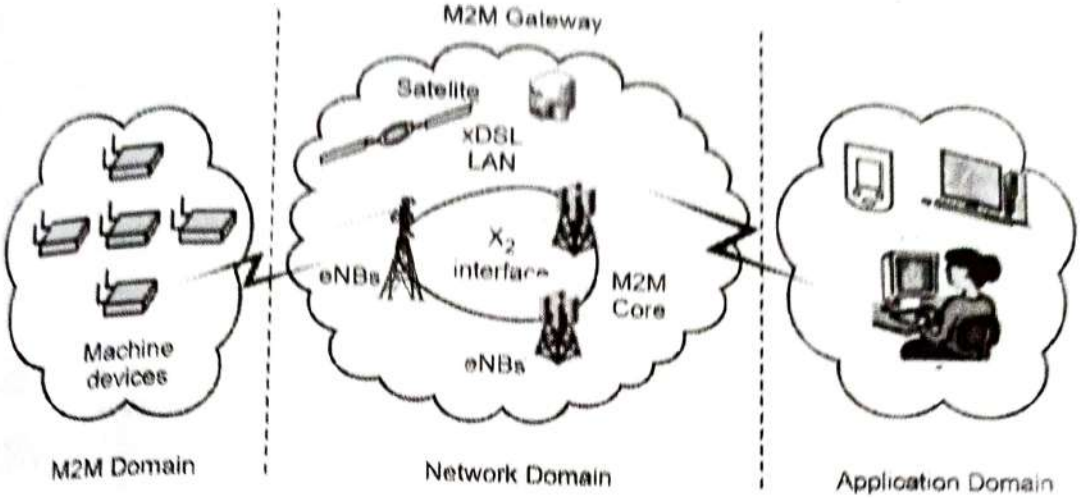


Fig. 1.2.1 : M2M architecture

- The system components of an M2M solution are as follows :

1. M2M Device : A device that runs application(s) using M2M capabilities and network domain functions. An M2M device is either connected straight to an access network or interfaced to M2M gateways via an M2M area network.

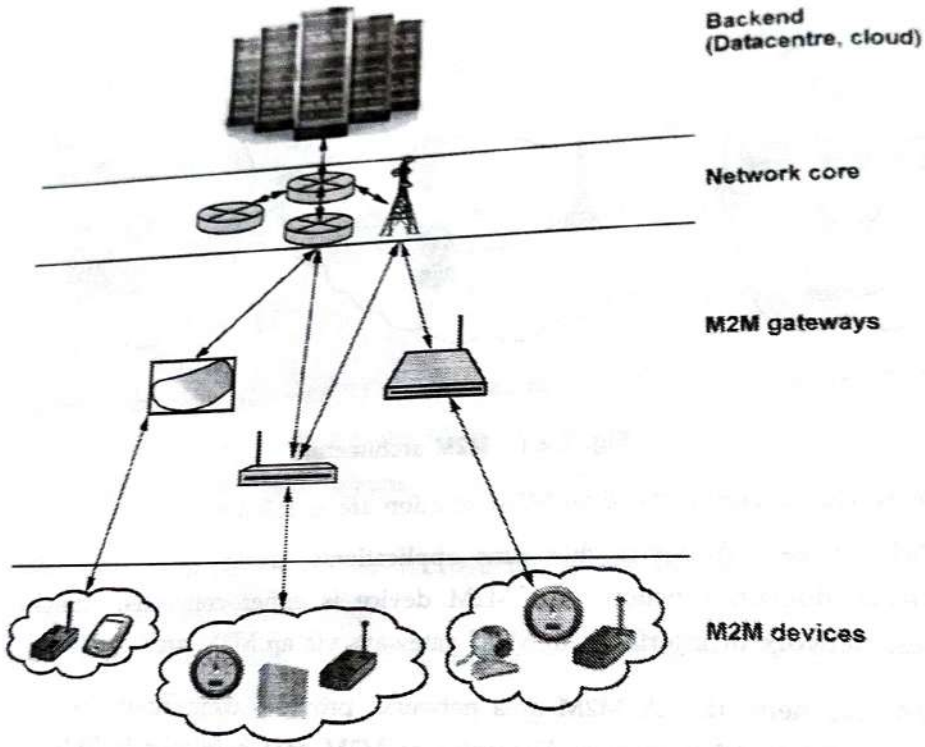
2. M2M area network : A M2M area network provides connectivity between M2M devices and M2M gateways. Examples of M2M area betworks include : Personal area network technologies such as IEEE 802.15, SRD, UWB, Zigbee, Bluetooth, etc or local networks such as PLC, M-BUS, Wireless M-BUS.

3. M2M gateways : Equipments using M2M capabilities to ensure M2M devices interworking and interconnection to the network and application domain. The M2M gateway may also run M2M applications.

4. M2M applications server : Applications that run the service logic and use service capabilities accessible via open interfaces.

5. M2M application : The application component of the solution is a realization of the highly specific monitor and control process. The application is further integrated into the overall business process system of the enterprise.

- Fig. 1.2.2 shows generic M2M solution.



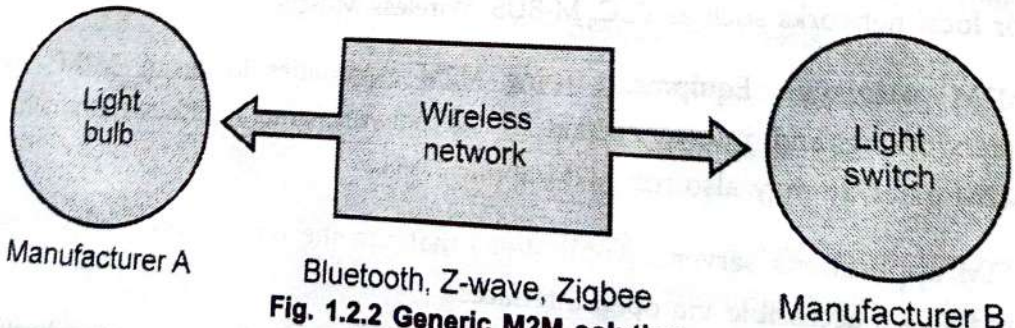


Fig. 1.2.2 Generic M2M solution

- A number of sub-sets of users of M2M services can be identified : Consumers in the home, business users and facility managers, city governments, logistics businesses, energy providers and more.

1.2.2 Difference between M2M and IoT

Machine-to-Machine	Internet of Things
It support single application with single device.	It support multiple application with multiple device.
It is communication and device centric.	It is information and service centric.
It support closed business operations.	It support open market place.
M2M uses vertical system solution approach.	IoT uses horizontal enabler approach.
It requires specialized device solutions.	It requires generic commodity devices.
Used in B2B.	Used in B2B and B2C.

1.3 End to End IoT Architecture

1.3.1 OneM2M Architecture

- Goal of this architecture is to create a common service layer, which can be readily embedded I field devices to allow communication with application servers.
- Fig 1.3.1 shows OneM2M architecture. OneM2M architecture consists of application layer, services layer and network layer.

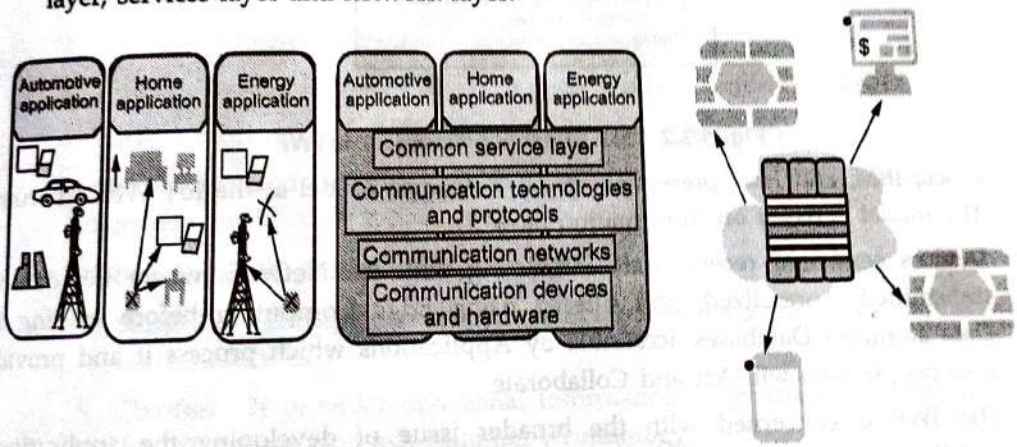


Fig. 1.3.1 : OneM2M architecture

1. Application layer : Comprises oneM2M application and related business and operational logic.
2. Services layer : Consists of OneM2M service function that enables oneM2M applications.
3. Network layer : It provides transport, connectivity and services functions.

1.3.2 IoT World Forum Standardized Architecture

- Fig 1.3.2 shows IoT World Forum (IoTWF) Standardized Architecture.

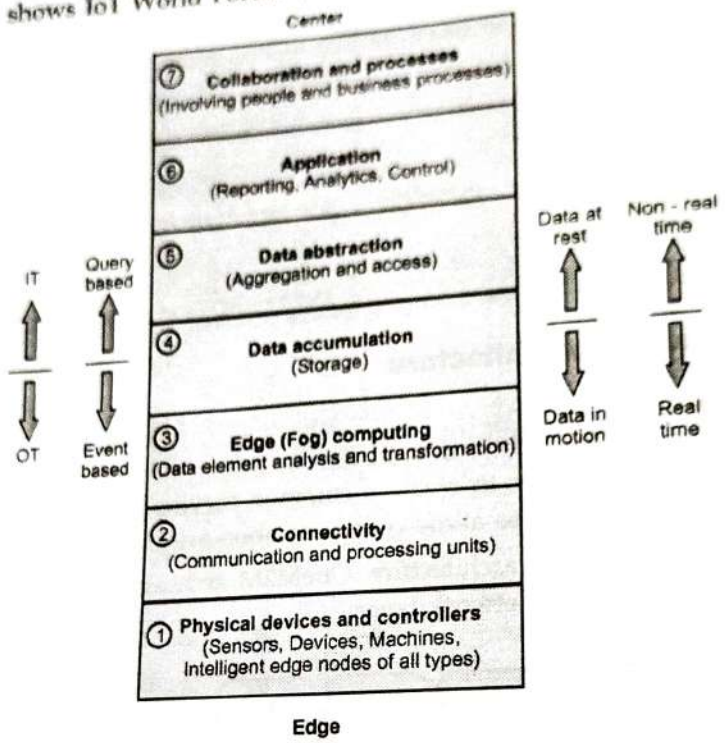


Fig. 1.3.2 : IoT reference model of IoTWF

- Cisco, IBM, and Intel presented an IoT Reference Model at the IoT World Forum. The model is based on "Information Flow".
- Devices send and receive data interacting with the Network where the data is transmitted, normalized, and filtered using Edge Computing before landing in Data storage / Databases accessible by Applications which process it and provide it to people who will Act and Collaborate.
- The IWF is concerned with the broader issue of developing the applications, middleware, and support functions for an enterprise based IoT.
- IoT reference model define a set of levels with control flowing from center to the edge which includes sensors, device, machines and other type of an intelligent nodes.
- Layer 1 : Comprises physical devices and controllers that might control multiple devices. This level enables devices to communicate with one another and to communicate, via the upper logical levels, with application platforms such as computers, remote-control devices, and smart-phones.

- Layer 2 : The IWF model includes gateways in level 2. Because the gateway is a networking and connectivity device, its placement at level 2 seems to make more sense.
- Layer 3 : It performs data element analysis and transformation.
- Layer 4 : The data accumulation level, is where data coming from the numerous devices, and filtered and processed by the edge computing level, is placed in storage that will be accessible by higher levels. This level marks a clear distinction in the design issues, requirements, and method of processing between lower-level (fog) computing and upper-level (typically cloud) computing.
- Layer 5, Data abstraction layer : Reconciles multiple data formats and ensures consistent semantics from various sources. Confirms that data set is complete and consolidates data into one place or multiple data stores using virtualization.
- Layer 6, Application layer : Interprets data using software applications. Applications may monitor, control and provide reports based on the analysis of the data.
- Layer 7, Collaboration and processes layer : Consumes and shares the application information.
- Using this reference model, following things are achieved :
  1. Decompose the problem into smaller parts.
  2. Identify different technologies at each layer.
  3. Define a system in which different parts can be provided by different vendors.
  4. Have a process are defining interface that leads to interoperability.
  5. Define a tiered security model that is enforced at one transition points between levels.

Characteristics of IoTWF model :

- Simplifies : It helps break down complex systems so that each part is more understandable.
- Clarifies : It provides additional information to precisely identify levels of the IoT and to establish common terminology.
- Identifies : It identifies where specific types of processing are optimized across different parts of the system.
- Standardizes : It provides a first step in enabling vendors to create IoT products that work with each other.
- Organizes : It makes the IoT real and approachable, instead of simply conceptual.

1.3.3 Simplified IoT Architecture

- Fig 1.3.3 shows Simplified IoT Architecture.

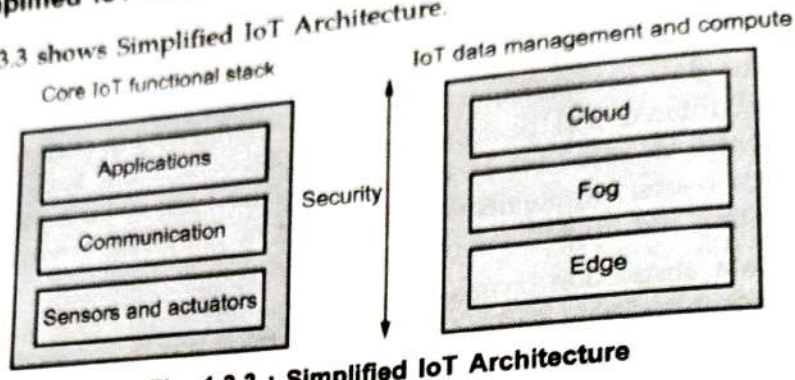


Fig. 1.3.3 : Simplified IoT Architecture

- It consists of Core IoT functional stack group and IoT data management and compute stack.

IoT data management and compute stack :

- Fog computing, or sometimes called edge computing, can be thought of as an extension of the cloud, with the infrastructure distributed at the edge of the network.
- Fog computing facilitates the operation of end devices, typically smart IoT devices, with cloud computing data centres.
- This helps in meeting the needs of high-speed mobile scenarios and geographical distribution scenarios and reduces the bandwidth load of the network core.
- The IoT Cloud Platform represents the software infrastructure and services required to enable an IoT solution.
- An IoT Cloud Platform typically operates on a cloud infrastructure or inside an enterprise data center and is expected to scale both horizontally, to support the large number of devices connected, as well as vertically to address the variety of IoT solutions.
- The IoT Cloud Platform will facilitate the interoperability of the IoT solution with existing enterprise applications and other IoT solutions.

Core IoT functional stack group :

- The IoT gateway acts as the aggregation point for a group of sensors and actuators to coordinate the connectivity of these devices to each other and to an external network.
- An IoT gateway can be a physical piece of hardware or functionality that is incorporated into a larger "Thing" that is connected to the network.

- For example, an industrial machine might act like a gateway, and so might a connected automobile or a home automation appliance.
- An IoT gateway will often offer processing of the data "at the edge" and storage capabilities to deal with network latency and reliability.
- For device to device connectivity, an IoT gateway deals with the interoperability issues between incompatible devices.
- A typical IoT architecture would have many IoT gateways supporting masses of devices.

1.4 Physical Design of IoT

- Physical Design of IoT system refers to IoT Devices and IoT Protocols.

1.4.1 Things in IoT

- IoT devices have unique identity and they are refer as "things" in IoT. Device can perform remote sensing, actuating and monitoring. IoT devices can exchange data between them and process data or send to centralized location for processing and storage. Fig. 1.4.1 shows block diagram of IoT device.

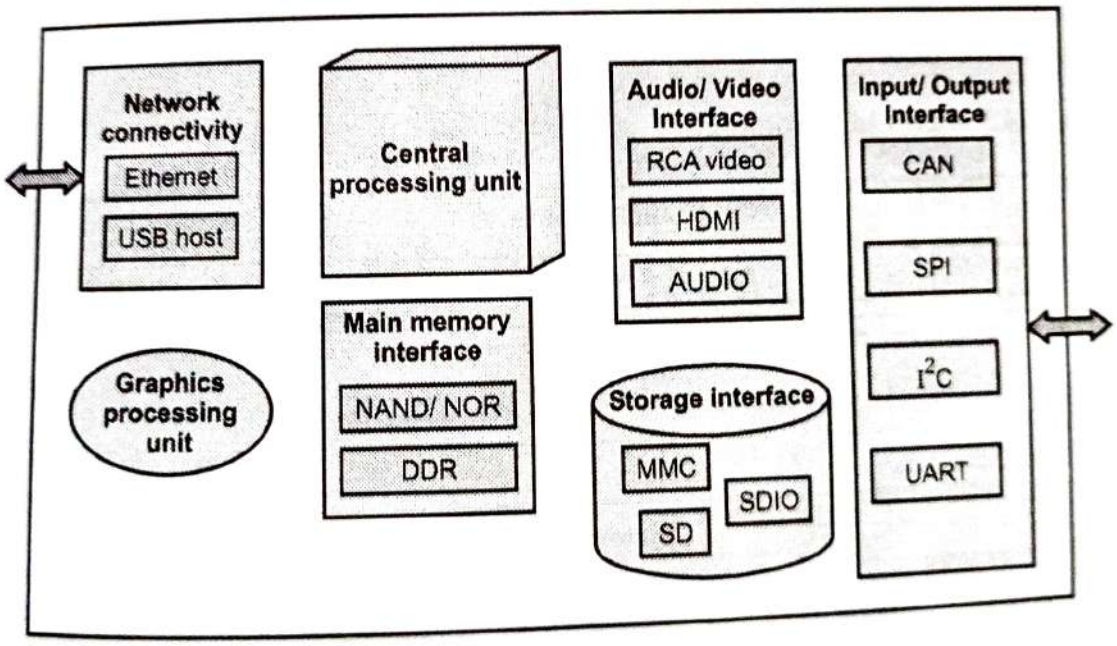


Fig. 1.4.1 Block diagram of IoT devices

- IoT devices provide interface to various wire and wireless devices. Interface includes memory interface, I/O interface for sensors, Internet connectivity interface, storage interface etc.

- Using sensors, IoT collects various information like temperature, light intensity, humidity, air pressure. Some application used cloud based storage. Collected information is stored in cloud and transmitted to other devices.
- Various types of IoT devices are smart clothing, smart watch, wearable sensors, LED lights, automobile industry etc. Fig. 1.4.2 shows IoT devices.

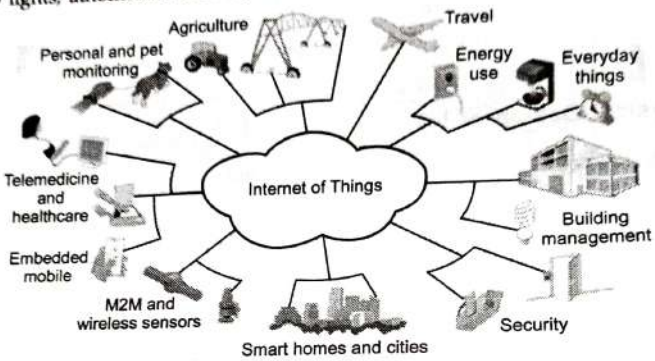


Fig. 1.4.2 IoT devices

1.4.2 IoT Protocol

- Fig. 1.4.3 shows IoT protocols.

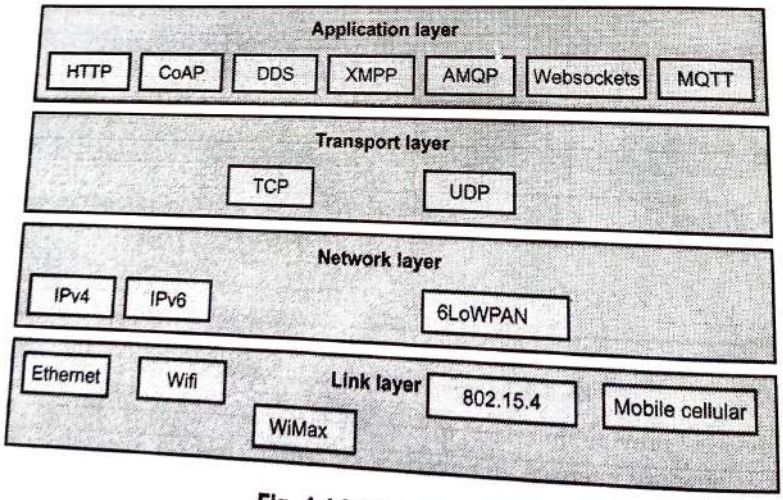


Fig. 1.4.3 IoT protocol

1. Link Layer

- Link layer protocols decide how data is sent on physical medium. Link layer works within the local area network. Protocol of link layer is explained below :

a. 802.3 Ethernet

- This protocol is used for wired medium. Ethernet, in its most basic version runs at 10 Mbit/s. Ethernet has traditionally been used to network enterprise workstations and to transfer non-real-time data.
- The Ethernet standard allows for several different implementations such as twisted pair and coaxial cable. The maximum length of an Ethernet is determined by the nodes' ability to detect collisions.
- The worst case occurs when two nodes at opposite ends of the bus are transmitting simultaneously. Ethernet does not provide any mechanism for acknowledging received frames, making it what is known as an unreliable medium.
- Carrier sense multiple access with collision detection (CSMA/CD) is the most commonly used protocol for LANs. 10BASE5 is generally used as low cost alternative for fiber optic media for use as a backbone segment with in a single building.
- 10BASET is 10 MHz Ethernet running over UTP cable. It also uses passive star topology. The maximum cable segment allowed is 100 - 150 meters. There is no minimum distance requirements between devices, such devices cannot be connected serially but in star wired

b. 802.11 Wifi

- Commonly referred to as Wi-Fi the 802.11 standards define a through-the-air interface between a wireless client and a base station access point or between two or more wireless clients.
- **802.11a :** The 802.11a standard uses the 5 GHz spectrum and has a maximum theoretical 54 Mbps data rate. The 5GHz spectrum has higher attenuation than lower frequencies, such as 2.4 GHz used in 802.11b/g standards. Products with 802.11a are typically found in larger corporate networks or with wireless Internet service providers in outdoor backbone networks.
- **802.11b :** The 802.11 standard provides a maximum theoretical 11 Mbps data rate in the 2.4 GHz Industrial, Scientific and Medical (ISM) band.
- 802.11b uses Complementary Code Keying (CCK) instead of Differential Quadrature Phase Shift Keying (DQPSK) used at lower rates.
- **802.11g :** It provides 20 Mbps and more in the 2.4 GHz band.

c. 802.16 WiMax

- WiMAX refers to broadband wireless networks that are based on the IEEE 802.16 standard, which ensures compatibility and interoperability between broadband wireless access equipment.
- The 802.16 astandard will support OFDM in the 2-to-11 GHz frequency range. The 802.16b standard will operate in the 5 GHz ISM band. A single WiMAX tower can provide coverage to a very large area as big as 3000 square miles
- WiMAX receiver : The receiver and antenna could be a small box or Personal Computer Memory card or they could be built into a laptop the way WiFi access is today.

d. 802.15.4 Zigbee

- In 2002, seeing that neither Wi-Fi nor Bluetooth could not fit some of their needs for embedded systems , a number of industrial companies formed the consortium called ZigBee Alliance, aimed at providing standards for low cost/low consumption wireless communications. Then, with the birth of IEEE 802.15.4 group.
- ZigBee communications can reach up to 500m, with a data rate of up to 250 kbs, for a typical power consumption of 125 to 400 μW.
- As ZigBee is based on IEEE 802.15.4, there is no wake-up signal, but slots for sleep or activity, or in asynchronous mode, devices sleeping anytime they have nothing to say, with an ever-vigilant coordinator.
- To use a ZigBee module with a microcontroller, you need to connect it to a UART. There are other, optional pins to use, including a number of analog inputs / digital IOs and a PWM output indicating the strength of the signal which you can directly connect to a LED pin for observation purposes.
- There are two modes of data transfer namely *Beacon mode* and *Non Beacon mode*.
- In Beacon mode, when the devices are not sending the data they may enter a low power state and reduces the power consumption.
- In Non-beacon mode, the end devices need to be wake up only while sending the data while the routers and coordinators need to be active most of the time.

e. Mobile Communication (2G/3G/4G)

- GSM frequencies originally designed on 900 MHz range, now also available on 800 MHz, 1800 MHz and 1900 MHz ranges. The backbone of a GSM network is a telephone network with additional cellular network capabilities
- 4G is also called as Long Term Evolution. It's promises data transfer rates of 100 Mbps.

2. Network Layer

- The network layer is responsible for the delivery of packets from the source to destination.
- Network layer uses IP address to choose one host among millions of host. In network layer, datagram needs a destination IP address for delivery and a source IP address for a destination reply.

a. IPv4

- IP is used for communicating all Internet enabled devices. The transport layer is responsible for delivery of message from one process to another.
- The network does the host to destination delivery of individual packets considering it as independent packet. But transport layer ensures that the whole message arrives intact and in order with error control and process control.
- An IP address is a numeric identifier assigned to each machine on an IP network. IP address is a software address, not a hardware address, which is hard-coded in the machine or NIC.
- An IP address is made up of 32 bits of information. These bits are divided into four parts containing 8 bit each.
- IPv4 addresses are unique. Two devices on the internet can never have the same address at the same time.
- Packets in the IPv4 layer are called datagrams. A datagram is a variable length.

b. IPv6

- IPv6 addresses are 128 bits in length. Addresses are assigned to individual interface on nodes, not to the node themselves.
- A single interface may have multiple unique unicast addresses. The first field of any IPv6 address is the variable length format prefix, which identifies various categories of addresses.

c. 6LoWPAN

- IPv6 over Low power Wireless Personal Area Network enables IPv6 in low-power and lossy wireless networks such as WSNs.
- 6LoWPAN defines header compression mechanisms.

3. Transport Layer

- A transport layer protocol provides for logical communication between application processes running on different hosts.
- The transport layer is responsible for delivery of message from one process to another. The network does the host to destination delivery of individual packets considering it as independent packet.

- But transport layer ensures that the whole message arrives intact and in order with error control and process control.
- A transport protocol can offer reliable data transfer service to an application even when the underlying network protocol is unreliable, even when the network protocol loses, garbles and duplicate packets.

a. TCP (Transmission Control Protocol)

- TCP is the connection oriented protocol whereas UDP is connectionless protocol. Both are internet protocols used in the transport layer.
- TCP provides a connection-oriented, reliable, byte stream service. The term connection oriented means the two applications using TCP must establish a TCP connection with each other before they can exchange data.
- TCP does not support multicasting and broadcasting. The application data is broken into what TCP considers the best sized chunks to send. The unit of information passed by TCP to IP is called a segment.
- When TCP sends a segment it maintains a timer, waiting for the other end to acknowledge reception of segment. If an acknowledgement isn't received in time, the segment is retransmitted.

b. (UDP) User Datagram Protocol

- UDP is a simple, datagram-oriented, transport layer protocol. This protocol is used in place of TCP.
- UDP is connectionless protocol provides no reliability or flow control mechanisms. It also has no error recovery procedures.
- UDP makes use of the port concept to direct the datagrams to the proper upper-layer applications. UDP serves as a simple application interface to the IP.
- UDP uses port numbers as the addressing mechanism in the transport layer.

4. Application Layer

- Application layer is responsible for accessing the network by user. It provides user interfaces and other supporting services such as e-mail, remote file access, file transfer, sharing database, message handling (X.400), directory services (X.500).

a. HTTP (HyperText Transport Protocol)

- HTTP is an application protocol. HTTP is used to retrieve Web pages from remote servers.
- HTTP uses the services of TCP. HTTP is a stateless protocol. The client initializes the transaction by sending a request message. The server replies by sending a response.

- HTTP includes commands such as GET, PUT, POST, HEAD, DELETE, MOVE, LINK and UNLINK.
- HTTP messages are two types : Request and Response
- URL is a standard for specifying any kind of information on the internet. HTTP uses URL.

b. CoAP - Constrained Application Protocol

- CoAP is a specialized web transfer protocol for use with constrained nodes and constrained (e.g., low-power, lossy) networks.
- CoAP is designed for simplicity, low overhead and multicast support in resource-constrained environments.
- CoAP is a web protocol that runs over the UDP for IoT. Datagram Transport Layer Security (DTLS) is used to protect CoAP transmission.
- The protocol is designed for machine-to-machine (M2M) applications such as smart energy and building automation.
- CoAP provides a request/response interaction model between application endpoints, supports built-in discovery of services and resources, and includes key concepts of the Web such as URIs and Internet media types.
- CoAP is designed to easily interface with HTTP for integration with the Web while meeting specialized requirements such as multicast support, very low overhead, and simplicity for constrained environments.

c. Websocket

- WebSocket is a communications protocol, providing full-duplex communication channels over a single TCP connection. The WebSockets protocol does not run over HTTP, instead it is a separate implementation on top of TCP.
- The WebSocket protocol starts by a handshake in order to start the communication and exchange messages formatted as frames.
- After connection is established, messages can be transmitted, either client or server initiated. This means that you can make a dynamic web page where changes occur in real time.
- In that way Websockest communication presents a suitable protocol for IoT world where changes are usually asynchronously occurring and number of clients can be very large.

d. MQTT (Message Queue Telemetry Transport)

- MQTT is Open Connectivity for Mobile, M2M and IoT. MQTT is designed for high latency, low-bandwidth or unreliable networks. The design principle minimizes the network bandwidth and device resource requirements.

- MQTT is a lightweight broker-based publish/subscribe messaging protocol designed to be open, simple, lightweight and easy to implement.
- The MQTT protocol works by exchanging a series of MQTT control packets in a defined way. Each control packet has a specific purpose and every bit in the packet is carefully crafted to reduce the data transmitted over the network.
- A MQTT topology has a MQTT server and a MQTT client. MQTT control packet headers are kept as small as possible.
- Having a small header overhead makes this protocol appropriate for IoT by lowering the amount of data transmitted over constrained networks.
- MQTT is the protocol built for M2M and IoT which is used to provide new and revolutionary performance.

e. XMPP (Extensible Messaging Presence Protocol)

- The XMPP is targeted at delivering instant messages and presence information. It is an open and XML - Based protocol.
- Instant messaging (IM) is a service, where communicating parties typically end users send messages in one- to-one or one -to -many fashion in near real - time.
- An open technology for real-time communication, which powers a wide range of applications including instant messaging, presence, multi-party chat, voice and video calls, collaboration, lightweight middleware, content syndication, and generalized routing of XML data.
- XMPP support server-to-server communication and client-to-server communication.

f. DDS (Data Distribution Service)

- The first open international middleware standard directly addressing publish-subscribe communications for real-time and embedded systems.
- The DDS is an Object Management Group (OMG) standard for Pub/Sub that addresses the needs of mission and business critical applications, such as, financial trading, air traffic control and management, and complex supervisory and telemetry systems.
- DDS provides a shared "global data space" where any application can publish the data it has & subscribe to the data it needs.
- DDS is highly configurable by means of QoS settings. Heterogeneous systems can be easily accommodated.

g. AMQP (Advanced Message Queuing Protocol)

- A protocol to communicate between clients and messaging middleware servers (brokers). The Broker is the AMQP Server.

- AMQP supports both publish-subscribe model and point-to-point communication, routing and queuing.
- AMQP divides the brokering task between exchanges and message queues, where the first is a router that accepts incoming messages and decides which queues to route the messages to, and the message queue stores messages and sends them to message consumers.
- AMQP supports username and password authentication as well as SASL authorization. It also supports TLS encryption.

1.5 Logical Design of IoT

- It is an abstract representation of an entities and processes but it can not specify low level implementation.

1.5.1 IoT Functional Blocks

- The functional model (FM) is derived from internal and external requirements. Functional view is derived from the Functional Model in conjunction with high-level requirements.
- IoT Functional model identifies Functional Groups (FGs) that is, groups of functionalities, grounded in key concepts of the IoT Domain Model.
- Functional Model is an abstract framework for understanding the main Functionality Groups (FG) and their interactions. This framework defines the common semantics of the main functionalities and will be used for the development of IoT-A compliant Functional Views.
- The Functional Model is not directly tied to a certain technology, application domain, or implementation. It does not explain what the different Functional Components are that make up a certain Functionality Group.
- Fig. 1.5.1 shows IoT functional model.
- The Application, Virtual Entity, IoT Service, and Device FGs are generated by starting from the User, Virtual Entity, Resource, Service, and Device classes from the IoT Domain Model.
- Device functional group contains all the possible functionality hosted by the physical devices. Device functionality includes sensing, actuation, processing, storage, and identification components, the sophistication of which depends on the device capabilities.
- Communication functional group support all the communication used by devices. It uses wired and wireless technology.

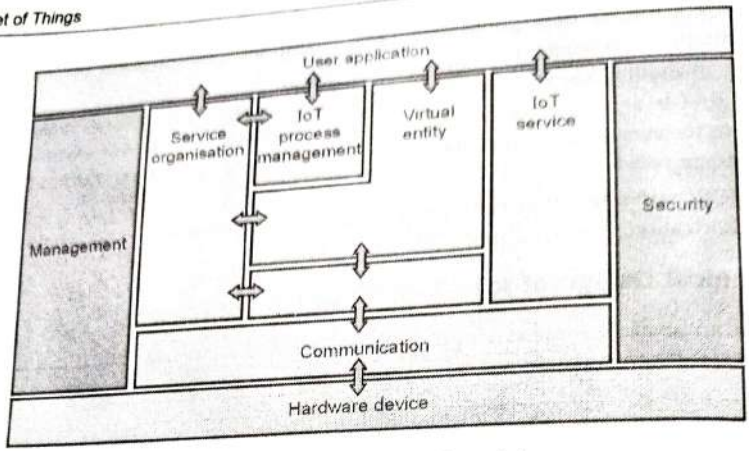


Fig. 1.5.1 : IoT functional model

- IoT Service functional group : Support functions such as directory services, which allow discovery of Services and resolution to resources.
- Virtual Entity functional group : It is related to the Virtual Entity class in the IoT Domain Model. Associations between Virtual Entities can be static or dynamic depending on the mobility of the Physical Entities related to the corresponding Virtual Entities.
- IoT Service Organization functional group : To host all functional components that support the composition and orchestration of IoT and Virtual Entity services.
- Finally, the "Management" transversal FG is required for the management of and/or interaction between the functionality groups.
- The IoT Process Management FG relates to the conceptual integration of (business) process management systems with the IoT ARM.
- The Service Organisation FG is a central Functionality Group that acts as a communication hub between several other Functionality Groups.
- The Virtual Entity and IoT Service FGs include functions that relate to interactions on the Virtual Entity and IoT Service abstraction levels, respectively.
- The Virtual Entity FG 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.
- Communication FG provides a simple interface for instantiating and for managing high-level information flow. It can be customized according to the different requirements defined in the Unified Requirements list.

- The Management FG combines all functionalities that are needed to govern an IoT system. The need for management can be traced back to at least four high-level system goals : Cost reduction; Attending unexpected usage issues; Fault handling and Flexibility.
- The Security Functionality Group is responsible for ensuring the security and privacy of IoT-A-compliant systems. It is in charge of handling the initial registration of a client to the system in a secure manner. This ensures that only legitimate clients may access services provided by the IoT infrastructure.

1.5.2 IoT Communication Model

Request/Response model :

- In the Request/Response model, client requests information from the server and waits till the response is served from the server. Fig. 1.5.2 shows Request/Response model.

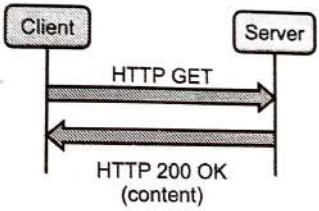


Fig. 1.5.2 Request/Response model

- HTTP protocol is used by Request/Response model. For example, a browser client may request a web page from the server through a "Request" and the corresponding web page will be served by the server as a "Response".
- The client and the server can communicate one to one, or one to many with more requests.
- This model is stateless communication model and each request-response pair is independent of others.

Publish/Subscribe Model:

- Publishers : Publishers generate event data and publishes them.
- Subscribers : Subscribers submit their subscriptions and process the events received
- Publish/Subscribe service : It's the mediator/broker that filters and routes events from publishers to interested subscribers.
- Fig. 1.5.3 shows Publish/Subscribe Model.
- The publishers and subscribers are autonomous, which means that they do not need to know the presence of each other.

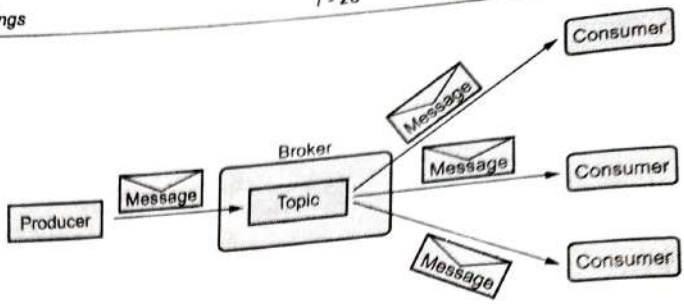


Fig. 1.5.3 Publish/Subscribe model

- This model is highly suited for mobile applications, ubiquitous computing and distributed embedded systems.
- Failure of publishers or subscribers does not bring down the entire system.
- No strong guarantee on broker to deliver content to subscriber. After a publisher publishes the event, it assumes that all corresponding subscribers would receive it.
- Potential bottleneck in brokers when subscribers and publishers overload them.

Push/Pull Model:

- Data procedure push the data to queues and consumers pull the data from the queues. Fig. 1.5.4 shows push-pull model.

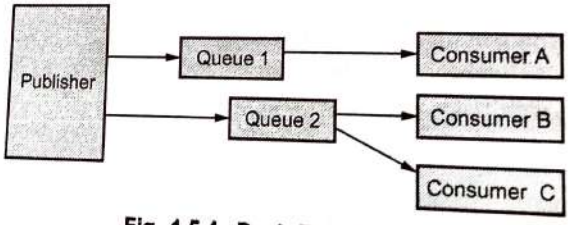


Fig. 1.5.4 Push-Pull model

- Sometimes queue act as buffer in between producer and consumer. Producer does not need to be aware of the consumers.

Exclusive Pair Model:

- This communication model is full duplex, bi-directional communication model. It uses persistent connection between client and server. Fig. 1.5.5 shows exclusive pair model.
- Client send request to server for opening the connection. This connection is open till the client send request for closing the connection.

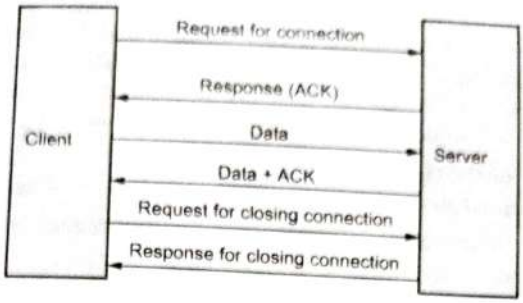


Fig. 1.5.5 Exclusive pair model

1.5.3 IoT Communication API's

1. REST Based Communication API :

- A large part of the interoperability, scale, and control for IoT can be achieved through API management. Standards-based design patterns for Web APIs, API management, and a RESTful architecture provide tremendous value in simplifying the task of interoperability across heterogeneous systems handling vast amounts of data.
- Representational State Transfer (REST) APIs follow the request-response communication model.
- Applications conforming to the REST constraints can be called RESTful. RESTful systems typically communicate over HTTP with the same Methods (GET, POST, PUT, DELETE etc) that browsers use to retrieve web pages and to send data to remote servers.

1. Client-Server : requires that a service offer one or more operations and that services wait for clients to request these operations.
2. Stateless : requires communication between service consumer (client) and service provider (server) to be stateless.
3. Cache : requires responses to be clearly labeled as cacheable or non-cacheable.
4. Uniform Interface : requires all service providers and consumers within a REST-compliant architecture to share a single common interface for all operations.
5. Layered System : requires the ability to add or remove intermediaries at runtime without disrupting the system.
6. Code-on-Demand : allows logic within clients (such as Web browsers) to be updated independently from server-side logic using executable code shipped from service providers to consumers.

- Each client request and server response is a message, and REST-compliant applications expect each message to be self-descriptive. That means each message must contain all the information necessary to complete the task. Other ways to describe this kind of message are "stateless" or "context-free." Each message passed between client and server can have a body and metadata.
- **HTTP request methods and actions :**
  1. GET : return whatever information is identified by the Request-URI
  2. HEAD : identical to GET except that the server must not return a message-body in the response, only the metadata
  3. OPTIONS : return information about the communication options available on the request/response chain identified by the Request-URI
  4. PUT : requests that the enclosed entity be stored under the supplied Request-URI
  5. POST : requests that the origin server accept the entity enclosed in the request as a new subordinate of the resource identified by the Request-URI
  6. DELETE : requests that the origin server delete the resource identified by the Request-URI.
- The first three are read-only operations, while the last three are write operations.

2. WebSocket based Communication API

- WebSocket support full-duplex, two-way communication between client and server.
- WebSocket APIs reduce the network traffic and latency as there is no overhead for connection setup and termination requests for each message.
- WebSocket uses a standard HTTP request-response sequence to establish a connection. When the connection is established, the WebSocket API provides a read and write interface for reading and writing data over the established connection in an asynchronous full duplex manner.
- WebSocket also provides an interface for asynchronously closing the connection from either side.

1.6 Overview of IoT Protocols

- IoT is enabled by several technologies including wireless sensor networks, cloud computing, Big data analytics, Embedded Systems, Security Protocols and architectures, communication protocols, web services, Mobile Internet, and Semantic Search engines.

1.6.1 Cloud Computing

- Cloud computing has the almost unlimited capacity of storage and processing power which is a more mature technology at least to a certain extent to solve the problem of most of the Internet of things.
- Cloud computing is a pay-per-use model for enabling available, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal management effort or service-provider interaction.
- Cloud storage services may be accessed through a web service API, a cloud storage gateway or through a web-based user interface.
- **Cloud computing services are offered to users in different forms :** Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS).
- **Software as a Service (SaaS) :** Model in which an application is hosted as a service to customers who access it via the Internet. The provider does all the patching and upgrades as well as keeping the infrastructure running. The traditional model of software distribution, in which software is purchased for and installed on personal computers.
- **Platform as a Service (PaaS) :** Platform as a service is another application delivery model and also known as cloud-ware. Supplies all the resources required to build applications and services completely from the Internet, without having to download or install software. Services includes application design, development, testing, deployment, and hosting, team collaboration, web service integration, database integration, security, scalability, storage, state management, and versioning. PaaS is closely related to SaaS but delivers a platform from which to work rather than an application to work with.
- **Infrastructure as a Service (IaaS) :** SaaS and PaaS are providing apples to customers, IaaS doesn't. It offers the hardware so that your organization can put whatever they want onto it. Rather than purchase servers, software, racks, and having to pay for the datacenter space for them, the service provider rents for resources like server space, network equipment, memory etc.

1.6.2 Big Data Analytic

- A category of technologies and services where the capabilities provided to collect, store, search, share, analyze and visualize data which have the characteristics of high-volume, high-velocity and high-variety.

- Examples of big data generated by IoT systems :
  a) Weather Monitoring Stations
  b) Machine sensor data from industrial systems
  c) Health and fitness data
  d) Location and tracking systems

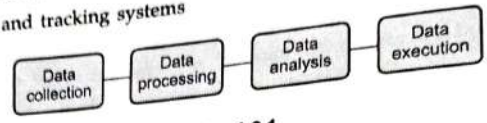


Fig. 1.6.1

1.6.3 Wireless Sensor Networks

- A Wireless Sensor Network (WSN) is a network formed by a large number of sensor nodes where each node is equipped with a sensor to detect physical phenomena such as light, heat, pressure, etc.
- WSNs nowadays usually include sensor nodes, actuator nodes, gateways and clients. A large number of sensor nodes deployed randomly inside of or near the monitoring area, form networks through self-organization.
- Sensor nodes monitor the collected data to transmit along to other sensor nodes by hopping. During the process of transmission, monitored data may be handled by multiple nodes to get to gateway node after multi-hop routing, and finally reach the management node through the internet or satellite.
- Standards for WSN technology have been well developed, such as Zigbee (IEEE 802.15.4). The IEEE 802.15.4 is simple packet data protocol for lightweight wireless networks.
- It works well for long battery life, selectable latency for controllers, sensors, remote monitoring and portable electronics.

1.6.4 Communication Protocols

- Communication protocols are used as a backbone of IoT systems. It enable network connectivity and coupling to applications. Communication protocols allow devices to exchange data over the network.
- Communication protocol also performs error correction and detection, flow control, data encoding, addressing mechanism etc.
- Sequence control, lost of packet, retransmission are the other functions of communication protocol.

1.6.5 Embedded System

- A system is a set of interacting or interdependent component parts forming a complex unit. It is a fixed plan to perform one or many task.
- Embedded system is an electronic system which is designed to perform one or a limited set of functions using software and hardware.
- General definition of embedded systems is embedded systems are computing systems with tightly coupled hardware and software integration that are designed to perform a dedicated function. The word embedded reflects the fact that these systems are usually an integral part of a larger system, known as the embedding system. Multiple embedded systems can coexist in an embedding system.
- An embedded system has three main components: Hardware, Software and time operating system.
- Hardware parts includes power supply, processor, memory, times & communication ports, system application specific circuit etc.
- Software parts includes the application software is required to perform the series of tasks. An embedded system has software designed to keep in view of three constraints :
  a. Availability of System Memory
  b. Availability of processor speed
  c. The need to limit power dissipation when running the system continuously in cycles of wait for events, run , stop and wake up.
- Demand for low cost and higher density platform requires the integration of devices. As integration levels increases, more and more logic is added to the processor die, creating families of applications specific service processors.
- System-on-Chip (SoC) designs increasingly become the driving force of a number of modern electronics systems. A number of key technologies integrate together in forming the highly complex embedded platform.

1.7 IoT Levels and Deployment Templates

- IoT system consists of following components
  1 Device : IoT device performs identification, remote monitoring, sensing and actuating functions.
  2 Resource : IoT device used software components as resources for accessing, processing, and storing sensor information. It also controls actuators.
  3 Controller Service : It sends data from the device to the web service and receives commands from the application for controlling the device.
  4 Database : IoT generates large amount of local database and cloud database.

5. Web Service : Web services act as a link between the IoT device, application, database and analysis components.
6. Analysis Component : It is responsible for analysing the IoT data and generate results.
7. Application : User use this interface for controlling and monitoring various IoT system.

- Levels of IoT system are IoT Level-1, IoT Level-2, IoT Level-3, IoT Level-4, IoT Level-5, IoT Level-6.
- IoT Development level are six types.

IoT Level 1	Single node, perform sensing, perform analysis and hosts the application
IoT Level 2	Single node, perform sensing, perform local analysis
IoT Level 3	Single node, data is analyzed in cloud and application is cloud based
IoT Level 4	Multiple Node, perform local analysis, application is cloud based
IoT Level 5	Multiple end node and coordinator node,
IoT Level 6	Multiple independent node, perform sensing, send data to cloud

1.7.1 IoT Level 1

- Physical devices and controllers that might control multiple devices. These are the "things" in the IoT, and they include a wide range of endpoint devices that send and receive information.
- Fig. 1.7.1 shows IoT level 1.

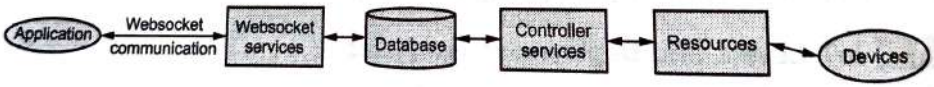


Fig. 1.7.1 IoT level 1

- Level 1 IoT systems are suitable for modeling low-cost and low complexity solutions where the data involved is not big and the analysis requirements are not computationally intensive.
- Eg : Home automation
- The system consists of a single node that allows controlling the lights and appliances in a home remotely. The device used in the system interfaces with the

light and appliances using electronic relay switches. The status information of each light or appliance is maintained in the local database. The controller service continuously monitors the state of each light or appliance and triggers the relay switches accordingly.

1.7.2 IoT Level 2

- In level 2, single node performs sensing and local analysis. Fig. 1.7.2 shows block diagram.

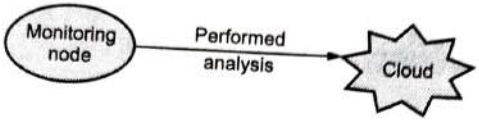


Fig. 1.7.2

- Both data and application is stored on the cloud. This type of system is suitable for big data used for analysis and that to performed on local side.
- Smart irrigation is an example of level 2 IoT system. System uses single sensor for monitoring the soil moisture level and controls the irrigation system.
- Controller service monitors continuously the moisture level. If the moisture level drops below a threshold, the irrigation system is turned ON.

1.7.3 IoT Level 3

- Single node is used in level 3 internet of things. Collected data is stored on the cloud and processed on the cloud. Application is also stored on the cloud.
- This is suitable for huge data and analysis requirement are computationally intensive.

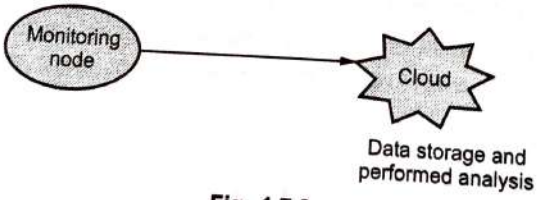


Fig. 1.7.3

- REST or websocket is used for communicating with client machine. Consider an example of tracking package handling.
- Single node monitors the vibration level for the package being shipped. For monitoring vibration level, accelerometer and gyroscope sensor is used.

- Accelerometer and gyroscope sensor is used. System allows shippers tracking cargo to communicate with the devices, changing reporting frequencies and investigating alerts generated by attached sensors.
- Controller device sends the sensor data to the cloud in real time using WebSocket service. Data is stored on the cloud and analysis is also performed on the cloud.
- Cloud can send alerts if the vibration levels is greater than threshold level.

1.7.4 IoT Level 4

- Level 4 IoT system uses multiple nodes (sensors) and processing is performed on local node. Data is stored on the cloud and application is stored on the cloud storage.
- Fig. 1.7.4 shows block diagram of Level 4 IoT.

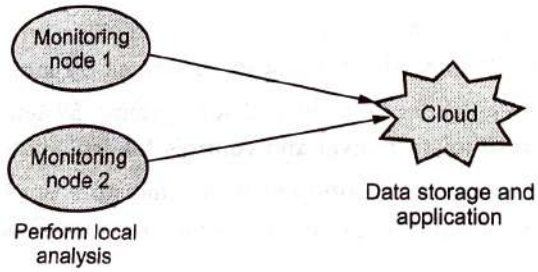


Fig. 1.7.4

- Level 4 IoT uses two observer sensor for local and cloud. These sensor subscribe and receive information form the IoT device to cloud .
- The observer sensor can process information and used for various purposes. Control function is not performed by this observer sensor.
- Noise monitoring system is used IoT level 4. Sensors are not depends upon each other. Each sensor runs its own controller service that sends the data to the cloud.

1.7.5 IoT Level 5

- It contains multiple end sensor and one coordinator sensor. Fig. 1.7.5 shows IoT level 5 block diagram.
- The end sensor perform sensing and/or actuation. Coordinator sensor collect data from the end sensor and sends to the cloud.
- Data is stored and analysis in the cloud and application is also cloud based. Forest fire detection system uses level 5 IoT system.

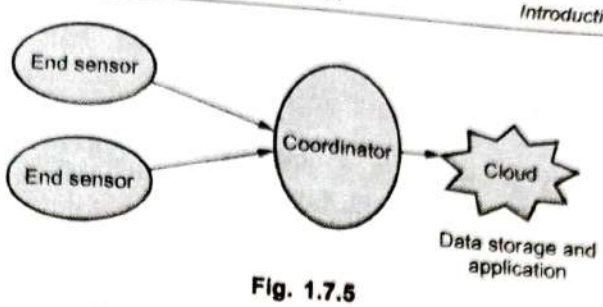


Fig. 1.7.5

- Multiple sensor are kept at different location to monitor temperature, humidity, carbon dioxide level. Coordinator node (sensor) collect data for end node and these nodes act as gateway and provides Internet services to the IoT systems.

1.7.6 IoT Level 6

- It contains multiple independent end nodes and it performs sensing and/or actuation function. It sends data to the cloud. Fig. 1.7.6 shows block diagram of level 6 IoT.
- Data is stored on the cloud and application is also cloud based. Result is displayed on the cloud. Centralized controller knows the status of monitoring node and sends control commands to the nodes.
- Weather monitoring system uses Level 2 IoT based system. Multiple nodes are kept at different location to monitor temperature, humidity level.

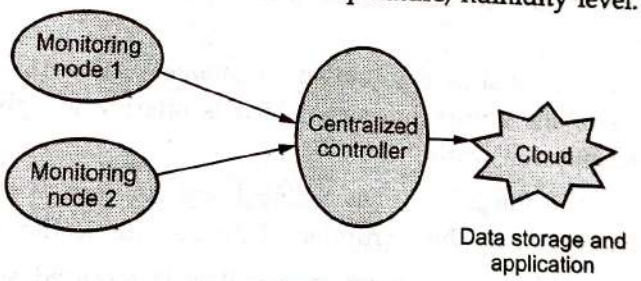


Fig. 1.7.6

- End node send real time data to cloud using WebSocket service. Cloud database is used for storing data. Data analysis is performed on cloud side. Cloud based application is used for display data.

1.8 Challenges for IoT

- IoT and ubiquitous integration of clinical environments define complex design challenges and requirements in order to reach a suitable technology maturity for its wide deployment and market integration.

- From the beginning, IoT devices present inherited challenges since they are constrained devices with low memory, processing, communication and energy capabilities.
  1. The first key challenge for a ubiquitous deployment is the integration of multi-technology networks in a common all-IP network to ensure that the communication network is reliable and scalable. For this purpose, IoT relies on the connectivity and reliability for its communications on Future Internet architecture.
  2. The second key challenge is to guarantee security, privacy, integrity of information and user confidentiality. The majority of the IoT applications need to take into considerations the support of mechanisms to carry out the authentication, authorization, access control, and key management.
  3. In addition, due to the reduced capabilities from the constrained devices enabled with Internet connectivity, a higher protection of the edge networks needs to be considered with respect to the global network.
  4. The third key challenge is to offer support for the mobility, since the Future Internet presents a more ubiquitous and mobile Internet. Mobility support increases the applicability of Internet to new areas.
- The most present nowadays are mobile platforms such as smart phones and tablets which enable a tremendous range of applications based on ubiquitous location, context awareness, social networking, and interaction with the environment.
- Future Internet potential is not limited to mobile platforms, else IoT is another emerging area of the Future Internet, which is offering a high integration of the cybernetic and physical world.
- Mobility support in the IoT enables a global and continuous connection of all the devices without requiring the disruption of the communication sessions.
- For example, mobility management in hospitals is required since clinical devices can be connected through wireless technologies. Mobility offers highly valuable features such as higher quality of experiences for the patients, since this allows the patients to move freely, continuous monitoring through portable/wearable sensors, extend the coverage within all the hospital, and finally a higher fault tolerance since the mobility management allows the connection to adapt dynamically to different access points.
- Therefore, clinical environment is one of the main scenarios where the mobility for the IoT-based applications exploit these capabilities, in terms of fault tolerance influences directly in the life support, and continuous monitoring influences the quantity of data available which is required for real-time diagnostic.

5. Other challenges are also arising from the application, economical, and technological perspectives. For example, from an application point of view are the requirements for processing large amounts of data for a growing number of devices, it is called Big Data.

- From the economic points of view, the needs to provide economies of scale, i.e., new services based on existing modules in order to leverage the related platform investment.
- From the networking point of view to offer an end-to-end support for Quality of Service , since the different IoT applications will present different requirements in terms of latency and bandwidth, for example, for clinical environments the traffic should be prioritized over other non-critical traffic coming from smart-metering.
- Fig. 1.8.1 shows the key challenges to offer an Internet of Everything. This covers from the integration of heterogeneous devices to the integration into a Web of Things.

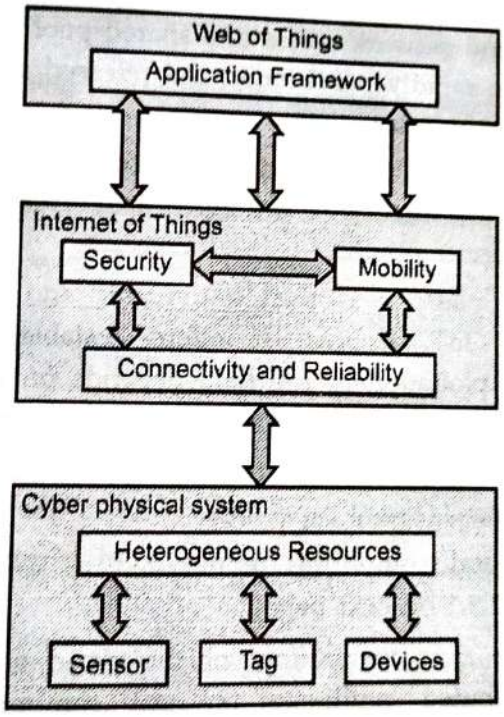


Fig. 1.8.1 Key challenges to offer an Internet of Everything

- The number of devices that are connected to the Internet is growing exponentially. This has led to defining a new conception of Internet, the commonly called Future Internet, which started with a new version of the Internet Protocol (IPv6) that extends the addressing space in order to support all the emerging Internet-enabled devices.

- IoT devices are small wireless devices that would be placed in public places. Wireless communication is made secure through encryption technique. But the IoT devices are very small and not powerful enough to support encryption methods. There is need to modify encryption algorithm in order to support IoT devices.
- In IoT, the different devices are traceable through the interconnected network, it creates threats to personal and private data.
- Many IoT devices are small in size and do not have the continuous power source. A device computation depends on battery size and cost of the device. Many IoT devices work as a single, limited purpose which could have customized network interfaces, operating systems, and programming models that make the most efficient use of limited computation, network, and energy resources.

1.9 Interdependencies of IoT and Cloud Computing

- According to the NIST definition, Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction.
- The Cloud paradigm can be delivered using essentially three different service models. These are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).
- A Cloud-based IoT platform is then a dynamic and flexible resource sharing platform delivering IoT services. It offers scalable resources and services management. The exploitation of IoT data depends on massive resources, which should be available when needed and scaled back when not needed.
- Cloud technologies offer decentralized and scalable information processing and analytics, and data management capabilities.
- Fig. 1.9.1 shows cloud computing represented as a stack of service offering categories. (See Fig. 1.9.1 on next page)
- Cloud Infrastructure : At the bottom of the cloud stack, Cloud Infrastructure provides the distributed multi-site physical components to support cloud computing, such as storage and processing resources. This layer allows the infrastructure provider to abstract away details such as which exact hardware an application is using and which data center the application is running in.
- Cloud Platform : Platform offerings provide an infrastructure for developing and operating web-based software applications. Examples include facilities for application design, application development, testing, deployment, and hosting, as well as application services such as team collaboration, security, application versioning, and application instrumentation.

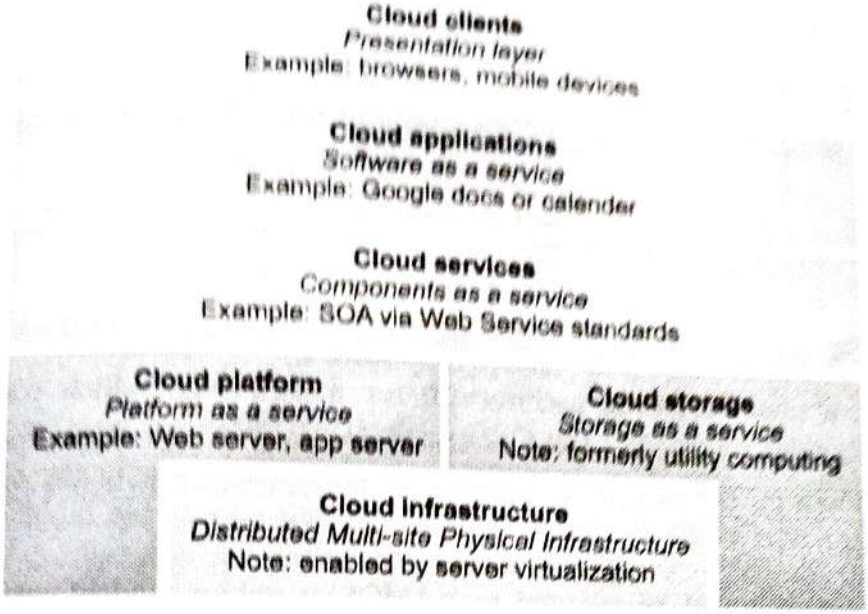


Fig. 1.9.1 : Cloud computing represented as a stack of service offering catagories

- Cloud Storage : Building upon the Cloud Infrastructure, this layer of the cloud stack is focused on the incremental renting of storage on the Internet.
- Cloud Services : This layer of the cloud computing stack includes the definition of software components, run in a distributed fashion, across the commercial Internet.
- Cloud Applications : This definition relies on the cloud for access to what would traditionally be local desktop software. For example, Adobe's Photoshop, a program to manipulate images, was distributed to end users on disks for many years. Today, you still can install a version of Photoshop from an installation disk, or you can go to a completely online version of an analogous application, entitled Express.
- Cloud Clients : Another application-related function of cloud computing focuses on the distribution of business and personal data across servers on the Internet.

1.9.1 Cloud Middleware

- Middleware helps in reducing the overhead of virtualization. Fig. 1.9.2 shows Multitiered cloud architecture based on middleware.

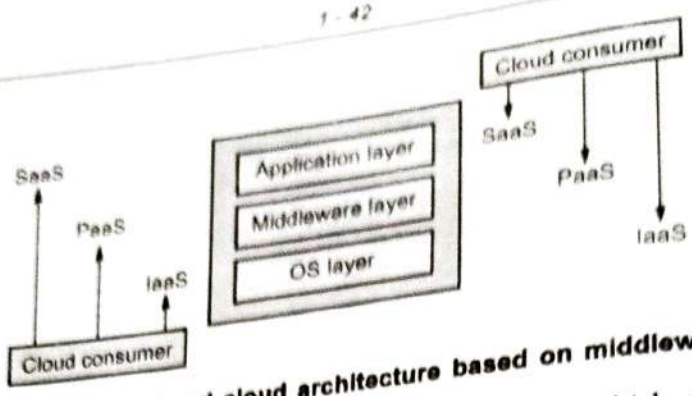


Fig. 1.9.2 : Multitiered cloud architecture based on middleware

- VAMOS, a novel software architecture for middleware, which runs middleware modules at the hypervisor level. VAMOS reduces I/O virtualization overhead by cutting down on the overall number of guest/hypervisor switches for I/O intensive workloads.
- Middleware code can be adapted to VAMOS at only a modest cost, by exploiting existing modular design and abstraction layers. Applying VAMOS to a database workload improved its performance by up to 32 %.
- VAMOS does not require changes in the guest OS and take advantage of the host OS by running selected middleware modules directly at the hypervisor level, minimizing the adaption cost and re-using existing code.
- The parallel computing environments such as PVM and MPI are (HPC) middleware by definition. The HPC middleware fills the gap that the operating systems and the programming languages lack to support parallel computing.
- Grid computing is the foundation of cloud computing infrastructure, so grid middleware is the basis of IaaS middleware.
- The grid computing middleware software will manage and execute all the activities related to identification, allocation, de-allocation and consolidation of all the computing resources to the end-users transparently, as in the case of a geographical distributed resources system.
- Grid computing environment is a necessity to many end-users who cannot afford huge computational resources, both hardware and software. Therefore, any large corporate body or government organization having a large geographical spread will be essentially required to set up at least some kind of grid computing environment, so that the expensive resources of their grid can be shared and effectively utilized by all the end users.

- The PaaS middleware is often referred to as the cloud middleware that underpins and supports the SaaS applications.
- Provisions and manages cloud infrastructure and middleware; provides development, deployment and administration tools. Infrastructure providers can transparently alter the platforms for their customers' unique needs.
- Users have to develop, test, deploy and manage applications hosted in a cloud environment. Example: Google App Engine, SalesForce.
- The cloud middleware consists of two kinds of middleware: IaaS and PaaS middlewar

1. Infrastructure as a Service (IaaS) : The raw computing resources are exposed to the consumers. There are options for including support software but usually there is no abstraction over the system complexities. This is in fact exactly the same as getting access to a physical computer attached to the Internet. Amazon EC2 is the leading IaaS provider and the most prominent cloud service provider today.
2. Platform as a Service (PaaS) : Consumers get access to an application development platform possibly hosted on a infrastructure cloud. The platform completely abstracts the complexities of the underlying host system. The platform also guarantees load balancing and scaling in a transparent manner to the cloud consumer. However these platforms typically are restrictive and requires the hosted applications to strictly adhere to certain specialized API, frameworks, and programming languages. Google AppEngine is an example of a PaaS.

1.9.2 Cloud Standards

- Cloud computing standardization organizations
  1. NIST : Working definition of cloud computing
  2. Distributed Management Task Force
  3. Cloud Management Working Group
  4. European Telecommunications Standards Institute
  5. Standards Acceleration to Jumpstart Adoption of Cloud Computing
  6. Open Cloud Consortium

- Cloud Standardization Efforts are listed below :

Project Name	URL	Focus
CloudAudit, also known as Automated Audit, Assertion, Assessment, and Assurance API (A6)	http://www.cloudaudit.org	• Open, extensible, and secure interface, namespace, and methodology for cloud-computing providers and their authorized consumers to automate the audit, assertion, assessment, and assurance of their environments • Part of the Cloud Security Alliance since October 2010
Cloud Computing Interoperability Forum	http://www.cloudforum.org	• Common, agreed-on framework/ontology for cloud platforms to exchange information in a unified manner • Sponsors of the Unified Cloud Interface Project to create an open and standardized cloud interface for the unification of various cloud APIs
Cloud Security Alliance	http://cloudsecurityalliance.org	• Recommended practices for cloud-computing security • Working on Version 3 of the Security Guidance for Critical Areas of Focus in Cloud Computing • Nonprofit organization that includes Google, Microsoft, Rackspace, Terre-mark, and others

Cloud Standards Customer Council	http://cloudstandardscustomercouncil.org	• Standards, security, and interoperability issues related to migration to the cloud. • End-user advocacy group sponsored by the Object Management Group (OMG) and creator of the Open Cloud Manifesto.
Cloud Storage Initiative	http://www.snia.org/cloud	• Adoption of cloud storage as a new delivery model (Data-Storage-as-a-Service) • Initiative sponsored by the Storage Net-working Industry Association (SNIA), the creator and promoter of the Cloud Data Management Interface (CDMI) • SNIA includes members from NetApp, Oracle, and EMC
DeltaCloud	http://incubator.apache.org/deltacloud	• Abstraction layer for dealing with differences among IaaS providers • API based on representational state transfer (REST) with a small number of operations for managing instances • Currently has libraries for seven providers including Amazon EC2, Eucalyptus, and Rackspace
Distributed Management Task Force (DMTF)	http://dmtf.org/standards/cloud	• Management interoperability for cloud systems • Developer of the Open Virtualization Framework (OVF) • Runs the Open Cloud Standards Incubator

IEEE P2301, Guide for Cloud Portability and Interoperability Profiles	http://standards.ieee.org/develop/project/2301.html	• Standards-based options for application interfaces, portability interfaces, management interfaces, interoperability inter-faces, file formats, and operation conventions
IEEE P2302, Draft Standard for Inter-cloud Interoperability and Federation	http://standards.ieee.org/develop/project/2302.html	• Protocols for exchanging data, programmatic queries, functions, and governance for clouds sharing data or functions or for federating one cloud to another
Open Cloud Computing Interface	http://occi-wg.org	• REST-based interfaces for management of cloud resources including computing, storage, and bandwidth • Working group of the Open Grid Forum
Open Cloud Consortium	http://opencloudconsortium.org	• Frameworks for interoperating between clouds and operation of the Open Cloud Testbed

1.9.3 The Cloud of Things Architecture

- Cloud computing creates a new way of designing, developing, testing, deploying running and maintaining applications on the Internet. The Internet of Things and Cloud computing are both emerging technologies and have their own features.
- Fig. 1.9.3 shows the main features of the Cloud-based IoT platform.
- CloudThings architecture is an online platform that allows system integrators and solution providers to leverage a complete. Things application infrastructure for developing, deploying, operating, and composing Things applications and services that consist of three major modules :
  1. The CloudThings service platform for Things is a set of Cloud services (IaaS), allowing users to run any applications on Cloud hardware. The CloudThings service platform for Things dramatically simplifies the application development, eliminates need for infrastructure development, shortens time to market, and reduces Things management and maintenance costs.

- The CloudThings service platform offers users unique device management capabilities. It communicates directly with devices and provides storage to collect Things data and transmit Things events. Vast amount of sensor data can be processed, analyzed, and stored using the computational and storage resources of the Cloud.
- The CloudThings service platform allows sharing of sensor resources by different users and applications under a flexible usage mode.

2. The CloudThings Developer Suite for Things is a set of Cloud service tools (PaaS) for Things application development. These tools include open Web service Application Programming Interfaces (APIs), which provide complete development and deployment capabilities to Things developers.

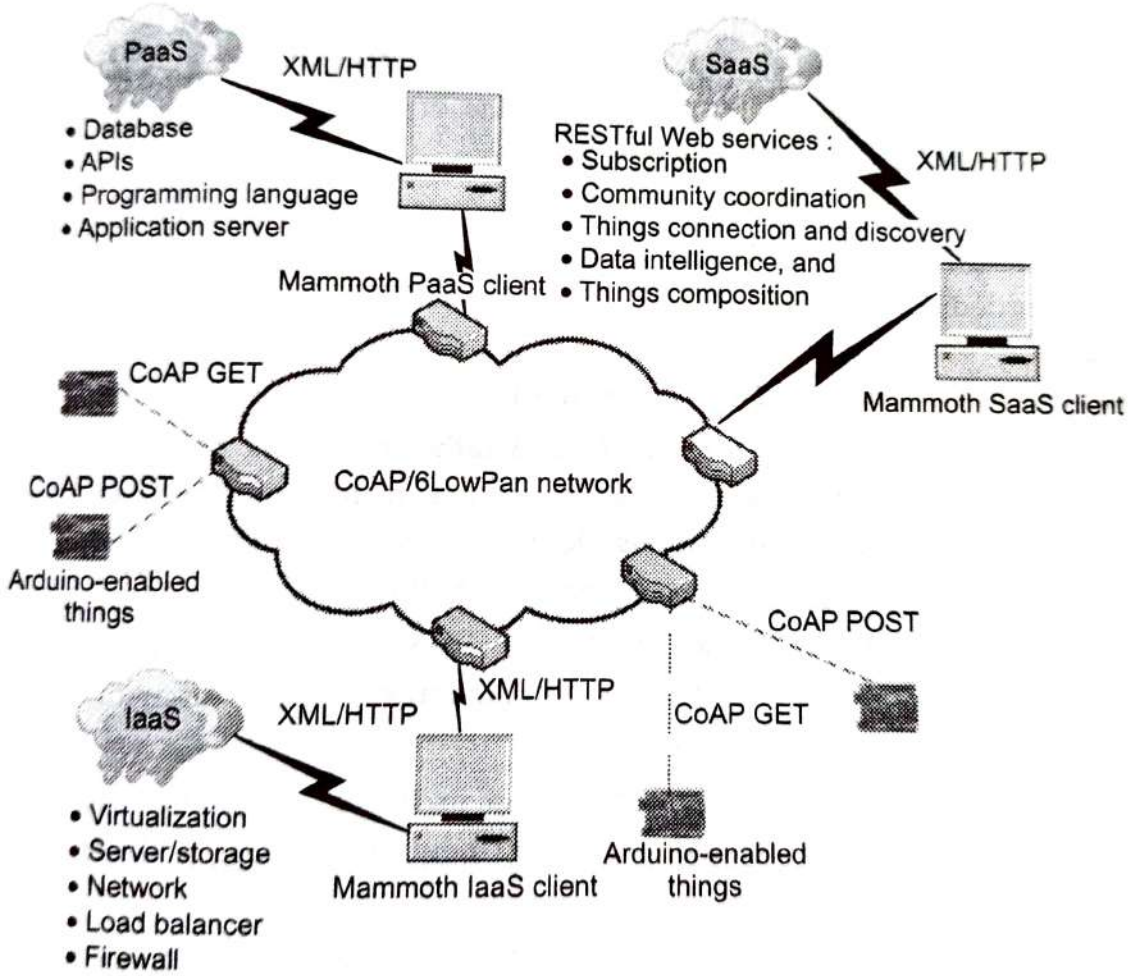


Fig. 1.9.3 : Cloud-based IoT platform

3. The CloudThings Operating Portal for Things is a set of Cloud services (SaaS) that support deployment and handle or support specialized processing services including service subscription management, community coordination, Things connection, Things discovery, data intelligence, and Things composition.

1.10 Web of Things

- Web of Things provides an Application Layer that simplifies the creation of Internet of Things applications. The Web, is a system of interlinked documents accessed via the Internet. The Web was originated from Tim Berners-Lee around 1990.
- The Web, like Email, is one of the services that runs on the Internet. Key components of web are as follows :
  1. Uniform Resource Locator (URL) & Uniform Resource Identifier (URI)
  2. HyperText Markup Language (HTML)
  3. Hypertext Transfer Protocol (HTTP)
- Internet is the term used to identify the massive interconnection of computer networks around the world. It refers to the physical connection of the paths between two or more computers. World Wide Web is the general name for accessing the Internet via HTTP. It is just one of the connection protocols that is available in the Internet, and not the only one.
- Unlike in the many systems that exist for the Internet of things, the Web of Things is about re-using the Web standards to connect the quickly expending eco-system of Embedded devices built into everyday smart objects. Well-accepted and understood standards and blueprints (such as URI, HTTP, REST, RSS, etc.) are used to access the functionality of the smart objects.
- WoT uses HTTP as an application protocol rather than as a transport protocol as done in the world of Web Services. Exposes the synchronous functionality of smart objects through a REST interface (also known as RESTful API) and more generally respects the blueprints of Resource Oriented Architectures.
- Exposes the asynchronous functionality (i.e. events) of smart objects through the use of largely accepted Web syndication standards such as Atom or server-push Web mechanisms such as Comet.
- The Web of Things is the next logical step in this IoT evolution toward global networks of sensors and actuators, enabling new applications and providing new opportunities.
- WoT has a flat architecture, compared to the traditional server-client architecture. To directly integrate things to the Web, it is first required that all the things must be addressable, i.e. everything must have an IP address, or must be IP-enabled when connected to the Internet. WoT also requires connectivity and interoperability at the application layer.
- The WoT applications are Arduino, Japan Geiger Map, Nanode, National Weather Study Project and AgSphere.

1. Arduino is an open-source electronics platform based on easy-to-use hardware and software.
2. Regulations per the Japanese Nuclear Safety Commission prescribe some standards that a monitoring system at a power producing nuclear plant must adhere to.
3. Nanode is an open- source Arduino- like board that has built-in web connectivity.
4. The National Weather Study Project (NWSP) is a large- scale environmental study project deploying hundreds of mini weather stations in schools throughout Singapore.

1.10.1 Two Pillars of The Web

- An application server is a software framework or middleware that provides an environment in which applications can run, no matter what the applications are or what they do. An application server acts as a set of components accessible to the software developer through an API defined by the middleware itself.
- For web applications, these components are usually performed in the same machine where the web server is running, and their main job is to support the construction of dynamic web pages.
- The application server is based on the three-tiered or multi-tiered software architecture. A "tier" can also be referred to as a "layer". Fig. 1.10.1 shows three tired architecture.

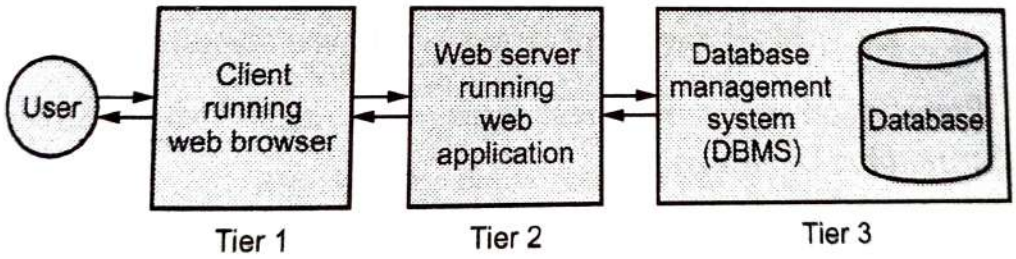


Fig. 1.10.1 : Three tired architecture

- Collectively, three-tier architectures are programming models that enable the distribution of application functionality across three independent systems, typically :
  1. Client components running on local workstations (tier one)
  2. Processes running on remote servers (tier two)
  3. A discrete collection of databases, resource managers, and mainframe applications (tier three)

- First tier : Responsibility for presentation and user interaction resides with the first-tier components. These client components enable the user to interact with the second-tier processes in a secure and intuitive manner.
- Second tier : The second-tier processes are commonly referred to as the application logic layer. These processes manage the business logic of the application, and are permitted access to the third-tier services. The application logic layer is where most of the processing work occurs.
- Third tier : The third-tier services are protected from direct access by the client components residing within a secure network. Interaction must occur through the second-tier processes.
- Communication among tiers : All three tiers must communicate with each other. Open, standard protocols and exposed APIs simplify this communication. You can write client components in any programming language, such as Java or C++. These clients run on any operating system, by speaking with the application logic layer. Databases in the third tier can be of any design, if the application layer can query and manipulate them. The key to this architecture is the application logic layer.
- The Java EE standard - based application server architecture is shown Fig. 1.10.2.

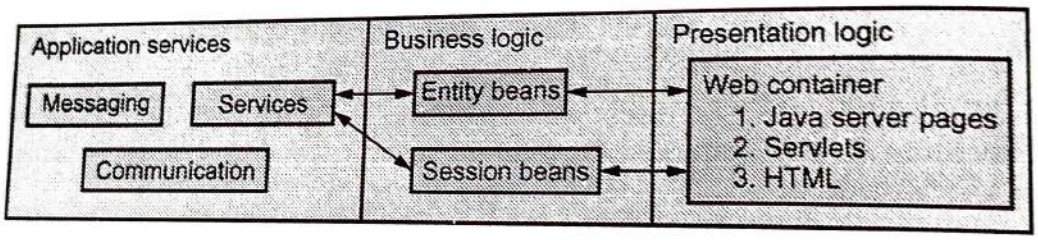


Fig. 1.10.2 : Java application server architecture

1. The Presentation layer requires skills such as HTML, CSS and possibly JavaScript, plus UI design.
2. The Business layer requires skills in a programming language so that business rules can be processed by a computer.
3. The Data Access layer requires SQL skills in the form of Data Definition Language (DDL) and Data Manipulation Language (DML), plus database design.

- Advantages of the 3 Tier Architecture :
  1. Flexibility - By separating the business logic of an application from its presentation logic, a 3-Tier architecture makes the application much more flexible to changes.
  2. Maintainability - Changes to the components in one layer should have no effect on any others layers.

3. Reusability - Separating the application into multiple layers makes it easier to implement re-usable components..
4. Scalability - A 3-Tier architecture allows distribution of application components across multiple servers thus making the system much more scalable.
5. Reliability - A 3-Tier architecture, if deployed on multiple servers, makes it easier to increase reliability of a system by implementing multiple levels of redundancy.

- Fig. 1.10.3 shows two pillars of the web.
- In two pillars of the web, one pillar is HTTP, HTML, URL and application server and web browser is second pillar.
- Communications protocol is a language of digital message formats and rules for exchanging those messages in or between computing systems and/ or in telecommunications. Protocols may include signaling, authentication, and error detection and correction capabilities.

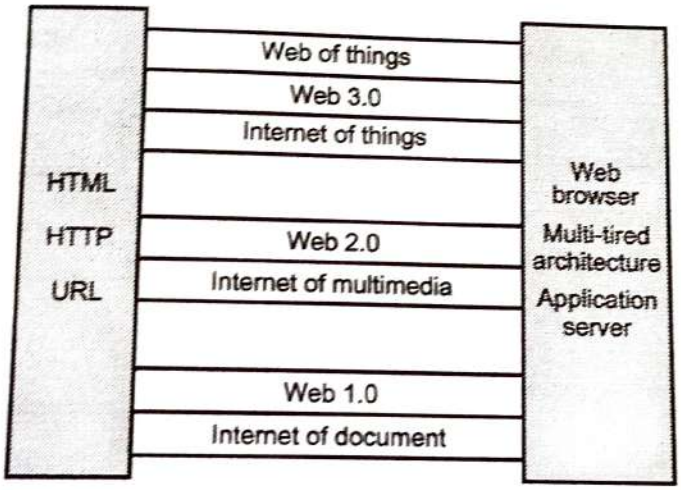


Fig. 1.10.3 : two pillars of the web

1.10.2 Architecture Standardization for WoT

- The W3C Web of Things (WoT) is intended to enable interoperability across IoT Platforms and application domains. Primarily, it provides mechanisms to formally describe IoT interfaces to allow IoT devices and services to communicate with each other, independent of their underlying implementation, and across multiple networking protocols. Secondarily, it provides a standardized way to define and program IoT behavior.
- Day by day, the number of connected Things is growing exponentially. Standardized solutions provided by the rapid evolution of the Internet have laid the foundation of what we call the Web of Things. Therefore, WoT architectures

aim to integrate everyday objects with web technologies. Those devices should be able to communicate with each other using existing web standards.

1.10.3 Platform Middleware for IoT

1. Standards for M2M

- The European Telecommunications Standards Institute (ETSI) produces globally-applicable standards for Information and Communications Technologies, including fixed, mobile, radio, converged, broadcast and Internet technologies.
- A new ETSI Technical Committee is developing standards for M2M Communications. This group aims to provide an end-to-end view of M2M standardization cooperating with ETSI's activities on Next Generation Networks and 3GPP standards initiative for mobile communication technologies.
- Fig. 1.10.4 shows the high-level ETSI M2M architecture.

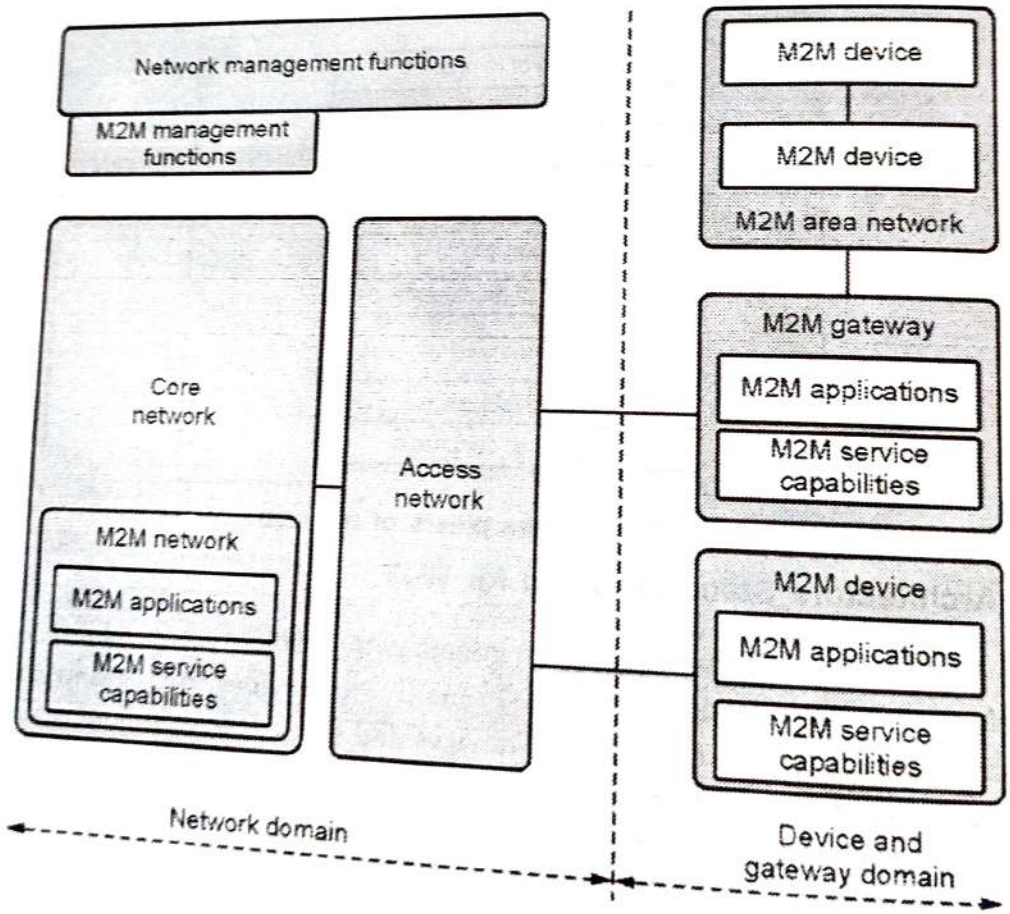


Fig. 1.10.4

- A high-level architecture of M2M system consists of a Device and Gateway Domain, and a Network Domain. key elements of the ETSI M2M architecture are as follows :
  1. **M2M Device** runs M2M Device Applications (DA) using M2M Device Service Capabilities Layer (DSCL).
  2. **M2M Gateway** runs M2M Gateway Applications (GA) using M2M Gateway Service Capabilities Layer (GSCL).
  3. **M2M Area Network** provides connectivity based on Personal or Local Area Network technologies (e.g. Zigbee, Bluetooth) between M2M devices and M2M gateways. The case of device-to-device communication is out of the scope of ETSI's effort.
- Network domain contains the following elements :
  1. **M2M Access Network** : It allows M2M devices and M2M gateways to communicate with the core network. It uses any one network solutions: Digital Subscriber Line (DSL), satellite, GSM EDGE Radio Access Network (GERAN), Universal Terrestrial Radio Access Network (UTRAN), evolved UTRAN (eUTRAN), Wi-Fi (IEEE 802.11), and Worldwide Interoperability for Microwave Access (WiMAX), that can be optimized for M2M communication if needed.
  2. **M2M Core Network** : This network enables interconnection with other networks, provides IP connectivity or other connectivity options, service and control functions, and roaming. Similarly to access network, it can be based on varied existing core networking solutions.
  3. M2M Network Service Capabilities Layer (NSCL) : It provides M2M functions that are shared by different M2M applications.
  4. **M2M Applications** run the service logic and use M2M service capabilities available via open interfaces.
  5. **M2M network** management functions consist of all the functions required to manage access and core networks.
  6. M2M management functions consist of all the functions used to facilitate the bootstrapping of permanent M2M service layer security credentials required to manage M2M service capabilities in the network domain.
- The M2M platform middleware normally covers the layers from M2M gateway to the M2M application server. Several (Open Mobile Alliance) OMA standards provide building blocks that map into the ETSI M2M framework :
  1. Device management can provide ETSI's remote entity management service
  2. Gateway management object fulfills some ETSI gateway service requirements
  3. Firmware updates, software updates, provisioning, diagnostics, and monitoring

4. Converged personal network services maps into ETSI M2M area network
5. Reachability, address mapping, inter/ intra- area- network messaging, service publication and discovery
6. Some OMA enablers (e.g., location) support services that can be used in M2M applications

2. Frameworks for WSN

- The use of sensor data has become very important in a broad range of applications. Often it is highly desirable to combine sensor data with other kinds of geospatial data in order to build more complex systems. To facilitate the interoperable access to sensor data and thus to allow the flexible (domain independent) re-use of sensor data within spatial data infrastructures a working group of the Open Geospatial Consortium (OGC) has developed the Sensor Web Enablement (SWE) framework.
- SWE is a suite of standard encodings and web services that enable the following :
  1. Discovery of sensors, processors, and observation
  2. Taking of sensors or models
  3. Access to observations and observation streams
  4. Publish-subscribe capabilities for alerts
  5. Robust sensor system and process descriptions
- During the development of the SWE framework, several aims had to be taken into account. Especially the following goals were the drivers of the design of the SWE architecture :
  1. Standardized access to sensor measurements
  2. Retrieval of metadata for determining sensor capabilities and the quality/reliability of measurements
  3. Controlling and tasking of sensors
  4. Alerting based on user defined criteria and sensor measurements
  5. Access to sensor parameters and automatic processing of measurements based upon pre-defined processes
- Web service specifications have been produced by the OGC SWE Working Group are as follows :
  1. Sensor observation service : standard web interface for accessing observations and model and requesting acquisitions
  2. Sensor planning service : standard web interface for tasking sensor systems
  3. Sensor alert service : standard web interface for publishing and subscribing to sensor alerts
  4. Web notification service : standard web interface for asynchronous notification

- The USN (Ubiquitous Sensor Networks) standardization of ITU - T is another effort being carried out under the auspices of the Next - Generation Network Global Standards Initiative (NGN - GSI).
- Ubiquitous Sensor Networks is a conceptual framework built over existing physical networks that makes use of sensed data and provide knowledge services. Its main components are as follows :
  1. Sensor Network : Comprising sensors and an independent power source (e.g., battery, solar power). The sensors can then be used for collecting and transmitting information about their surrounding environment;
  2. USN Access Network : Intermediary or "sink nodes" collecting information from a group of sensors and facilitating communication with a control centre or with external entities;
  3. Network Infrastructure : likely to be based on a next-generation network (NGN);
  4. USN Middleware : Software for the collection and processing of large volumes of data;
  5. USN Applications Platform : A technology platform to enable the effective use of a USN in a particular industrial sector or application.

Characteristics of a USN

1. Small-scale sensor nodes;
2. Limited power requirements that can be harvested (e.g., solar power) or stored (e.g., battery);
3. Able to withstand harsh environmental conditions;
4. Fault tolerant and designed to cope with high possibility of node failures;
5. Support for mobility;
6. Dynamic network topology;
7. Able to withstand communication failures;
8. Heterogeneity of nodes;
9. Large scale of deployment

3. Standards for SCADA

- ISO 16100-1:2009, one of the components of ISO 16100 standard for industrial automation systems and controls.

- ANSI/ ISA-95 is an international standard for developing an automated interface between enterprise and control systems. This standard is application to global manufacturers.
- A USA ANSI standard developed by an ISA Committee of volunteer experts :
  1. ANSI/ISA 95.01-2000"Enterprise - Control System Integration - Part 1 : Models and Terminology"
  2. ANSI-ISA 95.02-2001"Enterprise - Control System Integration - Part 2 : Object Attributes"
  3. ANSI/ISA 95.03-2005"Enterprise - Control System Integration - Part 3 : Models of Manufacturing Operations"
  4. ISA-95.04, Object Models & Attributes, Part 4: Object models and attributes for Manufacturing Operations Management.
  5. ANSI/ISA 95.05-2007"Enterprise - Control System Integration - Part 5 : Business to Manufacturing Transactions.
- SP95 is the committee developing the ISA95 standard. ISA 95 Level Definitions
  1. Level 0 : Defines the actual physical processes.
  2. Level 1 : Defines the activities involved in sensing and manipulating the physical processes.
  3. Level 2 : Defines the activities of monitoring and controlling the physical processes.
  4. Level 3 : Defines the activities of the work flow to produce the desired end-products.
  5. Level 4 : Defines the business - related activities needed to manage a manufacturing organization.
- Fig. 1.10.5 shows OPC unified architecture.

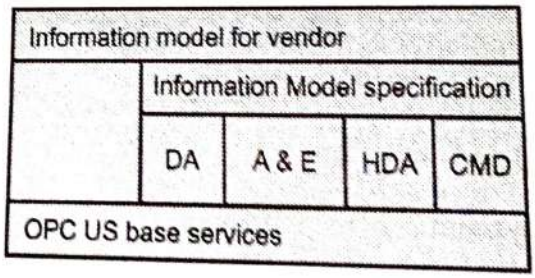


Fig. 1.10.5 : OPC unified architecture

1.11 Fill in the Blanks

Q.1 RESTful web service is a web API implemented using _____ and _____ principles.

Q.2 IEEE 802.15.4 is a collection of standards for _____wireless personal area networks.

Q.3 CoAP stands for _____.

Q.4 MQTT is a light weight messaging protocol based on the _____ model.

Q.5 XMPP is a _____ protocol and uses a client-server architecture.

Q.6 _____ APIs allows bi-directional, full duplex communication between clients and servers.

Q.7 A level - 5 IoT system has multiple end nodes and one _____ node.

Q.8 A level - 3 IoT system has a _____ node.

Q.9 AMQP stands for _____.

Q.10 CoAP is a _____ protocol that runs over the UDP for IoT.

Q.11 MQTT is a binary protocol, and it does not support the _____ or _____ of content.

Q.12 The XMPP protocol also uses _____ to bypass firewall barriers.

Q.13 A level _____ IoT system has multiple end nodes and one coordinator node.

Q.14 TCP is the _____ protocol whereas UDP is _____ protocol.

Q.15 IPv6 addresses are _____ bits in length.

Q.16 CoAP uses a _____ architecture where clients communicate with servers using connectionless datagrams.

1.12 Multiple Choice Questions

Q.1 IEEE _____ is a collection of wireless broadband standards, including extensive descriptions for the link layer.

a 802.3 b 802.11

c 802.15.4 d 802.16

Q.2 Representational State Transfer APIs follow the _____ communication model.

a request-response b publish-subscribe

c push-pull d exclusive pair

Q.3 XMPP stands for _____.

a EXtensible Markup Presence Protocol

b Extensible Messaging Presence Protocol

c Extensible Messaging Push-Pull

d Extensible Messaging Protocol Publish

Q.4 _____ is a lightweight broker-based publish / subscribe messaging protocol designed to be open, simple, lightweight and easy to implement.

a XMPP

b AMQP

c MQTT

d DDS

Q.5 _____ model is stateless communication model and each request-response pair is independent of others.

a Exclusive pair

b Request-response

c Publish-subscribe

d Push-pull

Q.6 _____ is an open application layer protocol for business messaging.

a MQTT

b DDS

c AMQP

d Websocket

Q.7 _____ is a web protocol that runs over the UDP for IoT.

a UPnP

b CoAP

c MQTT

d HTTP

Q.8 Which protocol is lightweight ?

a MQTT

b HTTP

c CoAP

d None of the these

Q.9 _____ is a peer-to-peer network architecture for connecting intelligent appliances, wireless devices, and PCs that are in close proximity.

a XMPP

b UART

c UPnP

d SPI

Q.10 Which of the following is not IoT servies as a platform ?

a SenseIoT
b Clayster
c Thinger.io
d XMPP

Q.11 _____ is a web protocol that runs over the UDP for IoT.

a UPnP
b CoAP
c MQTT
d HTTP

Q.12 MQTT stands for _____.

a Message Queue Transport Telemetry
b Machine Queue Telemetry Transport
c Message Queue Telemetry Transport
d Message Queue Transfer Transport

Q.13 List the 3I characteristics of the Internet of Things.

a Instrumented
b Interconnected
c Intelligently
d All of these

Q.14 Which of the following is NOT an IoT enabling technology ?

a Cloud computing
b Big data analytic
c Embedded system
d LAN

Q.15 CoAP uses _____ protocol.

a TCP
b UDP
c TCP and UDP
d IP

Q.16 HTTP protocol is used by _____ model.

a request/response
b publish/subscribe
c push/pull
d none

Q.17 Google AppEngine is an example of a _____.

a IaaS
b PaaS
c IaaS and PaaS
d None

Q.18 IoT stands for _____.

a Intranet of things
b Internet of Thoughts
c Internet of Things
d Information of Things

Q.19 The _____ in IoT usually refers to IoT devices which have unique identities and can perform remote sensing, actuating and monitoring capabilities.

a things
b information
c protocol
d data

Q.20 _____ model is stateless communication model and each request-response pair is independent of others.

a Exclusive pair
b Request-response
c Publish-subscribe
d Push-pull

Q.21 IoT communication models are _____.

a request/response model
b publish/subscribe model
c push/pull model
d all of the above

Q.22 A level _____ IoT system has multiple independent end nodes that perform sensing and/or actuation and send data to the cloud.

a 2
b 3
c 5
d 6

Q.23 A level-3 IoT system has a _____ node.

a multiple
b single
c end
d independent

Answer Keys for Fill in the Blanks

Q.1	HTTP, REST	Q.2	low rate	Q.3	Constrained Application Protocol
Q.4	publish subscribe	Q.5	decentralized	Q.6	webSocket
Q.7	coordinator	Q.8	single	Q.9	Advanced Message Queuing Protocol
Q.10	web	Q.11	encoding, decoding	Q.12	message brokers
Q.13	5	Q.14	connection oriented, connectionless	Q.15	128
Q.16	client - server				

Answer Keys for Multiple Choice Questions

Q.1	d	Q.2	a	Q.3	b	Q.4	c
Q.5	b	Q.6	c	Q.7	b	Q.8	a
Q.9	c	Q.10	d	Q.11	b	Q.12	c
Q.13	d	Q.14	d	Q.15	b	Q.16	a
Q.17	b	Q.18	c	Q.19	a	Q.20	b
Q.21	d	Q.22	d	Q.23	b		

❑❑❑

2 Embedded IoT Devices

Syllabus

Sensors and actuators for IoT applications, IoT components and implementation, Programming of NodeMCU and Raspberry PI, Implementation of IoT with Edge devices, Reading sensor data and transmit to cloud, Controlling devices through cloud using mobile application and web application, Types and configurations of gatways, Specifications of IoT gateways (Practical aspects of this chapter should be covered during lab sessions)

Contents

2.1 Sensors

2.2 Smart Objects

2.3 IoT System Building Blocks

2.4 Arduino

2.5 Raspberry Pi

2.6 Raspberry Pi Interface

2.7 Raspberry Pi with Python

2.8 Implementation of IoT with Edge Devices

2.9 Reading Sensor Data and Transmit to Cloud

2.10 Controlling Devices through Cloud using Mobile Application and Web Application

2.11 IoT Gateways

2.12 Fill in the Blanks

2.13 Multiple Choice Questions

2.1 Sensors

- Sensor converts a physical quantity into a corresponding voltage. Sensor is a device that when exposed to a physical phenomenon (temperature, displacement, force, etc.) produces a proportional output signal (electrical, mechanical, magnetic, etc.).
- The term transducer is often used synonymously with sensors. Sensor is a device that responds to a change in the physical phenomenon. On the other hand, a transducer is a device that converts one form of energy into another form of energy. Sensors are transducers when they sense one form of energy input and output in a different form of energy.
- Sensors can also be classified as passive or active. In passive sensors, the power required to produce the output is provided by the sensed physical phenomenon itself whereas the active sensors require external power source.
- In embedded system, sensor and actuators are used for controlling the system. Sensors are connected to input port. Actuators are connected to output port.
- Sensor captures the changes in the environmental variable. Middle system process the information. Actuators are changed according to the input variable. It displays the output.
- Example of control is air conditioner system. It controls the room temperature to a specified limit.
- Deflection : The signal produces some physical (deflection) effect closely related to the measured quantity and transuded to be observable.
- Null : The signal produced by the sensor is counteracted to minimize the deflection. That opposing effect necessary to maintain a zero deflection should be proportional to the signal of the measurand.
- Here, the output is usually an 'electrical quantity' and measurand is a 'physical quantity, property or condition which is to be measured'.
- A sensor is a device that responds to a physical stimulus, measures the physical stimulus quantity and converts it into a signal usually electrical, which can be read by an observer or by an instrument.
- A sensor can be very small and itself can be a trackable devices. The sensor itself, if not connected is not part of the IoT or WSN value chain.

Specifications of Sensor :

1. **Accuracy :** Error between the result of a measurement and the true value being measured.

2. **Resolution** : The smallest increment of measure that a device can make.
3. **Sensitivity** : The ratio between the change in the output signal to a small change in input physical signal. Slope of the input-output fit line.
4. **Repeatability/Precision** : The ability of the sensor to output the same value for the same input over a number of trials.
5. **Bandwidth** : The frequency range between the lower and upper cut-off frequencies within which the sensor transfer function is constant gain or linear.

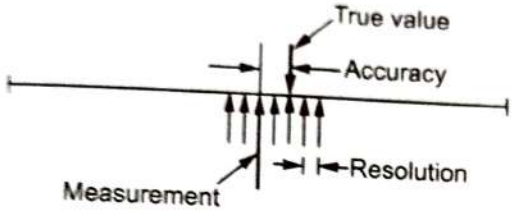


Fig. 2.1.1 Accuracy vs. Resolution

2.1.1 Types of Sensors

- Mechanical sensor : Any suitable mechanical / electrical switch may be adopted but because a certain amount of force is required to operate a mechanical switch it is common to use micro-switches.
- Pneumatic sensor : These proximity sensors operate by breaking or disturbing an air flow. The pneumatic proximity sensor is an example or a contact type sensor. These cannot be used where light components may be blown away.
- Optical sensor : In there simplest form, optimcal proximity sensors operate by breaking a light beam which falls onto a light sensitive device such as a photocell. These are examples of non contact sensors. Care must be exercised with the lighting environment of these sensors for example optical sensors can be blinded by flashes from are welding processes, airborne dust and smoke clouds may impede light transmission etc.
- Electrical sensor : Electrical proximity sensors may be contact or non-contact. Simple contact sensors operate by making the sensor and the component complete an electrical circuit. Non-contact electrical proximity sensors rely on the electrical principles of either induction for detecting metals or capacitance for detecting non metals as well.
- Range sensing : Range sensing concerns detecting how near or far a component is from the sensing position, although they can also be used as proximity sensors. Distance or range sensors use non-contact analog techniques. Short range sensing,

between a few millimetres and a few hundred millimetres is carried out using transmitted energy waves of various types e.g. radio waves, sound waves and lasers.

2.1.2 Sensor Data Communication Protocols

1. Direct transmission protocols :

- In direct communication protocol, each sensor sends its data directly to the base station. If the base station is far away from the nodes, direct communication will require a large amount of transmit power from each node.
- This will quickly drain the battery of the nodes and reduce the system lifetime. However, the only receptions in this protocol occur at the base station, so if either the base station is close to the nodes, or the energy required receiving data is large, this may be an acceptable method of communication.

2. Minimum transfer energy protocols :

- In these protocols, nodes act as routers for other nodes' data in addition to sensing the environment. These protocols differ in the way the routes are chosen.
- Some of these protocols only consider the energy of the transmitter and neglect the energy dissipation of the receivers in determining the routes.
- Depending on the real time costs of the transmit amplifier and the radio electronics, the total energy expended in the system might actually be greater using MTE routing than direct transmission to the base station.
- It is clear that in MTE routing, the nodes closest to the base station will be used to route a large number of data messages to the base station.
- Thus, these nodes will die out quickly, causing the energy required to get the remaining data to the base station to increase and more nodes to die.

2.1.3 Actuators

- A device or mechanism capable of performing a physical action. Actuators interact with the world. Sensors capture information from the world.
- The interface between the microcontroller and the sensors or the actuators is either analog or digital.
- An actuator requires a control signal and a source of energy. An actuator is the mechanism by which a control system acts upon an environment. The control system can be simple, software-based, a human, or any other input.
- When the actuation is a motion, motor have to be used for rotational or linear motion.

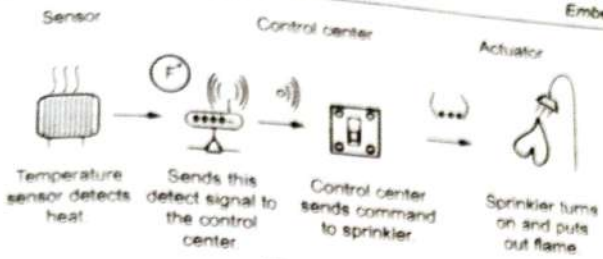


Fig. 2.1.2

- The selection of the proper actuator is more complicated than selection of the sensors, primarily due to their effect on the dynamic behaviour of the overall system. Furthermore, the selection of the actuator dominates the power needs and the coupling mechanisms of the entire system.
- In typical IoT systems, a sensor may collect information and route to a control centre where a decision is made and a corresponding command is sent back to an actuator in response to that sensed input.

2.2 Smart Objects

- Smart objects are an autonomous physical and/or digital object that have sensing, processing, and networking capabilities, and carry application logic.
- They make sense of their local environment and interact with human users.
- They sense, log, and interpret what's occurring within themselves and the world, act on their own, intercommunicate with each other, and exchange information with people.
- Smart objects are small computers with a sensor or actuator and a communication device, embedded in objects such as thermometers, car engines, light switches, and industry machinery.
- Smart objects enable a wide range of applications in areas such as home automation, building automation, factory monitoring, smart cities, structural health management systems, smart grid and energy management, and transportation.
- Until recently, smart objects were realized with limited communication capabilities, such as RFID tags, but the new generation of devices has bidirectional wireless communication and sensors that provide real-time data such as temperature, pressure, vibrations, and energy measurement.
- Smart objects can be battery-operated, but not always, and typically have three components: a CPU (8-, 16- or 32-bit micro-controller), memory and a low-power wireless communication device.
- The size is small and the price is low.

- Advantages in designing IoT systems based on smart objects are as follows :
  1. Energy saving is one of them. Smart objects are usually powered by battery.
  2. The second advantage is automation. IoT smart objects are autonomous and self-governed.
  3. They operate independently and can collaborate with other objects globally.
- Challenges of Using Smart Objects :
  1. Smart objects are often constrained devices and are usually powered by battery.
  2. Frequently they are working in real-time mode. These are the main causes of the challenges.
  3. Other challenge is connectivity. Currently a large number of networking technologies are being employed in connecting physical devices together and to the Internet.
  4. Security and privacy is of big concern for smart object based IoT systems.
  5. Diversity of communication technologies : Depending on the application and the environment in which the system is deployed, smart objects can use a wide range of communication technologies

2.2.1 Communication Patterns used for Smart Objects

- Smart object communication patterns can be divided into three categories : one-to-one, one-to-many and many-to-one.
- Smart objects have specific communication patterns based on their application
- Smart objects often communicate over unreliable communication channels. The radio transmission of a smart object with a radio transceiver may be disturbed by other radio senders in the vicinity.

1. **One-to-one communication** pattern occurs when one smart object communicates with another smart object. The communication may involve other smart objects, however, as the communication may be routed through a network of smart objects. Fig 2.2.1 shows one-to-one communication in a smart object network.

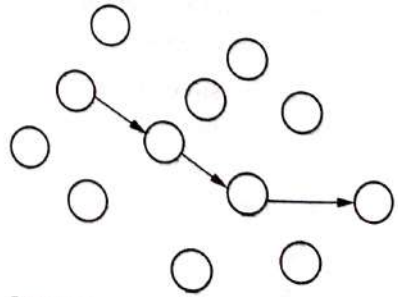


Fig. 2.2.1 One-to-one communication in a smart object network

2. **One-to-many communication** pattern is used for sending messages from one node to several other nodes and possibly all other nodes in the network. This can be used, for example, for sending a command to a set of nodes in the network. Fig 2.2.2 shows one-to-many communication in a smart object network.

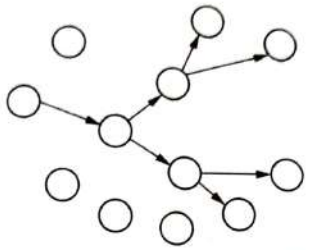


Fig. 2.2.2 One-to-many communication in a smart object network

3. **Many-to-one communication** occurs frequently in smart object networks that collect data from the nodes. In many-to-one communication, several nodes send data toward a single node. This node is often called a sink node. Fig 2.2.3 shows Many-to-one communication in a smart object network.

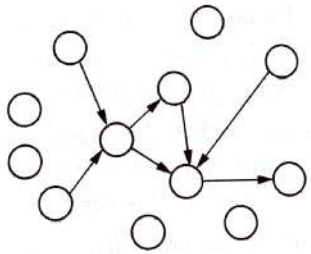


Fig. 2.2.3 Many-to-one communication in a smart object network

- Many-to-one communication can be used to collect sensor data, such as temperature data, from the nodes in the network, but it can also be used for network health status information.
- Nodes send periodic status reports to a sink node. The sink node then reports the overall performance of the network to an outside observer.

2.2.2 Connecting Smart Objects

- Range is one the communication criteria for smart objects. Range may be short, medium and long.

1. **Short range** : Classical example is serial cable. It supporting upto 10meters as maximum distance between two devices. Example of wireless technologies are Bluetooth and visible light communication.
2. **Medium range** : Its range is 10 to 100 meters. It is main categories of IoT access technologies. Maximum distance between two devices is less than 1 mile. Example is 802.11 Wi-Fi, IEEE 802.15.4 and 802.15.4g WPAN. For wired technology, 802.3 Ethernet is an example.
3. **Long range** : Here the distance is greater than 1 miles between two devices. Example of wireless technology are 2G, 3G, 4G etc.

Frequency bands :

- Frequency band is a specific range of frequencies in the radio frequency (RF) spectrum, which is divided among ranges from very low frequencies (vlf) to extremely high frequencies (ehf). Each band has a defined upper and lower frequency limit.
- Very low frequencies (vlf) range from 3 to 30 kilohertz (kHz). Time signals and standard frequencies are among the users of this band.
- Low frequencies (lf) range from 30 to 300 kHz. Fixed, maritime mobile and navigational systems and radio broadcasting are among the users of this band.
- Medium frequencies (mf) range from 300 to 3000 kHz. Land, maritime mobile and radio broadcasting are among the users of this band.
- High frequencies (hf) - also called shortwaves - range from 3 to 30 megahertz (MHz). Fixed, mobile, aeronautical and marine mobile, amateur radio, and radio broadcasting are among the users of this band.
- Cellular networks operate on different frequency bands including the 450 MHz band, 700 MHz band, 800 MHz band, 900 MHz band, 1800 MHz band, 2100 MHz band, and 2600 MHz band.
- The industrial, scientific, and medical radio band (ISM band) refers to a group of radio bands or parts of the radio spectrum that are internationally reserved for the use of radio frequency (RF) energy intended for scientific, medical and industrial requirements rather than for communications.
- In the U.S., the 902 - 928 MHz, 2.4 GHz and 5.7 - 5.8 GHz bands were initially used for machines that emitted radio frequencies, such as RF welders, industrial heaters and microwave ovens, but not for radio communications.

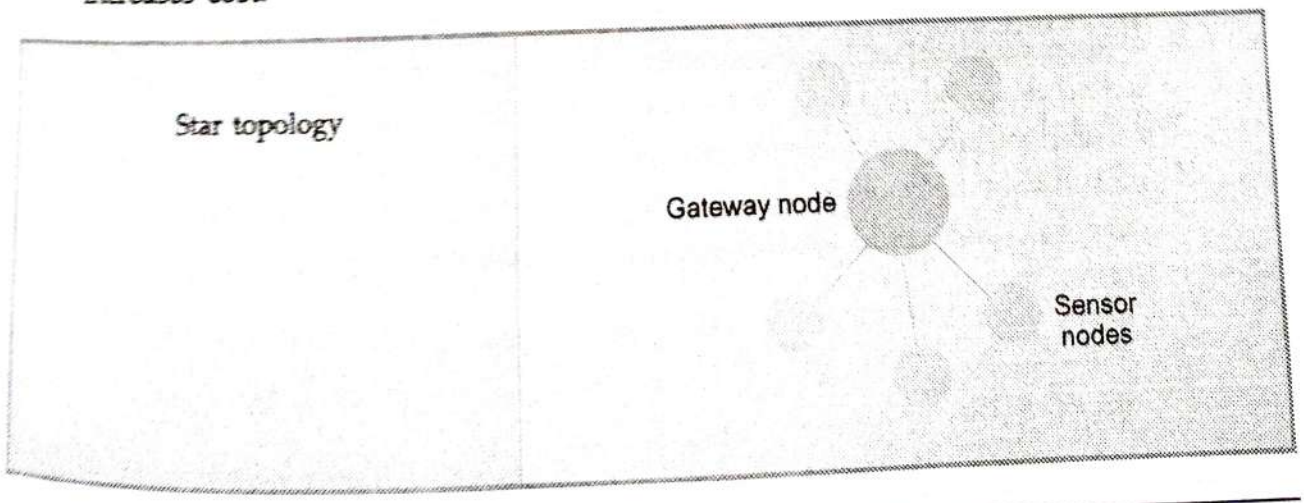
Power Consumption :

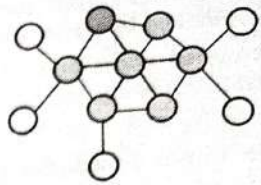
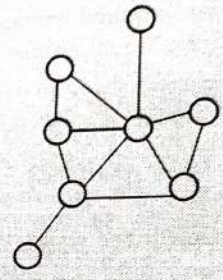
- IoT uses two types of devices : Powered node and battery powered node.

- Powered node has a direct connection to a power source and communications are usually not limited by power consumptions criteria.
- Battery powered nodes bring much more flexibility. Batteries can be changed or device can be replaced.
- Wireless sensor nodes are battery-powered devices, since it is generally difficult or impossible to run a mains supply to their deployment site. Power to the wireless sensor nodes is usually provided through primary batteries
- Primary batteries are typical power sources for sensor nodes. By supplying their energy at the required voltage levels, they eliminate the need for intermediate power conditioning electronics.
- When a primary battery is the single power source of a sensor node, the amount of initially stored energy determines the node's lifetime. The main metric of primary batteries is their energy density.

Topology :

- For connecting IoT devices, three main topologies are used. They are star, mesh and pee-to-peer topology.
- In the star network topology, a central node has a direct connection to all other nodes.
- In the mesh topology every node can be both an end device or a router, meaning that each node has several links to the coordinator. This means that if one of the routers goes offline, most of the network is still intact by rerouting through the remaining routers.
- Peer-to-peer topology rely on multiple full function devices.
- A network is said to be fully meshed if all nodes are directly connected to all other nodes, and partially meshed if only some nodes have multiple connections to others. Meshing to create multiple paths increases resiliency under failure, but increases cost.



Mesh topology	
Peer to Peer topology	

- Peer-to-peer topology rely on multiple full function devices.
- A network is said to be fully meshed if all nodes are directly connected to all other nodes, and partially meshed if only some nodes have multiple connections to others. Meshing to create multiple paths increases resiliency under failure, but increases cost.

Constrained devices

- Following are the classes of constrained nodes as defined by RFC 7228

1. Class 0 devices : They are so severely constrained in memory and processing capabilities that most likely they will not have the resources required to communicate directly with the Internet in a secure manner. Class 0 devices will participate in Internet communications with the help of larger devices acting as proxies, gateways, or servers. Class 0 devices generally cannot be secured or managed comprehensively in the traditional sense. They will most likely be preconfigured with a very small data set. For management purposes, they could answer keepalive signals and send on/ off or basic health indications.

2. Class 1 devices are quite constrained in code space and processing capabilities, such that they cannot easily talk to other Internet nodes employing a full protocol stack such as using HTTP, Transport Layer Security (TLS), and related security protocols and XML-based data representations. However, they are capable enough to use a protocol stack specifically designed for constrained nodes and participate in meaningful conversations without the help of a gateway node. In particular, they can provide support for the security functions required on a large network.

3. Class 2 devices are less constrained and fundamentally capable of supporting most of the same protocol stacks as used on notebooks or servers. However, even these devices can benefit from lightweight and energy-efficient protocols and from consuming less bandwidth. Furthermore, using fewer resources for networking leaves more resources available to applications. Thus, using the protocol stacks defined for more constrained devices on Class 2 devices might reduce development costs and increase the interoperability.

2.3 IoT System Building Blocks

- The hardware utilized in IoT systems includes devices for a remote dashboard, devices for control, servers, a routing or bridge device, and sensors. These devices manage key tasks and functions such as system activation, action specifications, security, communication, and detection to support-specific goals and actions.
- Major components of IoT devices are as follows :

1. **Control units :** A small computer on a single integrated circuit containing processor core, memory and a programmable I/O peripheral. It is responsible for the main operation.

2. **Sensor :** Devices that can measure a physical quantity and convert it into a signal, which can be read and interpreted by the microcontroller unit. These devices consist of energy modules, power management modules, RF modules, and sensing modules. Most sensors fall into 2 categories: Digital or analog. An analog data is converted to digital value that can be transmitted to the Internet.

   a. Temperature sensors : Accelerometers

   b. Image sensors : Gyroscopes

   c. Light sensors : Acoustic sensors

   d. Micro flow sensors : Humidity sensors

   e. Gas RFID sensors : Pressure sensors

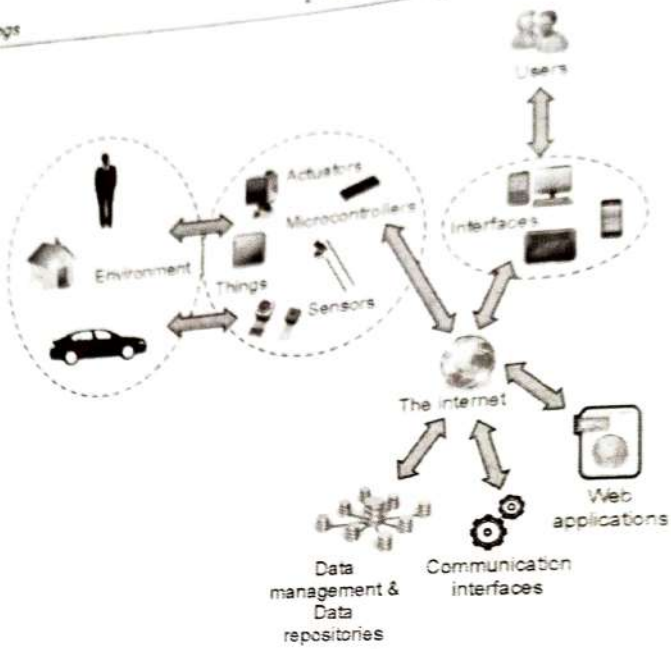


Fig. 2.3.1 Working of IoT

3. **Communication modules :** These are the part of devices and responsible for communication with rest of IoT platform. They provide connectivity according to wireless or wired communication protocol they are designed. The communication between IoT devices and the Internet is performed in two ways :

   A) There is an Internet-enable intermediate node acting as a gateway;

   B) The IoT Device has direct communication with the Internet.

- The communication between the main control unit and the communication module uses serial protocol in most cases.

4. **Power sources :** In small devices the current is usually produced by sources like batteries, thermocouples and solar cells. Mobile devices are mostly powered by lightweight batteries that can be recharged for longer life duration.

- **Communication Technology and Protocol :** IoT primarily exploits standard protocols and networking technologies. However, the major enabling technologies and protocols of IoT are RFID, NFC, low-energy Bluetooth, low-energy wireless, low-energy radio protocols, LTE-A, and WiFi-Direct. These technologies support the specific networking functionality needed in an IoT system in contrast to a standard uniform network of common systems.

Working :

1. **Collect and transmit data :** The device can sense the environment and collect information related to it and transmit it to a different device or to the Internet.
2. **Actuate device based on triggers :** It can be programmed to actuate other devices based on conditions set by user.
3. **Receive information :** Device can also receive information from the network.
4. **Communication assistance :** It provides communication between two devices of same network or different network.

- Fig. 2.3.1 shows working of IoT.
- Sensors for various applications are used in different IoT devices as per different applications such as temperature, power, humidity, proximity, force etc.
- Gateway takes care of various wireless standard interfaces and hence one gateway can handle multiple techologies and multiple sensors. The typical wireless technologies used widely are 6LoWPAN, Zigbee, Zwave, RFID, NFC etc. Gateway interfaces with cloud using backbone wireless or wired technologies such as WiFi, Mobile , DSL or Fibre.

2.4 Arduino

- Arduino is an open-source electronics platform based on easy-to-use hardware and software.
- Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online.
- The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.

Features :

- Support fast computations, ARM based MCU
- AVR micro-controller clock is ATSAMSX8I
- Operating input voltages is 3.3 Volt
- It uses EEPROM, SRAM and Flash memory
- It also support USB and UART

- Fig. 2.4.1 shows Arduino board.

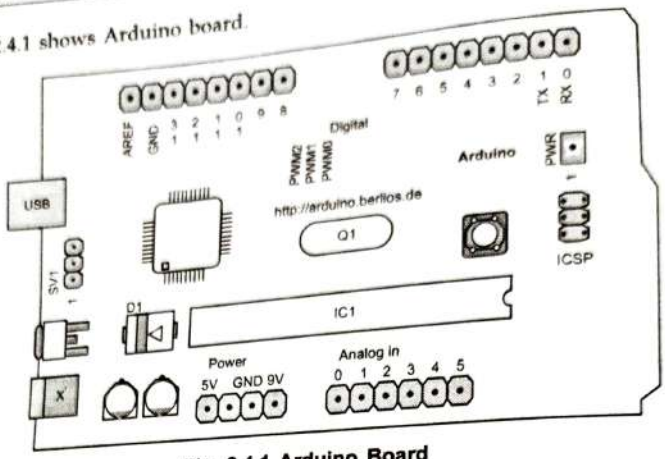


Fig. 2.4.1 Arduino Board

Starting clockwise from the top center,

- Analog Reference pin (1st pin)
- Digital Ground
- Digital Pins 2-13 (green)
- Digital Pins 0-1/Serial In/Out - TX/RX : These pins cannot be used for digital I/O (digital Read and digital Write) if you are also using serial communication
- Reset Button - S1
- In-circuit Serial Programmer
- Analog In Pins 0-5
- Power and Ground Pins
- External Power Supply In (9-12VDC) - X1
- Toggles External Power and USB Power (place jumper on two pins closest to desired supply) - SV1 (purple)
- USB

Digital Pins

- In addition to the specific functions listed below, the digital pins on an Arduino board can be used for general purpose input and output via the pinMode( ), digitalRead( ), and digitalWrite( ) commands.
- Each pin has an internal pull-up resistor which can be turned on and off using digitalWrite( ), when the pin is configured as an input. The maximum current per pin is 40 mA.

- **Serial** : 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. On the Arduino Diecimila, these pins are connected to the corresponding pins of the FTDI USB-to-TTL Serial chip. On the Arduino BT, they are connected to the corresponding pins of the WT11 Bluetooth module. On the Arduino Mini and LilyPad Arduino, they are intended for use with an external TTL serial module (e.g. the Mini-USB Adapter).
- **External Interrupts** : 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM** : 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function. On boards with an ATmega8, PWM output is available only on pins 9, 10, and 11.
- **BT Reset** : 7. (Arduino BT-only) Connected to the reset line of the bluetooth module.
- **SPI** : 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- **LED** : 13. On the Diecimila and LilyPad, there is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

Analog Pins

- In addition to the specific functions listed below, the analog input pins support 10-bit analog-to-digital conversion (ADC) using the analogRead() function.
- Most of the analog inputs can also be used as digital pins: analog input 0 as digital pin 14 through analog input 5 as digital pin 19. Analog inputs 6 and 7 (present on the Mini and BT) cannot be used as digital pins.
- **I2C** : 4 (SDA) and 5 (SCL). Support I2C (TWI) communication.

Power Pins

- VIN (sometimes labelled "9 V"). The input voltage to the Arduino board when it's using an external power source.
- You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin. Note that different boards accept different input voltages ranges, please see the documentation for your board. Also note that the LilyPad has no VIN pin and accepts only a regulated input.

- **5 V** : The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5 V supply.
- **3V3** : (Diecimila-only) A 3.3 volt supply generated by the on-board FTDI chip.
- **GND** : Ground pins.

Other Pins

- **AREF** : Reference voltage for the analog inputs. Not currently supported by the Arduino software.
- **Reset** : Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.
- It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.
- The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7 V, however, the 5 V pin may supply less than five volts and the board may be unstable. If using more than 12 V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

Arduino Uno R3 Programming

- The programming of an Arduino Uno R3 can be done using IDE software. The microcontroller on the board will come with pre-burned by a boot loader that permits to upload fresh code without using an exterior hardware programmer.
- The communication of this can be done using a protocol like STK500.
- We can also upload the program in the microcontroller by avoiding the boot loader using the header like the In-Circuit Serial Programming.

2.5 Raspberry Pi

- A Raspberry Pi is a credit card-sized computer originally designed for education, inspired by the 1981 BBC Micro.
- Creator Eben Upton's goal was to create a low-cost device that would improve programming skills and hardware understanding at the pre-university level.
- The Raspberry Pi is slower than a modern laptop or desktop but is still a complete Linux computer and can provide all the expected abilities that implies, at a low-power consumption level.

Versions	Remarks
Raspberry Pi 1	• The original Raspberry Pi had 256 Mb of RAM, which increased to 512 MB in a later revision. • It has a 26-way GPIO connector
Pi Zero	• The Pi Zero includes the GPIO connector, but the header pins are not soldered
Raspberry Pi 2	• The Raspberry Pi 2 swapped the single-core processor for a much faster quad-core processor and increased the memory to 1 GB RAM
Raspberry Pi 3	• The Raspberry Pi 3 changes the processor to an even more powerful 64-bit processor. • It also adds Wi-Fi and bluetooth which previously needed to be added as a USB device. • The Raspberry Pi 3 Model B was launched in February 2016.

- To get the Raspberry Pi working an SD card needs to be prepared with the Linux operating system installed.
- Raspberry Pi users have made many creative and impressive projects using this device. It can also be programmed to assist in 'housekeeping' your network by functioning as NAS, LDAP server, web server, media server, DNS server etc.
- The Raspberry Pi Foundation recommends Python. Any language which will compile for ARMv6 can be used. Installed by default on the Raspberry Pi : C, C++, Java, Scratch and Ruby.

2.5.1 About the Board

- Fig 2.5.1 shows the Raspberry Pi board. The Raspberry Pi does not have a separate CPU, RAM or GPU. Instead they are all squeezed into one component called a system on Chip or SoC unit.
- Raspberry Pi is open hardware with the exception of its primary chip, the Broadcomm SoC which runs the main components of the board - CPU, graphics, memory, USB controller etc.

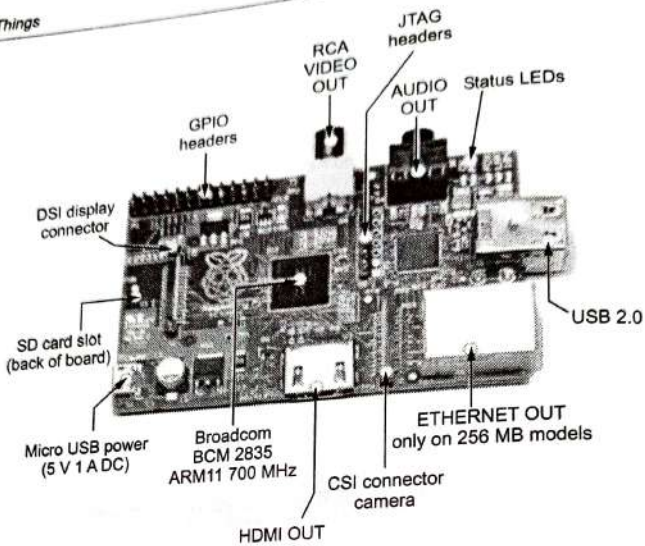


Fig 2.5.1 (a) Raspberry Pi circuit board

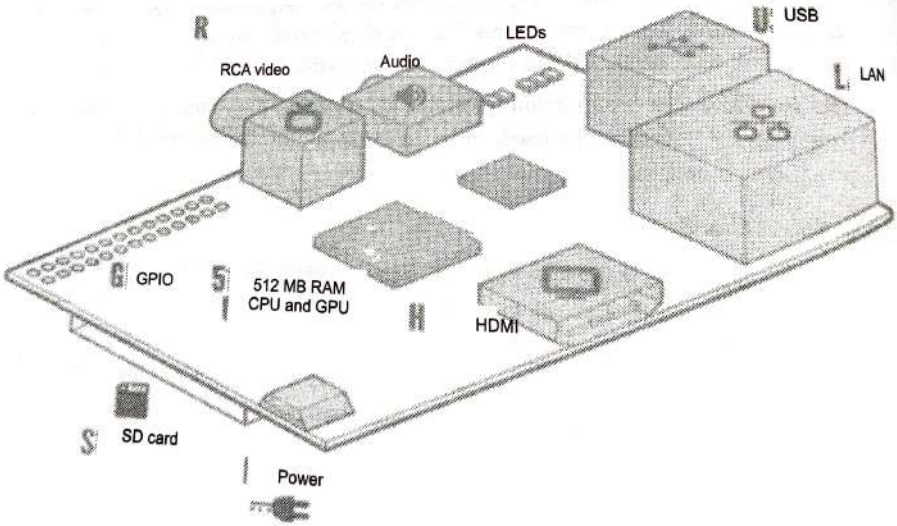


Fig 2.5.1 (b) Block diagram

- All of these Raspberry Pi Models share the following features :
  1. Operating systems : Raspbian RaspBMC, Arch Linux, Rise OS, OpenELEC Pidora
  2. Video output : HDMI Composite RCA
  3. Supported resolutions : 640x350 to 1920x1200, including 1080p, PAL and NTSC standards
  4. Power source : Micro USB

Components	Description
Processor	• Raspberry Pi uses an ARM processor which is also installed in a wide variety of mobile phones. • This CPU is single core, however it does have a co-processor to perform floating point calculations
Memory	• Model B Raspberry Pi has 512 MB SDRAM (Synchronous Dynamic RAM). • It store programs that are currently being run in the CPU
USB ports	• Board has two USB ports. USB port can provide a current upto 100 mA • Using powered hub, it is possible to connect more devices.
HDMI Output	• High Definition Multimedia Interface (HDMI) supports high-quality digital video and audio through a single cable. • It is also possible to connect a computer monitor with a DVI connection to HDMI using a converter.
Composite Video Output	• It support composite video output with RCA jack and also support PAL and NTSC. • The TVDAC pin can be used to output composite video.
Audio Output	• Audio output jack is 3.5 mm. • This jack is used for providing audio output to old television along with the RCA jack for video
GPIO Pins	• Both models have a total of 26 GPIO pins, organized into one pin header, named the P1 header • The newer Raspberry Pi (model B revision 2) adds 8 more GPIO pins in a new pin header called P5

	• Not all the GPIO pins are programmable. Some of them are 5.0 VDC or 3.3 VDC positive power pins, some of them are negative ground pins and a few of them are marked DNC (do not connect). • The P1 header has 17 programmable pins and the P5 header adds 4 more. • Fig 2.5.2 shows GPIO pin header. • Reading from various environmental sensors. Writing output to dc motors, LEDs for status.
Power Input	• Micro-USB connector is used for power input
Status LED	• It has five status LED.
CSI	• Camera Serial Interface (CSI) can be used to connect a camera module to Raspberry Pi
SD Card Slot	• This card is used for loading operating system

- The Raspberry Pi comes with a set of 26 exposed vertical pins on the board. These pins are a General Purpose Input /Output interface that is purposely not linked to any specific native function on the Raspberry Pi board.

Fig. 2.5.2 GPIO pin header

- Instead, the GPIO pins are there explicitly for the end user to have low-level hardware access directly to the board for the purposes of attaching other hardware boards, peripherals, LCD display screens and other hardware devices to the Pi.

The Status LEDs

Status LED	Color	Functions
ACT	Green	Lights when the SD card is accessed (marked OK on earlier boards)
PWR	Red	Hooked up to 3.3 V power
FDX	Green	On if network adapter is full duplex
LNK	Green	Network activity light
100	Yellow	On if the network connection is 100 Mbps

- The Raspberry Pi draws its power from a microUSB port and requires a microUSB-to-AC adapter. Because the Pi is a micro computer and not simply a cell phone getting a battery topped off, you need to use a high quality charger with stable power delivery that provides a consistent 5 V with at least 700 mA minimum output for older model units and 2.5 A for the Pi 3.

2.5.2 Linux on Raspberry Pi

- There are several unix like operating systems for the RPI and there is an operating system called RISC OS that has its origin at the developers of the first ARM chips.
- The Raspberry Pi Foundation recommends the use of the following Linux Distributions :

1. Debian 7
2. Raspbian
3. Arch Linux ARM
4. QtonPi

- Raspbian is a free operating system based on Debian optimized for the Raspberry Pi (RPI) hardware.
- The default command prompt on the Pi consists of four components shown in Fig. 2.5.3.
- Raspbian is the desired operating system for the Raspberry Pi. In order to download and install the operating system onto our Raspberry Pi; you will need Raspbian, Win32DiskImager and USB memory card reader.

1. Download both Raspbian and Win32DiskImager and save somewhere easily accessible
2. Plug the USB memory card reader into your computer
3. Open Win32DiskImager
4. Find the location of the image file and the memory card
5. Click "Write"

Fig. 2.5.3 Command prompt

Logging In

- Now it is time to turn on our Raspberry Pi. When the memory card, HDMI lead, Ethernet cable, mouse and keyboard are plugged in, plug in the power lead.
- As soon as you do this. You screen should be black and filled with white text. This will be visible every time you turn on your raspberry pi.
- Wait until your screen reads "raspberrypi login :"

  Username = pi [ENTER]

  Password = raspberry [ENTER]

```
Debian GNU/Linux wheezy/sid raspberrypi tty1

raspberrypi login: pi
Password:
Last login: Tue Aug 21 21:24:50 EDT 2012 on tty1
Linux raspberrypi 3.1.9+ #168 PREEMPT Sat Jul 14 18:56:31 BST 2012 armv6l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

Type 'startx' to launch a graphical session

pi@raspberrypi ~ $
```

Starting the Raspbian GUI

- GUI stands for Graphical User Interface and is a type of operating system. It is the most common type of user interface as it is a very 'friendly' way for people to interact with the computer. It makes use of pictures, graphics, icons and pointers, hence the name 'Graphical' User Interface. Fig. 2.5.4 shows Rasbian Linux desktop

1. Type the line : "startx"

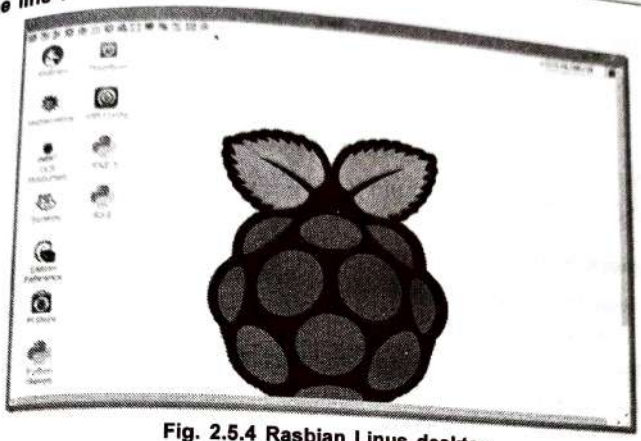


Fig. 2.5.4 Rasbian Linus desktop

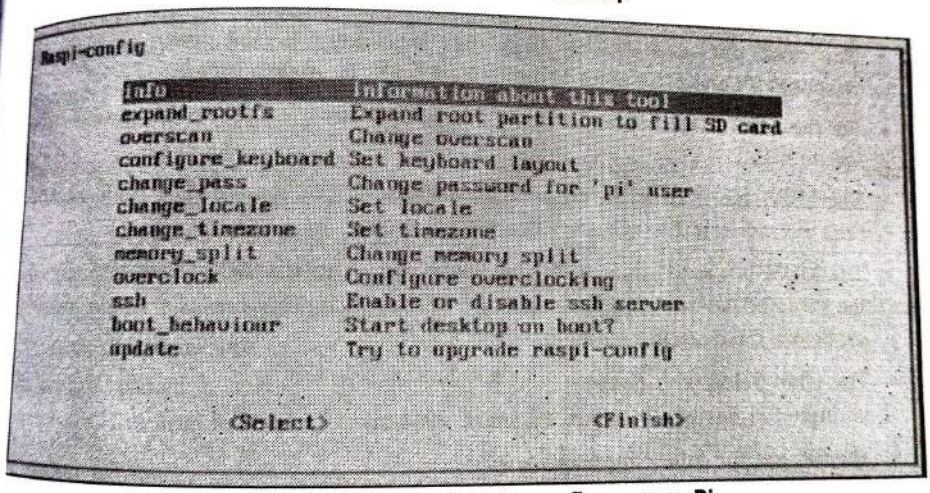


Fig. 2.5.5 First boot : time to configure your Pi

2.5.3 Difference between Raspberry Pi is and Desktop Computers

- In Raspberry Pi, operating system is installed on SD card whereas in desktop computer, operating system is installed in hard disk.
- Raspberry Pi does not have their own CPU and RAM.
- Processing power of Raspberry Pi is less as compared to desktop computers.
- Raspberry Pi uses less power than desktop computers.

2.6 Raspberry Pi Interface

- Three types of interface is supported by Raspberry Pi.

1. Serial

- It uses serial peripherals for serial communication.
- Transmit (Tx) and Receive (Rx) pin is used for serial communication.

2. Serial Peripheral Interface (SPI)

- SPI is a communication protocol used to transfer data between micro-computers like the Raspberry Pi and peripheral devices. These peripheral devices may be either sensors or actuators.
- SPI uses 4 separate connections to communicate with the target device. These connections are the serial clock (CLK), Master Input Slave Output (MISO), Master Output Slave Input (MOSI) and Chip Select (CS).
- The clock pin sense pulses at a regular frequency, the speed at which the Raspberry Pi and SPI device agree to transfer data to each other.
- For the ADC, clock pulses are sampled on their rising edge, on the transition from low to high.
- The MISO pin is a data pin used for the master to receive data from the ADC. Data is read from the bus after every clock pulse.
- The MOSI pin sends data from the Raspberry Pi to the ADC. The ADC will take the value of the bus on the rising edge of the clock. This means the value must be set before the clock is pulsed.
- The Chip Select line chooses which particular SPI device is in use. If there are multiple SPI devices, they can all share the same CLK, MOSI, and MISO.
- The SPI has the following features :

1. 16-bit shift register
2. 16-bit Receive buffer register (SPIBUF) and 16-bit Receive buffer emulation alias register (SPIEMU)
3. 16-bit Transmit data register (SPIDAT0) and 16-bit Transmit data and format selection register (SPIDAT1)
4. 8-bit baud clock generator
5. Serial clock (SPICLK) I/O pin
6. Slave in, master out (SPISIMO) I/O pin
7. Slave out, master in (SPISOMI) I/O pin

8. Multiple slave chip select (SPISCS[n]) I/O pins (4 pin mode only)
9. Programmable SPI clock frequency range
10. Programmable character length (2 to 16 bits)
11. Programmable clock phase (delay or no delay)
12. Programmable clock polarity (high or low)
13. Interrupt capability
14. DMA support (read/write synchronization events)
15. Up to 66 MHz operation

Master-slave configuration of SPI :

- Fig. 2.6.1 shows SPI system. SPI bus is composed by four signals, namely the Master Out Slave In (MOSI), Master In Slave Out (MISO), serial clock (SCK) and active low slave select (\SS).
- MOSI : This pin is used to transmit data out of the SPI module when it is configured as a Master and receive data when it is configured as Slave.
- MISO : This pin is used to transmit data out of the SPI module when it is configured as a Slave and receive data when it is configured as Master.
- /SS : This pin is used to output the select signal from the SPI module to another peripheral with which a data transfer is to take place when its configured as a Master and its used as an input to receive the slave select signal when the SPI is configured as Slave.
- SCLK : This pin is used to output the clock with respect to which the SPI transfers data or receive clock in case of Slave.

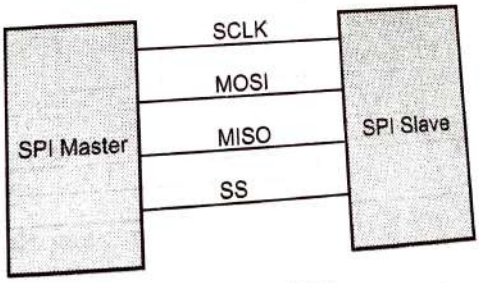


Fig. 2.6.1 SPI

- SCK master device will generate a pulse and the data will be synchronized in both master and slave devices. There are four different clock types to define SIP protocol, depending on what the SCK polarity and phase may be. It must ensure these signals between the master and slave devices compatible with each other.

- SPI is a synchronous protocol. The clock signal is provided by the master to provide synchronization. The clock signal controls when data can change and when it is valid for reading.
- SPI creates a data loop between two devices. Data leaving the master exits on the SDO (serial data output) line. Data entering the master enters on the serial data input, SDI line.
- A clock (SCK), is generated by the master device. It controls when and how quickly data is exchanged between the two devices.
- SS allows a master device to control when a particular slave is being addressed. This allows the possibility of having more than one slave and simplifies the communications. When the SS signal goes low at a slave device, only that slave is accessed by SPI.
- For SPI, there are Serial Clocks (SCLK), Chip Select lines (CS), Serial Data In (SDI) and Serial Data Out(SDO). There is only one master, there number of slaves depends on the number of chip select lines of the master.
- Synchronous operation, latch on rising or falling edge of clock, SDI on rising edge SDO on falling edge. It operates in 1 to 2 MHz range.
- Master sends out clocks and chip selects. Activates the slaves it wants to communicate with.
- Fig. 2.6.2 master with multiple slave interface.

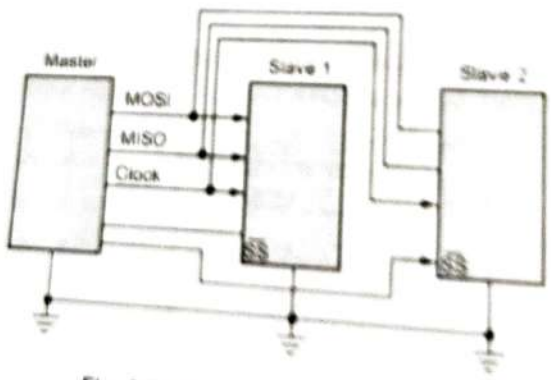


Fig. 2.6.2 Multiple slave interface

- SPI data transmit and data receive register are the main elements of the SPI. When the communication takes place the data on the transmit register are transferred into the shift register.

- The shift register in the master of width (8,16,32) and the shift register in the slave are linked by MOSI and MISO pins to form a distributed 16,32,64 bit register respectively.
- When the data transfer operation needs to be performed these 16,32,64- bit registers are serially shifted eight, sixteen, thirty-two bit positions by the serial clock generated by the master so that the data can be exchanged between the master and the selected slave.
- Data on the master SPI data transmit register becomes the input data for the slave read from the MOSI and the data read from the master SPI data receive register was the data send from the slave from MISO.
- Data on the shift registers are transferred into data receive register when the transfer completes and this data may be read from the data receive register any time before next transfer has completed.

3. I2C

- I2C is a communication protocol that the Raspberry Pi can use to speak to other embedded devices (temperature sensors, displays, accelerometers, etc).
- I2C is a useful bus that allows data exchange between microcontrollers and peripherals with a minimum of wiring.
- I2C is a two wire bus, the connections are called SDA (Serial Data) and SCL (Serial Clock). Each I2C bus has one or more masters (Raspberry Pi) and one or more slave devices, like the I/O Expander.
- As the same data and clock lines are shared between multiple slaves, we need some way to choose which device to communicate with.

2.7 Raspberry Pi with Python

- General Purpose Input/Output (GPIO) is a generic pin on a chip whose behavior can be controlled by the user at run time. The GPIO connector has a number of different types of connection :

1. True GPIO pins that you can use to turn LEDs on and off etc.
2. I2C interface pins that allow you to connect hardware modules with just two control pins.
3. SPI interface with SPI devices, a similar concept to I2C but uses a different standard.
4. Serial Rx and Tx pins for communication with serial peripherals.

2.7.1 Controlling LED with Raspberry Pi

- Fig 2.7.1 shows diagram of connecting LED to Raspberry Pi. The LED will initially be off because the GPIO pins are initialized as inputs at power-on.
- Install Python 2 library Rpi.GPIO. A library that will let us control the GPIO pins. Install commands :

  sudo apt - get update

  sudo apt - get install python - dev

  sudo apt - get install python - rpi.gpio

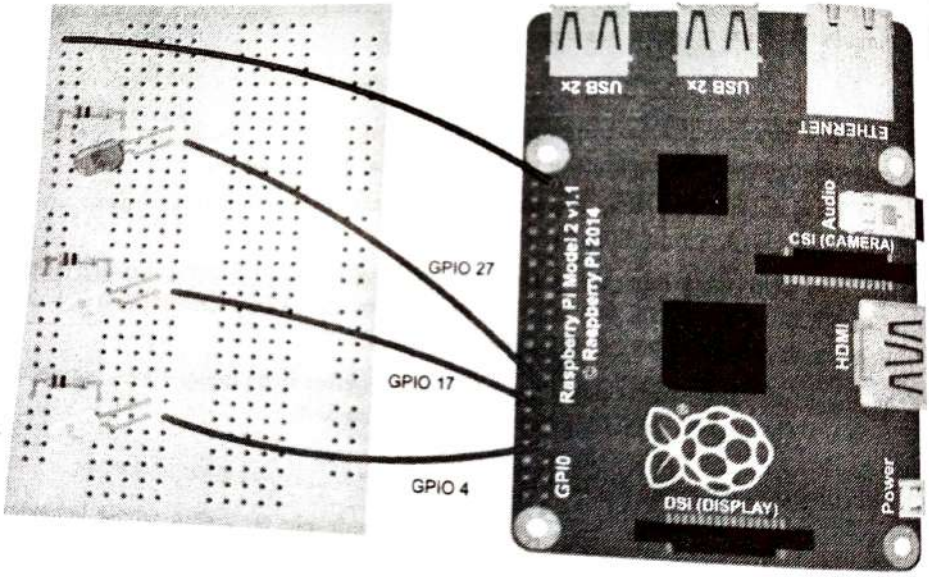


Fig. 2.7.1 Diagram of connecting LED to Raspberry Pi

- Simple LED circuit is shows below :

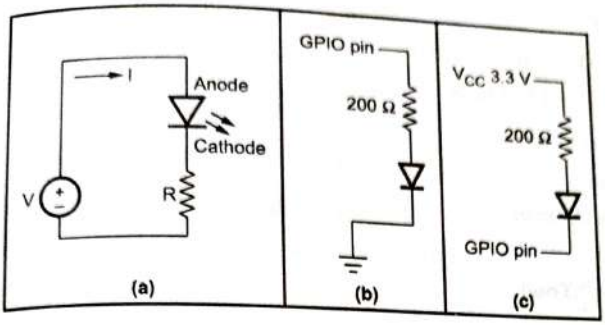


Fig. 2.7.2

- Current flows from the anode (+) to cathode (–). Anode is longer pin and cathode is shorter pin.
- Open up IDLE, the Python programming software and create a New file. Save it as led.py and input the code from the code listing. What the code does is first tell Python to use the GPIO module so we can connect to the GPIO pins, by importing the module.
- We then import the time module so we can create a delay between commands. We then tell the code to treat the GPIO pins as the number they are on the board and to turn the seventh pin into an output.
- We alternate between True and False so that it turns the pin on and off. Once it's cycled a few times, it will print the message 'Done' into IDLE and finally turn off the GPIO pins.

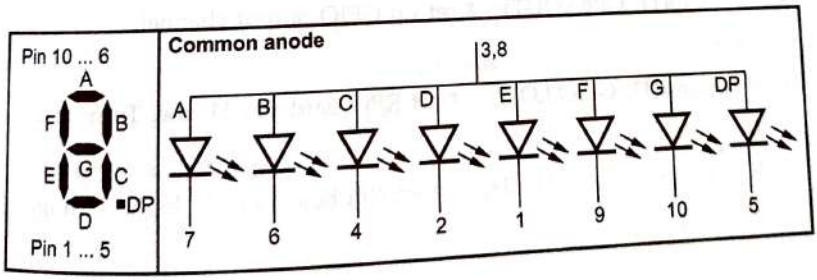


Fig. 2.7.3

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)
GPIO.setup(7, GPIO.OUT)
GPIO.output(7,True)
time.sleep(1)
GPIO.output(7,False)
time.sleep(1)
GPIO.output(7,True)
time.sleep(1)
GPIO.output(7,False)
print"Done"
GPIO.cleanup()
```

Task 1 : Turn LED on for 2 seconds and off for 1 second, loop forever. Code is given below :

(In this example, we use diagram (b), i.e. controlling the LED by controlling the voltage at the anode (+)).

```
import RPi.GPIO as GPIO
import time
def main( ) :
GPIO.cleanup( )
GPIO.setmode(GPIO.BOARD)  # to use Raspberry Pi board pin numbers
GPIO.setup(11, GPIO.OUT)  # set up GPIO output channel
while True :
GPIO.output(11, GPIO.LOW)  # set RPi board pin 11 low. Turn off LED.
time.sleep(1)
GPIO.output(11, GPIO.HIGH)  # set RPi board pin 11 high. Turn on LED.
time.sleep(2)
main( )
```

- Example : Display digit on 7-segment LED. It is most direct way to control display :

1. Connect pin 3/8 of 7-seg-LED to Vcc

2. Connect the other 8 pins to 8 GPIO pins
3. Configure the 8 GPIO pins as out

- For example : display "2". Turn on segments A, B, D, E, G and turn off segments C, F, DP. Set A, B, D, E, G to LOW and set C, F, DP to HIGH. Set Pin 7, 6, 2, 1, 10 LOW and Set pin 4, 9, 5 HIGH (Refer Fig. 2.7.3)

2.7.2 Interfacing an LED and Switch with Raspberry Pi

- When the switch is not pushed : GPIO detects Vcc (HIGH)
- When the switch is pushed : GPIO detects GND (LOW)

GPIO Input Sample Code

```
import RPi.GPIO as GPIO
# Use the pin numbers from the ribbon cable board
GPIO.setmode (GPIO.BCM)
# Set up this pin as input.
GPIO.setup (17, GPIO.IN)
# Check the value of the input pin
GPIO.input (17)
# Hold down the button, run the command again. The output should be "true".
GPIO.input(17)
```

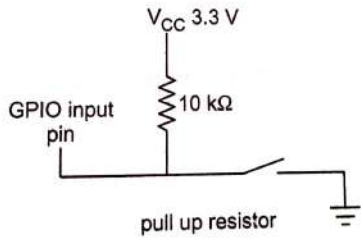


Fig. 2.7.4

2.7.3 Interfacing Light Sensor

- Unlike some other devices the Raspberry Pi does not have any analogue inputs. All 17 of its GPIO pins are digital. They can output high and low levels or read high and low levels.
- For sensors that act as a variable resistor such as LDRs (Light Dependent Resistors) or thermistors (temperature sensors) there is a simple solution. It allows

you to measure a number of levels using a single GPIO pin. In the case of a light sensor this allows you to measure different light levels.

- Fig 2.7.5 shows diagram of connecting an LDR to Raspberry Pi.

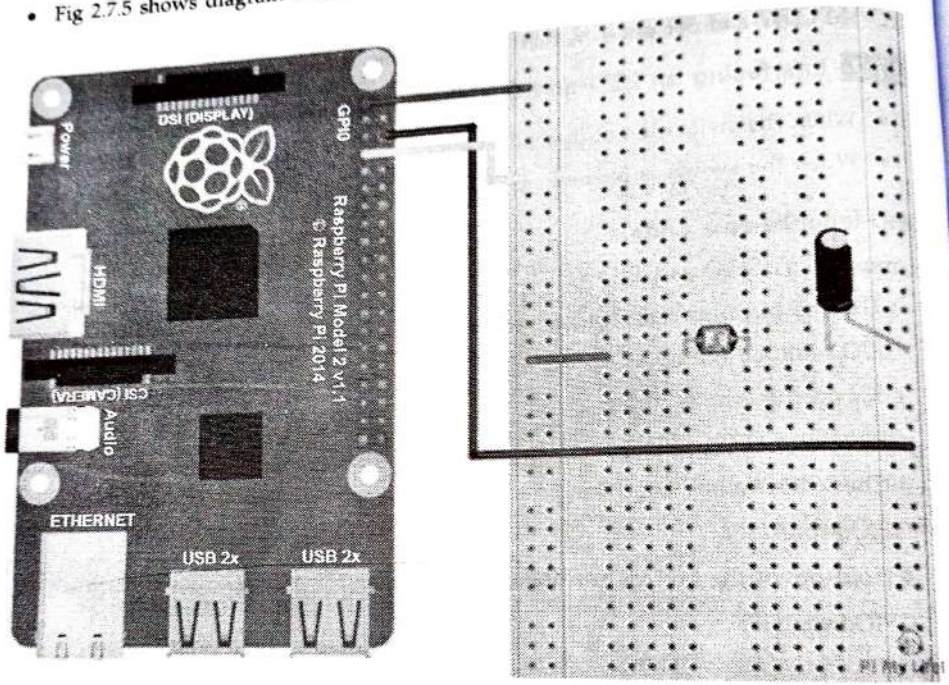


Fig. 2.7.5 Diagram of connecting an LDR to Raspberry Pi

- Following are steps :

1. First connect pin number 1 (3v3) to the positive rail on the breadboard.
2. Next connect pin number 6 (ground) to the ground rail on the breadboard.
3. Now place the LDR sensor onto the board and have a wire go from one end to the positive rail.
4. On the other side of the LDR sensor place a wire leading back to the Raspberry Pi. Hook this to pin number 7.
5. Finally place the capacitor from the wire to the negative rail on the breadboard. Make sure you have the negative pin of the capacitor in the negative rail.

- Fig 2.7.6 shows circuit diagram for above configuration.

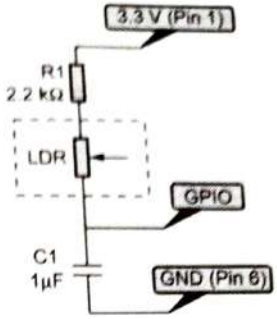


Fig. 2.7.6 Circuit diagram for LDR

The sequence of events :

1. Set the GPIO pin as an output and set it Low. This discharges any charge in the capacitor and ensures that both sides of the capacitor are 0 V.
2. Set the GPIO pin as an input. This starts a flow of current through the resistors and through the capacitor to ground. The voltage across the capacitor starts to rise. The time it takes is proportional to the resistance of the LDR.
3. Monitor the GPIO pin and read its value. Increment a counter while we wait.
4. At some point the capacitor voltage will increase enough to be considered as a High by the GPIO pin (approx 2v). The time taken is proportional to the light level seen by the LDR.
5. Set the GPIO pin as an output and repeat the process as required.

Python Code :

```
#!/usr/local/bin/python

# Reading an analogue sensor with a single GPIO pin

import RPi.GPIO as GPIO, time

# Tell the GPIO library to use

# Broadcom GPIO references

GPIO.setmode(GPIO.BCM)

# Define function to measure charge time

def RCtime (PiPin) :

 measurement = 0

 # Discharge capacitor

 GPIO.setup(PiPin, GPIO.OUT)

 GPIO.output(PiPin, GPIO.LOW)

 time.sleep(0.1)
```

```
GPIO.setup(PiPin, GPIO.IN)
# Count loops until voltage across
# capacitor reads high on GPIO
while (GPIO.input(PiPin) == GPIO.LOW) :
 measurement += 1
return measurement
# Main program loop
while True :
print RCtime(4) # Measure timing using GPIO4
```

2.8 Implementation of IoT with Edge Devices

- An edge device is a device that provides an entry point into enterprise or service provider core networks. Examples include routers, routing switches, Integrated Access Devices (IADs), multiplexers, and a variety of Metropolitan Area Network (MAN) and Wide Area Network (WAN) access devices.
- Edge devices also provide connections into carrier and service provider networks. An edge device that connects a local area network to a high speed switch or backbone may be called an edge concentrator.
- Edge Devices : A device which produces data is edge devices like machines and sensors, or any devices through which information is collected and delivered.
- Edge Gateway : It's a buffer where edge computing processing is done. The gateway is the window into the environment beyond the edge of the network.
- Edge computing aims to speed up critical data processing closer to where the device is located. It implies that data processing is done more efficiently. There is no need for the centralisation of attributes. In other words, edge computing is a mesh interface to process and store micro data centres locally.
- One of the most common types of edge devices is an edge router. Usually deployed to connect a campus network to the internet or a WAN, edge routers chiefly function as gateways between networks.
- Firewalls can also be classified as edge devices, as they sit on the periphery of one network and filter data moving between internal and external networks.
- Edge devices are comprised of hardware that performs the two essential functions of providing physical connectivity and enabling traffic between networks. A full range of edge device functions might include the transmission, routing, processing, monitoring, filtering, translation and storage of data between networks.

- Edge computing was developed due to the exponential growth of IoT devices, which connect to the internet for either receiving information from the cloud or delivering data back to the cloud. And many IoT devices generate enormous amounts of data during the course of their operations.
- Data produced by the IoT devices to be processed where it is created instead of taking away to the routes to data centers with the help of edge computing.

Why Edge Computing Matters ?

a) When IoT devices have poor connectivity.

b) Not efficient for IoT devices to be in constant touch with the central cloud.

c) The latency factor reduces latency because data doesn't have to traverse over a network to a central cloud for processing.

d) Where latencies are untenable like manufacturing or financial services.

e) As soon as data is produced, it doesn't need to send over a network; instead, it compiles the data and sends daily reports to the cloud for long term storage, i.e., reduces the data traversing.

f) Direct access to gateway into the telecom provider's network, which connects to a public IaaS cloud provider.

- Fig. 2.8.1 shows edge-centric IoT architecture.

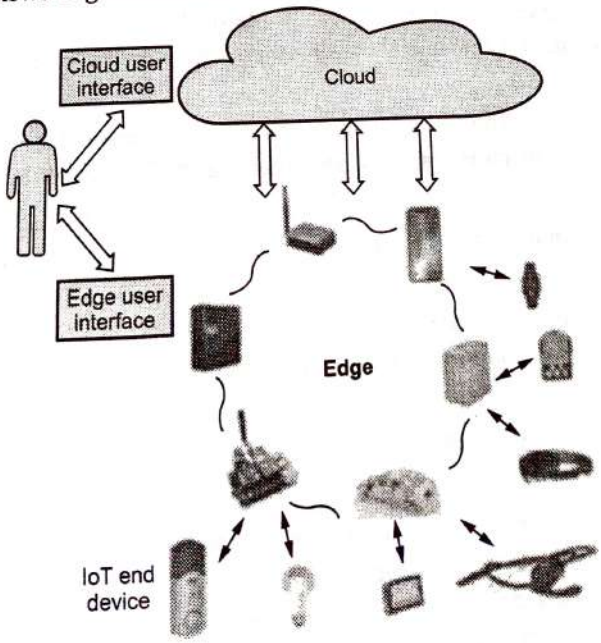


Fig. 2.8.1 Edge-centric IoT architecture

- The edge-centric IoT architecture contains four major parts : The cloud, the IoT end devices, the edge, and the users.
- The IoT end devices are deeply embedded in the physical world. They sense the physical world and take actions to control the physical world.
- In the edge-centric IoT architecture, IoT users submit queries to access IoT data or commands to control IoT devices. These queries and commands will eventually arrive at the edge layer through a web or mobile app-based interface provided by the cloud or the edge.
- Then they are handled by the edge layer, who will either forward them to IoT end devices or handle them at the edge layer on behalf of IoT end devices.
- Interacting with IoT end devices, the edge layer not only bridges them with the users and the cloud, but also can store data collected and uploaded from IoT end devices and offload substantial computational needs, such as big data analysis.
- In addition, many existing services for IoT end devices can be migrated from the cloud to the edge, and can be customized based on the needs of IoT end devices.

2.9 Reading Sensor Data and Transmit to Cloud

- Summary of most important terms which are used widely in sensor network is provided in the following :

  a) **Sensor :** A transducer that converts a physical phenomenon such as heat, light, sound or motion into electrical or other signal that may be further processed by other devices.

  b) **Sensor node :** The basic unit constituting a sensor network; it embeds a processor, a memory, a wireless interface and a local autonomous power supply.

  c) **Network topology :** A graph, where nodes are sensor nodes and edges are communication links.

  d) **Routing :** The process of forwarding data of interest along a network path from the source node to its final destination.

  e) **Resource :** This term is used to address the sensors, the communication links, the computational capabilities, the data storage and the energy amount of a node

- Sensor includes a processor which is tough enough to configure the data uploading to cloud. The Ethernet connection would connect to wired Internet service. Some places don't have wired Internet. The processor could also have the ability to update or modify the functions of the sensor. There is no involvement of radio link.

- Cloud computing is a model for enabling convenient, on demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.
- In a cloud computing platform, instead of having local servers for collecting and managing information coming from applications, remotely located servers are dynamically provisioned and (re)configured, according to the actual needs.
- As an example, if we consider a wind or solar farm, the weather - related information collected by a WSN can be processed together with grid - related information to improve plant efficiency and better satisfy the power demand.
- Cloud computing also describes those applications that can be remotely accessed through the Internet. Such cloud applications exploit large data centers and powerful servers similarly to Web applications and Web services we use every day.

Cloud computing features :

1. **Service on demand :** Clients requests are satisfied automatically without the intervention of human operator.
2. **Elasticity of demand :** Available resources are used by the clients according to their own needs for a flexible period of time.
3. **Abstraction :** Actual resources (hardware and/or software) are hidden to clients. Customers of the service exploits resources offered by the provider without knowing the locations from where processed data will arrive or where will be stored.
4. **Service measurement :** The provider exploits tools for measuring the actual usage of offered services.
5. **Resource pooling :** Available services, constituting a pool of services, are assigned dynamically according to clients' requests.
6. **Network access :** The client application can run over various platforms by means of Internet access exploiting devices as mobile phones, tablets, laptops and so on.

- According to its implementation, the cloud can provide different services levels.

1. **Infrastructure as a Service (IaaS) :** This model provides basic storage and computing capabilities as standardized services over the network. Consequently, the user (on the client side) does not need to buy its own hardware. The user would typically execute its own applications exploiting the workload offered by the infrastructure. As an example, consider services offered by Amazon EC2.

2. **Platform as a Service (PaaS)** : This model provides application software and/or development environment as a service; in addition, other higher level software applications can be executed exploiting the same service.
   Consequently, the user has the freedom to create his own applications, that are executed by the infrastructure of the provider. As an example, consider the Google's App Engine or Windows Azure.
3. **Software as a Service (SaaS)** : This model provides services to clients according to their requests. A single instance of the service is executed on the cloud and can be used by multiple end users. There is no need for investment on the client side for hardware or software licenses.

- Fig. 2.9.1 shows Sensor to Mobile - Phone Network to Cloud

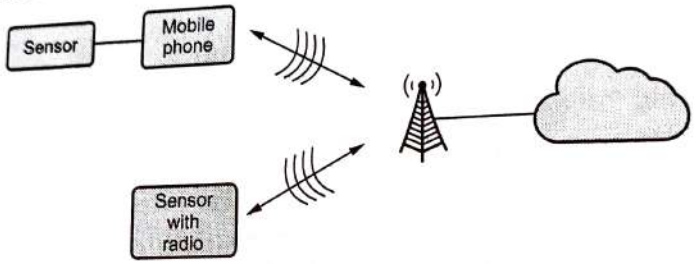


Fig. 2.9.1 Sensor to Mobile - Phone Network to Cloud

- The IoT, i.e. the capability to interconnect every possible device, opens new scenarios in WSNs. Cloud computing services and the availability of powerful and inexpensive smart devices allow to optimize information management, sharing measurement results and improving quality of services.
- The communication patterns for these devices can be categorized in the following categories : Telemetry, inquiries, commands and notifications.

1. **Telemetry** : A client device sends data one way to a cloud service.
2. **Inquiry** : A client device sends a query to the cloud service and receives a response.
3. **Command** : A cloud service issues a command to a client device and the client device returns a success or failure response.
4. **Notification** : A cloud service issues a one-way-out-of-band notification to a client device that's important for the device's operation.

2.10 Controlling Devices through Cloud using Mobile Application and Web Application

- Modern mobile devices, such as smartphones and tablets, have made many pervasive computing dreams come true. Still, many mobile applications do not perform well due to the shortage of resources for computation, data storage, network bandwidth, and battery capacity.
- While such applications can be re-designed with client - server models to benefit from cloud services, the users are no longer in full control of the application, which has become a serious concern for data security and privacy.
- Since the Internet became popular, a mobile device might overcome the constraints by offloading portions of application workload onto a server machine via the network to save execution time and conserve energy.
- Cloud computing has changed software infrastructures and business models of Internet services with technologies to provide and manage abundant resources of computation and data storage over the network at relatively low amortized operation costs.
- Fig. 2.10.1 shows Sensor to Mobile Phone to Cloud.

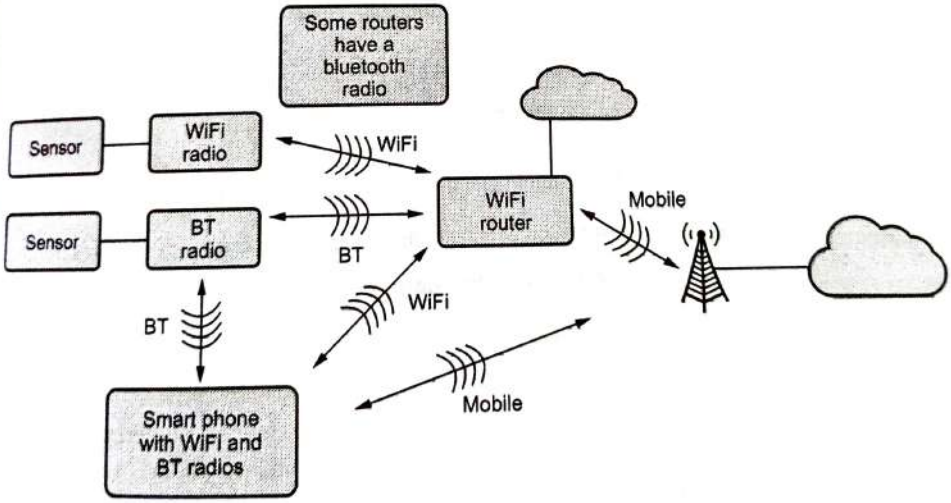


Fig. 2.10.1 Sensor to Mobile Phone to Cloud

- Application re-design and deployment : Traditional client - server models have been successfully used for mobile applications to leverage the resources in the cloud, but additional efforts, such as application partitioning and deployment of services, are required to enable dynamic, fine - grain client - server collaboration,

where the client may decide to offload a piece of work to the server during the runtime when the resources are available, if the offloading is beneficial.

- Network condition and service availability : The quality of the mobile networks is not adequate for delivering satisfactory user experiences via the collaboration between mobile devices and cloud services.
- Mobile and web applications usually require a back - end server. Web applications require a web server to deliver content. Applications also need to store user profiles and media such as images and videos. Communication between the application and the server is often done using an API, which is usually REST.

2.11 IoT Gateways

- IoT Gateway, as a hardware device or a virtual software code, acts as a communication bridge between IoT Sensor Network and Cloud Server. IoT gateway device has a layered architecture.
- Fig. 2.11.1 shows IoT gateway.

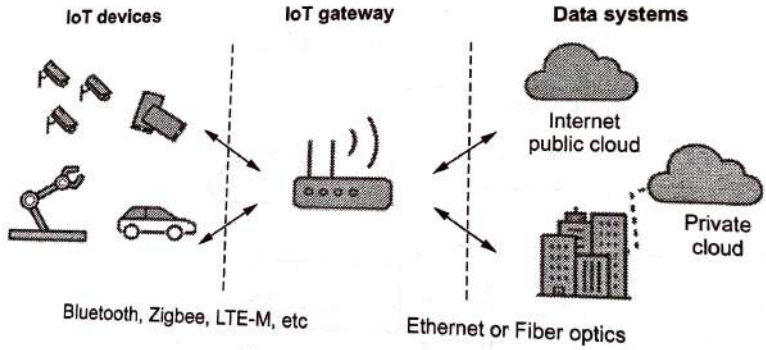


Fig. 2.11.1 IoT gateway

- IoT gateway development process :

1. **Hardware platform** : This defines the processing power & memory specifications of the IoT Gateway. This is the gateway powerhouse and a hardware platform is selected based on the complexity of IoT application(s) that need to be deployed.
2. **Operating system** : The decision of opting for a particular OS depends on the legacy systems. It is a best practice to continue to use the OS compatible with the existing systems in order to save costs and hassle-free integration.
3. **Analytics engine** : This layer ensures raw data is converted to actionable insights.

4. **Integrated application development platform and device drivers** : This layer supports development and/or addition of new devices, applications or systems to the IoT network.

- Features of IoT gateway :
  a) Facilitating communication with legacy or non-internet connected devices.
  b) Data caching, buffering and streaming.
  c) Data pre-processing, cleansing, filtering and optimization.
  d) Some data aggregation.
  e) Device to device communications.
  f) Networking features and hosting live data.
  g) Data visualization and basic data analytics via IoT gateway applications.
  h) Short term data historian features.
  i) Security - manage user access and network security features.
  j) Device configuration management.
  k) System diagnostics.
- The IoT gateway follows this simple process.
  a) Pre - processes, cleans, and filters raw data.
  b) Translates protocols for encryption and communication.
  c) Sends data to a destination on the Internet or Intranet.

2.12 Fill in the Blanks

Q.1 Raspberry Pi also allows interfacing sensors and actuators through the general purpose _____.

Q.2 Raspberry Pi is based on an _____ processor.

Q.3 Raspberry Pi has _____ status LED.

Q.4 _____ port on Raspberry Pi provides both video and audio output.

Q.5 Raspberry Pi comes with a standard _____ Ethernet port.

Q.6 USB ports on Raspberry Pi can provide a current upto _____.

Q.7 IoT Gateway, as a hardware device or a virtual software code, acts as a communication bridge between IoT Sensor Network and _____.

Q.8 IoT gateway device has a _____ architecture.

Q.9 Sensor converts physical energy into _____ energy.

Q.10 ARM means _____.

Q.11 The disadvantages of Raspberry Pi is, it does not have a _____ associated with it.

Q.12 In GPIO, DNC stands for _____.

Q.13 The BCM is also referred _____.

Q.14 GPIO stands for _____.

Q.15 ARM means _____.

Q.16 In ARM, word means _____ bits.

Q.17 CPSR stands for _____.

Q.18 The Raspberry Pi has a _____ interface to allow it to perform serial data communications.

Q.19 The _____ rate is calculated from bits per second, which includes the start, data, parity, and stop bits.

Q.20 Linux _____ command is used to create empty files or change time stamp for existing files.

Q.21 The head command displays the _____ of a file.

2.13 Multiple Choice Questions

Q.1 Arduino Uno has _____ digital I/O pins and _____ analog input pins.

a 8, 16 b 4, 8 c 14, 6 d 6, 14

Q.2 _____ pin is used to provide an input voltage when the Arduino is using an external power source.

a VIN b VOUT c USB d REST

Q.3 The pins A5 and A4 can support _____ communication.

a serial b parallel c analog d TWI

Q.4 Raspberry Pi uses an _____ processor.

a embedded b UNO c ARM d microprocessor

Q.5 Model B Raspberry Pi has _____ SDRAM.

a 64 MB b 128 MB c 256 MB d 512 MB

Q.6 The Raspberry Pi models A support _____ number of USB port.

a 1 b 2 c 4 d 8

Q.7 An Accelerated Processing Unit (APU) combines the _____ CPU and _____ GPU onto a single chip

a CPU, GPU b CPU, RAM c GPU, RAM d GPU, SoC

Q.8 Raspberry Pi is a _____ pin expansion header.

a 16 b 24 c 40 d 48

Q.9 The _____ command displays the beginning of a file.

a ls b cp c tail d head

Q.10 _____ is a server-side platform built on Google Chrome's JavaScript Engine.

a Python b Node.js c GPIO d SPI

Q.11 Which of the following is NOT Raspberry Pi Interface ?

a UART b GPIO c SPI d FPGA

Q.12 What the distributions are supported by Raspberry Pi ?

a Arch Linux b Debian
c Fedora Remix d Arch Linux, Debian, and Fedora Remix

Q.13 Which instruction set architecture is used in Raspberry Pi ?

a X86 b MSP c AVR d ARM

Q.14 Raspberry Pi comes with a standard _____ Ethernet port.

a RJ11 b RJ45
c RJ11 and RJ45 d RJ5

Q.15 Which of the following is NOT Raspberry Pi Interface ?

a UART b GPIO c SPI d FPGA

Q.16 Raspberry Pi is a _____ pin expansion header.

a 16 b 24 c 40 d 48

Q.17 Raspberry Pi is a low cost mini - computer with the physical size of _____.

a credit card b micro card
c nano card d SIM card

Q.18 The disadvantages of Raspberry Pi is, it does not have a _____ associated with it.

a memory b hard disk c USB port d ALU

Answer Keys for Fill in the Blanks

1.	I/O pins	2.	ARM
3.	five	4.	HDMI
5.	RJ45	6.	100 mA
7.	Cloud Server	8.	layered
9.	electrical	10.	Advanced RISC Machines
11.	hard disk	12.	Do Not Connect
13.	Broadcom chip-specific pin numbers	14.	General Purpose Input/Output
15.	Advanced RISC machines	16.	32
17.	Current Program Status Register	18.	UART
19.	baud	20.	touch
21.	beginning		

Answer Keys for Multiple Choice Questions

Q.1	c	Q.2	a	Q.3	d	Q.4	c
Q.5	d	Q.6	a	Q.7	a	Q.8	c
Q.9	d	Q.10	b	Q.11	d	Q.12	d
Q.13	d	Q.14	b	Q.15	d	Q.16	c
Q.17	a	Q.18	b				

□□□

3 IoT Protocols

Syllabus

Link layer protocols, Network/internet layer protocols, Transport layer protocols, Application layer protocols: Hypertext transfer protocol (HTTP), Systematic HTTP access methodology, Web Socket, Constrained application protocol (CoAP), Message Queue Telemetry Transport Protocol (MQTT), XMPP, DDS, AMQP

Contents

3.1 Introduction of IoT Protocol

3.2 IEEE 802.15.4

3.3 LoRaWAN

3.4 IPv4

3.5 IPv6

3.6 Transport Protocol

3.7 Application Layer Protocols

3.8 Fill in the Blanks

3.9 Multiple Choice Questions

3.1 Introduction of IoT Protocol

- IoT protocols are an integral part of the IoT technology stack. IoT protocols and standards are broadly classified into two separate categories. These are :
  a) IoT data protocols (Presentation / Application layers)
  b) Network protocols for IoT (Datalink / Physical layers)
- IoT data protocols are used to connect low-power IoT devices. They provide communication with hardware on the user side, without the need for any internet connection. The connectivity in IoT data protocols and standards is through a wired or cellular network.
- A protocol is a standard set of regulations and requirements that allow two electronic items to connect to and exchange information with one another. Protocols regulate data transmission among devices as well as within a network of linked devices, through both error control and specifying which data compression method to use.

3.1.1 Link Layer Protocols

- Link layer protocols determines how the data is physically sent over the network's physical layer. Link layer determines how the packets are coded and signalled by the hardware device over the medium to which the host is attached.
- Link layer protocols are Ethernet, 802.11, 802.16, 802.15.4, mobile communication etc.

3.1.2 Network / Internet Layer Protocols

- The network layer is responsible for the delivery of packets from the source to destination.
- Network layer uses IP address to choose one host among millions of host. In network layer, datagram needs a destination IP address for delivery and a source IP address for a destination reply.

3.1.3 Transport Layer

- A transport layer protocol provides for logical communication between application processes running on different hosts.
- The transport layer is responsible for delivery of message from one process to another. The network does the host to destination delivery of individual packets considering it as independent packet.
- But transport layer ensures that the whole message arrives intact and in order with error control and process control.

- A transport protocol can offer reliable data transfer service to an application even when the underlying network protocol is unreliable, even when the network protocol loses, garbles and duplicate packets.

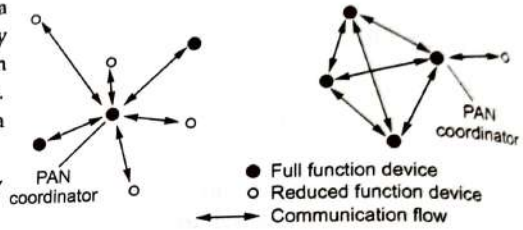
3.1.4 Application Layer

- Application layer is responsible for accessing the network by user.
- It provides user interfaces and other supporting services such as e-mail, remote file access, file transfer, sharing database, message handling (X.400), directory services (X.500).

3.2 IEEE 802.15.4

- 802.15.4 is an IEEE standard for PHY and MAC layers upon which further networking and communication protocols are built. Networking standards built upon 802.15.4 include Thread and Zigbee.
- The IEEE 802.15.4 protocol is designed for enabling communication between compact and inexpensive low power embedded devices that need a long battery life.
- It defines standards and protocols for the physical and link (MAC) layer of the IP stack. It supports low power communication along with low cost and short-range communication.
- In the case of such resource constrained environments, we need a small frame size, low bandwidth, and low transmit power.
- Characteristic of 802.15.4
  1. Data rates of 250 kbps, 20 kbps and 40 kbps.
  2. Star or peer-to-peer operation.
  3. Support for low latency devices.
  4. CSMA-CA channel access.
  5. Dynamic device addressing.
  6. Fully hand-shaked protocol for transfer reliability.
  7. Low power consumption.
  8. 16 channels in the 2.4 GHz ISM band, 10 channels in the 915 MHz ISM band and one channel in the European 868 MHz band.
- IEEE 802.15.4 is simple packet data protocol for lightweight wireless networks. Channel access is via Carrier Sense Multiple Access with collision avoidance and optional time slotting. It provides multi-level security.

- It works well for long battery life, selectable latency for controllers, sensors, remote monitoring and portable electronics. Configured for maximum battery life, has the potential to last as long as the shelf life of most batteries.
- There are two different device types : A Full Function Device (FFD) and a Reduced Function Device (RFD).
- The FFD can operate in three modes serving : Device, Co-coordinator, PAN co-coordinator. It must have memory sufficient to store routing information as required by the algorithm employed by the network.
- The RFD can only operate in a device mode. An RFD is very low cost device with minimal memory requirements. It can only function as a network device.
- Fig. 3.2.1 shows Star topology and peer-to-peer network.



a) Stratopology **b) Peer to Peer technology**

Fig. 3.2.1

- Network Device : An RFD or FFD implementation containing an IEEE 802.15.4 medium access control and physical interface to the wireless medium.
- Coordinator : An FFD with network device functionality that provides coordination and other services to the network.
- PAN Coordinator : A coordinator that is the principal controller of the PAN. A network has exactly one PAN coordinator.

3.2.1 802.15.4 Physical Layer

- Original physical layer transmission options are as follows :
  1. 2.4 GHz, 16 channels with a data rate of 250 kbps
  2. 915 MHz, 10 channels with a data rate of 40 kbps
  3. 868 MHz, 1 channels with a data rate of 20 kbps
- 802.15.4 only addresses the physical (PHY) and media access control (MAC) layers in the OSI network model, layers one and two. It leaves the upper layers to the implementer.
- Physical layer (PHY) provides two services : the PHY data service and the PHY management service.
- The PHY data service enables the transmission and reception of PHY Protocol Data Units (PPDUs) across the physical radio channel.

- At the physical layer, IEEE 802.15.4 manages the RF transceiver and channel selection, as well as energy and signal management facilities.
- There are six PHYs currently defined, depending on the frequency range and data performance required. Four of them use Direct Sequence Spread Spectrum (DSSS) frequency hopping techniques.
- Chirp Spread Spectrum (CSS) is in use in the Ultra-Wide Band (UWB) and 2450 MHz frequency bands. Parallel Sequence Spread Spectrum (PSSS) is available only with the hybrid binary/amplitude shift keying modulation technique found in the European 868 MHz band.
- Functions of physical layer :
  1. Activation and deactivation of the radio transceiver.
  2. Energy detection within the current channel.
  3. Link quality indication for received packets.
  4. Clear channel assessment for CSMA-CA.
  5. Channel frequency selection.
  6. Data transmission and reception.
- The IEE 802.15.4 standard incorporates two physical layers :
  a. The lower band : 868 MHz/915 MHz Direct Sequence Spread Spectrum (DSSS) physical (11 channels). 1 channel (20 kB/s) in European 868 MHz band and 10 channels (40 kB/s in 915 (902-928) MHz ISM band.
  b. The upper band : 2450 MHz DSSS physical (16 channels).

3.2.2 MAC Layer

- The IEEE 802.15.4 MAC layer is responsible for :
  1. Joining and leaving the PAN;
  2. Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA) for channel access;
  3. Guaranteed Time Slot (GTS) transmissions;
  4. Establishing a reliable link between two peer MAC entities;
  5. Beacon transmissions for a coordinator;
  6. Synchronization to the beacons.
- In addition, the MAC layer supports the use of symmetric encryption using the AES-128 encryption algorithm.
- The features of the IEEE 802.15.4 MAC are
  1. Association and disassociation,
  2. Acknowledged frame delivery,

3. Channel access mechanism,
4. Frame validation,
5. Guaranteed time slot management,
6. Beacon management.

- The IEEE 802.15.4 MAC provides services to an IEEE 802.2 type-I LLC through the service-specific convergence sublayer (SSCS). IEEE 802.15.4 fits into the ISO OSI reference model. Fig. 3.2.2 shows 802.15.4 device architecture.
- The MAC sublayer provides two services to higher layers that can be accessed through two service access points (SAPs).
- The MAC data service through the MAC common part sublayer (MCPS-SAP). The MAC management service through the MAC layer management entity (MLME-SAP). These two services provide an interface between the SSCS or another LLC and the PHY layer

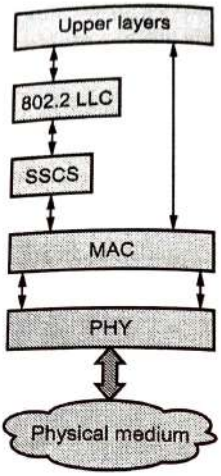


Fig. 3.2.2 IEEE 802.15.4 device architecture

- The upper layers consist of a network layer, which provides network configuration, manipulation, and message routing, and an application layer, which provides the intended function of the device.
- The PHY layer provides services to the MAC layer. MAC layer uses packets or frames as the basic unit of transmission. A frame has a structure. From the PHY layer perspective, a frame is just a block of bits. It's function is to modulate and demodulate the carrier with the provided block of bits (frame).
- The receiver, at the PHY layer, must know certain properties of an incoming waveform to make sense of it and detect a frame (frequency, phase, start and end of bits/symbols, and start and end of frames). In other words; it need to be in sync with the transmitter.
- Carrier processing involves use of oscillators and local clocks. Several factors (fabrication process, temperature differences, aging effects, etc) deviate oscillators frequencies from their nominal values. This drift is expressed in ppm (parts per millions).

- The frame size for 802.15.4 is 133 bytes including PHY, MAC, and the data payload.
- IEEE STD 802.15.4 specifies the RF, PHY and MAC layers. The IEEE 802.15.4 MAC provides services to an IEEE 802.2 type I LLC through the service-specific convergence sublayer (SSCS). IEEE 802.15.4 fits into the ISO OSI reference model.
- By keeping the frame relatively short, we can limit the amount of time needed to transmit it while simultaneously limiting the probability of radio interference due to the normal operation of industrial equipment.
- Fig. 3.2.3 shows MAC frame format.

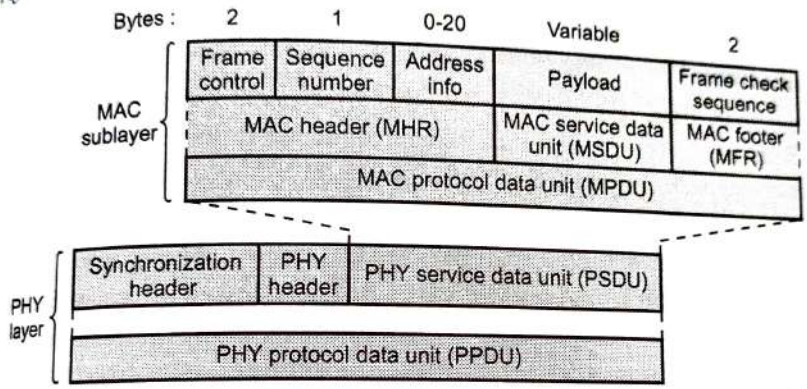


Fig. 3.2.3 MAC frame format

- The PHY uses Carrier Sense Multiple Access (CSMA) with Collision Avoidance (CA) to access the radio channel.
- Physical layer (PHY) provides two services : The PHY data service and the PHY management service.
- The PHY data service enables the transmission and reception of PHY protocol data units (PPDUs) across the physical radio channel.
- The MAC sublayer provides two services : The MAC data service and the MAC management service interfacing to the MAC sublayer management entity (MLME) service access point (SAP). The MAC data service enables the transmission and reception of MAC protocol data units (MPDUs) across the PHY data service.
- The features of the MAC sublayer are beacon management, channel access, GTS management, frame validation, acknowledged frame delivery, association, and disassociation.

- The frame size for 802.15.4 is 133 bytes including PHY, MAC, and the data payload.
- IEEE STD 802.15.4 specifies the RF, PHY and MAC layers. The IEEE 802.15.4 MAC provides services to an IEEE 802.2 type I LLC through the service-specific convergence sublayer (SSCS). IEEE 802.15.4 fits into the ISO OSI reference model.
- By keeping the frame relatively short, we can limit the amount of time needed to transmit it while simultaneously limiting the probability of radio interference due to the normal operation of industrial equipment.
- Fig. 3.2.3 shows MAC frame format.

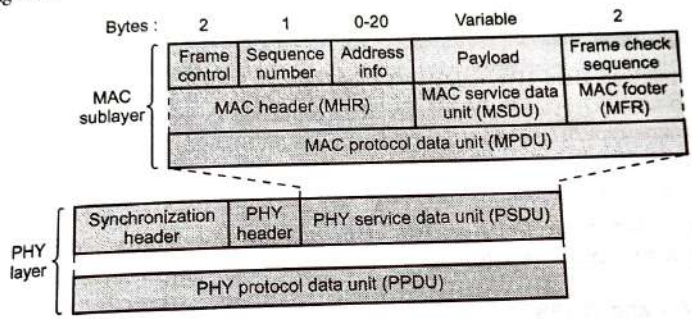


Fig. 3.2.3 MAC frame format

- The PHY uses Carrier Sense Multiple Access (CSMA) with Collision Avoidance (CA) to access the radio channel.
- Physical layer (PHY) provides two services : The PHY data service and the PHY management service.
- The PHY data service enables the transmission and reception of PHY protocol data units (PPDUs) across the physical radio channel.
- The MAC sublayer provides two services : The MAC data service and the MAC management service interfacing to the MAC sublayer management entity (MLME) service access point (SAP). The MAC data service enables the transmission and reception of MAC protocol data units (MPDUs) across the PHY data service.
- The features of the MAC sublayer are beacon management, channel access, GTS management, frame validation, acknowledged frame delivery, association, and disassociation.

- **Frame control** specifies how the rest of the frame looks and what it contains. It indicates the type of MAC frame being transmitted and specifies the format of the address field. It also controls the acknowledgment.
- The **sequence number** matches the acknowledgment frame with the previous transmission.
- The size of the **address field** may vary between 0 and 20 bytes. Data frame may contain both source and destination information. Return acknowledgment frame does not contain any address information at all. Beacon frame may only contain source address information.
- Short 8-bit device addresses or 64-bit IEEE device addresses may be used. This flexible structure helps increase the efficiency of the protocol by keeping the packets short.
- The **payload field** is variable in length; however, the complete MAC frame may not exceed 127 bytes in length. The data contained in the payload is dependent on the frame type. The **Frame Check Sequence** (FCS) helps verify the integrity of the MAC frame.
- The FCS in an IEEE 802.15.4 MAC frame is a 16-bit International Telecommunication Union - Telecommunication Standardization Sector (ITU-T) cyclic redundancy check (CRC).

3.2.3 FHSS and DHSS

- A family of wireless LAN (WLAN) specifications developed by a working group at the Institute of Electrical and Electronic Engineers (IEEE). It defines standard for WLANs using the following four technologies
  1. Frequency Hopping Spread Spectrum (FHSS)
  2. Direct Sequence Spread Spectrum (DSSS)
  3. Infrared (IR)
  4. Orthogonal Frequency Division Multiplexing (OFDM)

Frequency Hopping Spread Spectrum (FHSS)

- Used in Bluetooth. Available spectrum is subdivided into a number of equal-sized sub-bands or channels.
- It uses frequency hopping spread spectrum method. It also uses 79 non-overlapping 1 MHz channels to transmit a 1 Mbps data signal over the 2.4 GHz ISM band.

- A pseudorandom number generator selects the hopping sequence. The modulation technique in this specification is either two-level FSK or four-level FSK with 1 or 2 bits/baud, which results in a data rate of 1 or 2 Mbps.
- Within North America, FHSS uses seventy nine 1 MHz channels centered on every 1 MHz from 2.400 GHz to 2.4835 GHz.
- FHSS systems use a small frequency bandwidth, which I will simply call a frequency, within the 79 MHz allocated, to communicate and then hop to another frequency and then another until a hopping pattern known as a hopping sequence has been completed.
- When the hopping sequence is completed, it is then repeated, and this process continues until the information being communicated has been transferred. Additionally, a dwell time is specified, which determines how long each frequency will be utilized before hopping to the next position in the hopping sequence.
- User data is always transmitted within one channel at a time; it's bandwidth is thus limited. All nodes in the network hop synchronously through the channels according to a prespecified schedule. Different networks can share the same geographic area by using non-overlapping hopping schedules.

Direct Sequence Spread Spectrum (DSSS)

- Used in IEEE 802.11 and IEEE 802.15.4.
- It uses the direct sequence spread spectrum method. DSSS uses the 2.4 GHz ISM band. The modulation technique is this specification is PSK at 1 Mbaud/s. The system allows 1 or 2 bits/buad which result in a data rate of 1 or 2 Mbps.
- The IEEE specifies that DSSS should operate in the 2.4 GHz ISM band and that it should use frequencies ranging from 2.401 to 2.473 GHz in North America.
- The IEEE further specifies that the DSSS supported by IEEE 802.11 devices should implement differential binary phase shift keying (DBPSK) at 1 Mbps and differential quadrature phase shift keying (DQPSK) at 2 Mbps. DBPSK and DQPSK are modulation techniques that use phase based modulation.
- The IEEE standards divide the DSSS Physical layer into two components : the Physical Layer Convergence Procedure (PLCP) and the Physical Medium Dependent (PMD). The PMD defines that actual method used to transmit data between two wireless devices, and the PLCP acts as an abstraction layer between the PMD and the Medium Access Control (MAC) services.

3.2.4 Choice of 802.15.4 Communication channel

- IEEE 802.15.4 was developed to meet the needs for simple, low-power and low-cost wireless communication.
- The 802.15.4 standard specifies that communication can occur in the 868-868.8 MHz, the 902-928 MHz or the 2.400-2.4835 GHz Industrial Scientific and Medical (ISM) bands.
- While any of these bands can technically be used by 802.15.4 devices, the 2.4 GHz band is more popular as it is open in most of the countries worldwide.
- The 868 MHz band is specified primarily for European use, whereas the 902-928 MHz band can only be used in the United States, Canada and a few other countries and territories that accept the FCC regulations.
- For interference immunity, 802.15.4 specifies the use of Direct Sequence Spread Spectrum (DSSS) and uses an Offset Quadrature Phase Shift Keying (O-QPSK) with half-sine pulse shaping to modulate the RF carrier.
- 802.15.4 is commonly used in lighting, digital signage, industrial automation, and simple/fast networking where the leaner of code of the 802.15.4 protocol means that the network is faster, it has more bandwidth and less latency than the protocols that are built on top of the 802.15.4 stack such as ZigBee and DigiMesh.

3.2.5 Beacon Enabled Mode

- All IEEE 802.15.4-based networks use beacons from a Co-ordinator when joining devices to the network.
- The Co-ordinator sends out a periodic train of beacon signals containing information that allows network nodes to synchronise their communications. A beacon also contains information on the data pending for the different nodes of the network.
- Normally, two successive beacons mark the beginning and end of a superframe. A superframe contains 16 timeslots that can be used by nodes to communicate over the network. Fig 3.2.4 shows superframe.

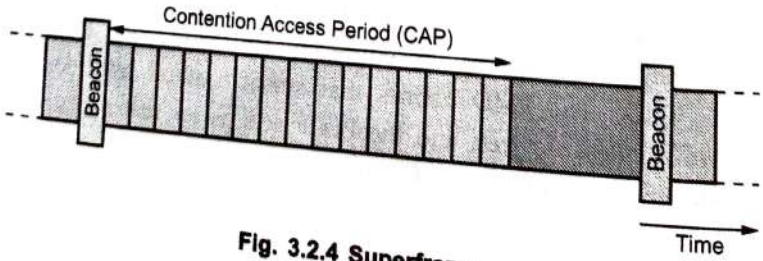


Fig. 3.2.4 Superframe

- The total time interval of these timeslots is called the Contention Access Period (CAP), during which nodes can attempt to communicate using slotted CSMA/CA

Guaranteed Time Slot Concept

- A Guaranteed Time Slot (GTS) allows a device to operate on the channel within a portion of the superframe. A GTS shall only be allocated by the PAN co-ordinator. The PAN co-ordinator can allocated up to seven GTSs at the same time. The PSN co-ordinator decides whether to allocate GTS based on :

1. Requirements of the GTS request
2. The current available capacity in the superframe.

- A GTS can be deallocated.

1. At any time at the discretion of the PAN co-ordinator or
2. By the device that originally requested the GTS.

- A device that has been allocated a GTS may also operate in the CAP. A data frame transmitted in an allocated GTS shall use only short addressing. The PAN co-ordinator shall be able to store the info of devices that necessary for GTS, including starting slot, length, direction and associated device address. Fig. 3.2.5 shows superframe with GTS.

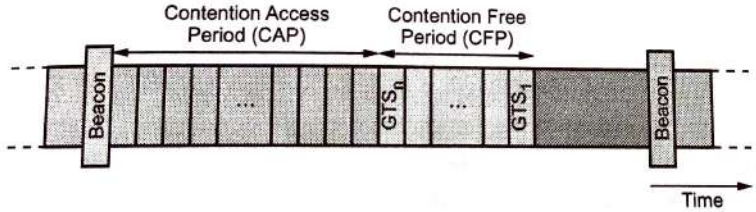


Fig. 3.2.5 Superframe with GTSs

- Before GTS starts, the GTS direction shall be specified as either transmit or receive. Each device may request one transmit GTS and/or one receive GTS. A device shall only attempt to allocate and use a GTS if it is currently tracking the beacon.
- If a device loses synchronization with the PAN co-ordinator, all its GTS allocations shall be lost. The use of GTSs be an RFD is optional.

3.2.6 Non-Beacon Enabled Mode

- In non-beacon enabled mode, beacons are not transmitted on a regular basis by the Co-ordinator. Instead, communications are asynchronous - a device communicates with the Co-ordinator only when it needs to, which may be relatively infrequently. This allows power to be conserved.
- To determine whether there is data pending for a node, the node must poll the Co-ordinator.

- Non-beacon enabled mode is useful in situations where only light traffic is expected between the network nodes and the Co-ordinator. In this case, the use of regular beacons may not be needed and will waste valuable power.

3.3 LoRaWAN

- LoRaWAN is a low power wide area network (LPWAN) specification intended for wireless battery operated things in a regional, national or global network.
- LoRaWAN provides long-range (up to 15km) communication between sensors and base stations, resulting in networks with 2-3x times fewer base stations compared to cellular.
- Fully bidirectional communication enables a wide variety of uses cases requiring uplinks and downlinks: for example, street lighting, smart irrigation, energy optimization or home automation
- LoRaWAN targets key requirements of the internet of things (IoT) such as secure bi-directional communication, mobility and localization services.
- LoRaWAN networks are deployed on cost-free ISM bands allowing any service provider to deploy and operate LoRaWAN networks without having to acquire a license for any frequency.
- The LoRaWAN specification provides seamless interoperability among smart things without the need of complex local installations, enabling the roll out of IoT applications, according to the association.
- In the LoRaWAN network architecture, gateways are connected to the network server via standard IP connections while end-devices use single-hop wireless communication to one or many gateways.
- All end-point communication is generally bi-directional, but also supports operation such as multicast, enabling software upgrades over the air or other mass distribution messages to reduce the on-air communication time.
- LoRaWAN operates in unlicensed radio spectrum. It uses lower radio frequencies with a longer range.
- The LoRaWAN specification defines three device types. All LoRaWAN devices must implement Class A, whereas Class B and Class C are extensions to the specification of Class A devices.
- Class A devices support bi-directional communication between a device and a gateway. Uplink messages (from the device to the server) can be sent at any time (randomly).
- Class B devices extend Class A by adding scheduled receive windows for downlink messages from the server. Using time-synchronized beacons transmitted by the gateway, the devices periodically open receive windows.

- Class C devices extend Class A by keeping the receive windows open unless they are transmitting, as shown in the figure below. This allows for low-latency communication but is many times more energy consuming than Class A devices.
- Fig. 3.3.1 shows LoRaWAN layers.

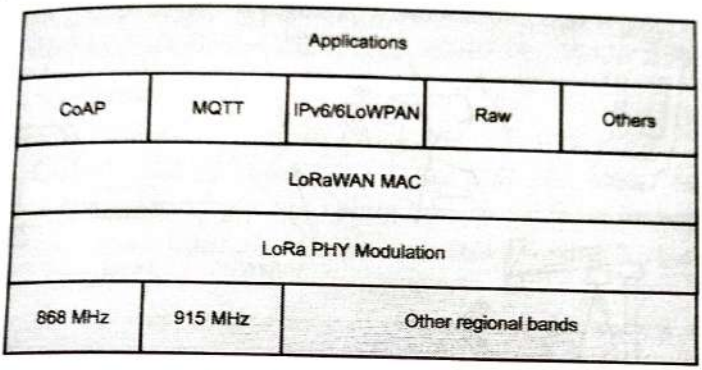


Fig. 3.3.1 LoRaWAN layers

- The LoRaWAN protocol was defined specifically for LPWAN applications, keeping security, scalable capacity, cost, and ease of deployment in mind.
- The gateways support bidirectional communication and can simultaneously process messages from many LoRa-based sensor nodes.
- A typical LoRaWAN network uses a simple architecture where LoRaWAN base stations connect to the internet through a variety of available backhauls and manage the bidirectional data flow between LoRa sensors and the centralized ThingPark network server provided by Activity.
- Fig. 3.3.2 shows architecture of LoRaWAN.
- End Nodes transmit directly to all gateways within range, using LoRa.
- Gateways relay messages between end devices and a central network server using IP.
- It is designed to allow low-powered devices to communicate with Internet-connected applications over long range wireless connections.
- LoRaWAN messages, either uplink or downlink have a PHY payload composed of a 1-Byte MAC header, a variable byte MAC payload and MIC that is 4 byte in length.
- MAC payload size depends on the frequency band and data rate, ranging from 59 to 230 bytes for the 863-870MHz and 19 to 250 bytes for the 902-928MHz band.

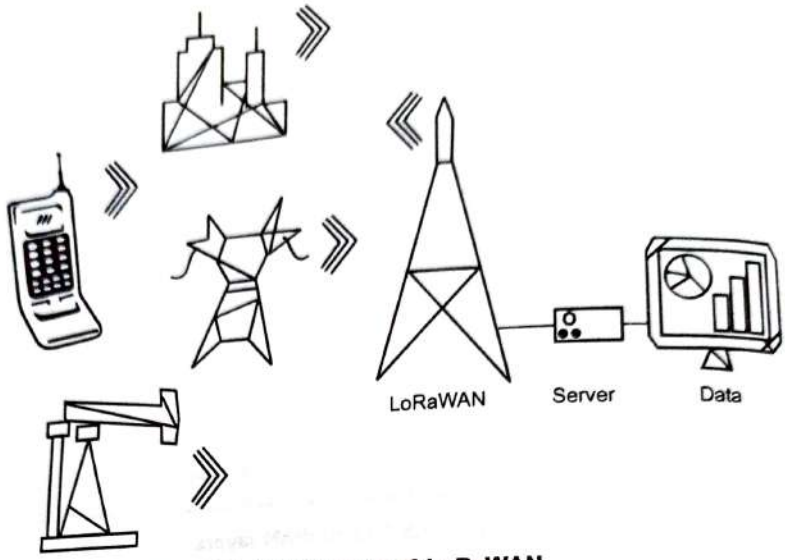


Fig. 3.3.2 Architecture of LoRaWAN

- LoRaWAN endpoints are uniquely addressable through a variety of methods, including the following :
  1. An endpoints can have a global end device ID
  2. An endpoints can have a global application ID
  3. In a LoRaWAN network, endpoints are also known by their end device address, known as a DevAddr a 32-bit address.

Node Type :

Class Type A	Class Type B	Class Type C
Battery Powered	Low Latency	Low Latency
Unicast Messages	Unicast and Multicast Messages	Unicast and Multicast Messages
Server communicates with end device (downlink) during predetermined response windows	Server can initiate transmission at fixed intervals	End-device is constantly receiving
End Device Initiates Communication (Uplink	Extra Receive Window	Server can initiate transmission at any time

Bidirectional with 1 UL+2DL Slot	Bidirectional with Scheduled Downlink Slots	Bidirectional with Most of the Time Listening Mode
• Small Payloads • Long Intervals	• Small Payloads • Long Intervals • Periodic Beacon from Gateway	• Small Payloads

Advantages of LoRaWAN

1. Low Powered sensors that can cover a wide area measured in miles.
2. Operates in the industrial, scientific, and medical (ISM) radio bands. These are free (unlicensed) frequencies, having no upfront licensing cost to use the technology.
3. Low power means long battery life for devices. Sensor batteries can last for two to five years (Class A and Class B).
4. Single LoRa Gateway device is designed to take care of thousands of end devices or nodes.
5. Perfect for monitoring field deployed assets.
6. It is widely used for M2M/IoT applications.
7. LoRaWAN is governed by an alliance.
8. Long-range functionality enables smart solutions, such as smart city applications.
9. Wireless, easy to set up and fast deployment.
10. Security : a layer of security for the network and one for the application with AES encryption.
11. Fully bi-directional communication.

Disadvantages of LoRaWAN

1. Not for large data payloads
2. Has no support for audio or video
3. Limited to Line of Sight (LOS) communication
4. Not for continuous monitoring (except Class C devices).
5. Not ideal candidate for real time applications requiring lower latency and bounded jitter requirements.

3.4 IPv4

- IPv4 is Internet Protocol version 4. It is the network layer protocol of the TCP/IP protocol suite.
- IP is a connectionless, unreliable, best-effort delivery protocol. All the nodes are identified using an IP address. Packets are delivered from the source to the destination using IP address
- IPv4 addresses are encoded as a 32 bits field. IPv4 addresses are often represented in dotted-decimal format as a sequence of four integers separated by a dot.
- An IPv4 address is used to identify an interface on a router or a host. IPv4 addresses are unique. Two devices on the internet can never have the same address at the same time.
- The address structure was originally defined to have a two-level hierarchy : **Network ID and host ID.**
- The network ID identifies the network the host is connected to. The host ID identifies the network connection to the host rather than the actual host.

3.4.1 Packet Format

- The IP datagram consists of a header followed by a number of bytes of data. Fig 3.4.1 shows header format of IPv4. All IPv4 packets use the 20 bytes header. (See Fig. 3.4.1 on next page.)
- **Version field** indicates the version of IP used to build the header. Current version is 4.
- **Header Length** indicates the length of the IP header in 32 bits words. This field allows IPv4 to use options if required, but as it is encoded as a 4 bits field, the IPv4 header cannot be longer than 64 bytes.
- **Type of service :** The service type is an indication of the quality of service requested for this IP datagram.
- **Protocol field** indicates the transport layer protocol that must process the packet's payload at the destination. Common values for this field are 6 for TCP and 17 for UDP
- **Length field** indicates the total length of the entire IPv4 packet (header and payload) in bytes. This implies that an IPv4 packet cannot be longer than 65535 bytes.
- **Fragment offset** is used to reassemble the full datagram. The value in this field contains the number of 64-bit segments (header bytes are not counted) contained in earlier fragments. If this is the first (or only) fragment, this field contains a value of zero.

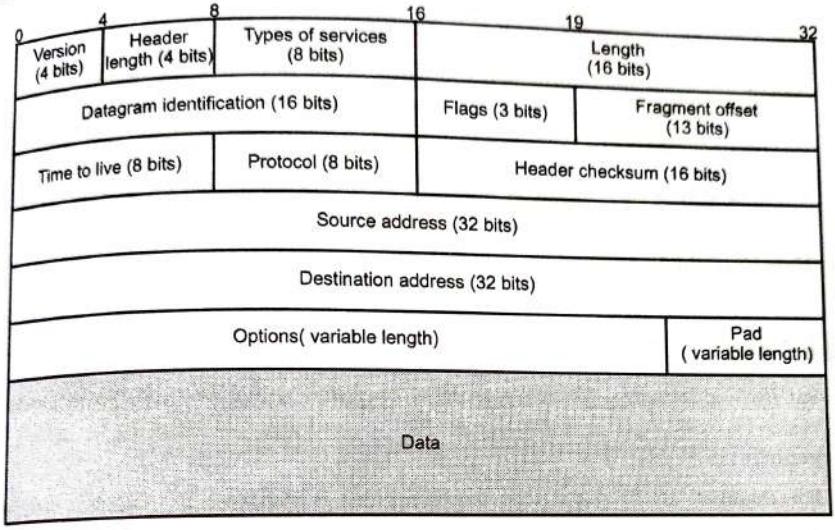


Fig. 3.4.1 : IPv4 header format

- **Flags (3 bits) :** Only two of the bits are currently defined : MF (More Fragments) and DF (Don't Fragment).
- **More Fragments (MF) flag** is a single bit in the Flag field used with the Fragment Offset for the fragmentation and reconstruction of packets. The More Fragments flag bit is set, it means that it is not the last fragment of a packet. When a receiving host sees a packet arrive with the MF = 1, it examines the Fragment Offset to see where this fragment is to be placed in the reconstructed packet. When a receiving host receives a frame with the MF = 0 and a non-zero value in the Fragment offset, it places that fragment as the last part of the reconstructed packet.
- **Don't Fragment (DF) flag** is a single bit in the Flag field that indicates that fragmentation of the packet is not allowed. If the Don't Fragment flag bit is set, then fragmentation of this packet is NOT permitted. If a router needs to fragment a packet to allow it to be passed downward to the Data Link layer but the DF bit is set to 1, then the router will discard this packet.
- **Time-to-Live (TTL)** is an 8-bit binary value that indicates the remaining "life" of the packet. The TTL value is decreased by at least one each time the packet is processed by a router (that is, each hop). When the value becomes zero, the router discards or drops the packet and it is removed from the network data flow.
- **Source address** field that contains the IPv4 address of the source host.

- **Destination address** field that contains the IPv4 address of the destination host
- **Checksum** that protects only the IPv4 header against transmission errors. Checksum is for the information contained in the header. If the header checksum does not match the contents, the datagram is discarded.
- **Options (variable)** : Encodes the options requested by the sending user.
- **Padding (variable)** : Used to ensure that the datagram header is a multiple of 32 bits.
- **Data (variable)** : The data field must be an integer multiple of 8 bits. The maximum length of the datagram (data field plus header) is 65,535 octets.

3.4.2 Classes of IP Address

- An IPv4 address is a 32-bit sequence of ones and zeros. To make the IP address easier to work with, it is usually written as four decimal numbers separated by periods.
- For example, an IP address of one computer is 192.168.1.2. This is called the dotted decimal format.
- Each part of the address is called an octet because it is made up of eight binary digits. For example, the IP address 192.168.1.8 would be 11000000.10101000.00000001.00001000 in binary notation.
- Address 0.0.0.0, 127.0.0.1 and 255.255.255.255 carries special meaning. IP address is divided into a network number and a host number. The network prefix identifies a network and the host number identifies a specific host.
- Example :

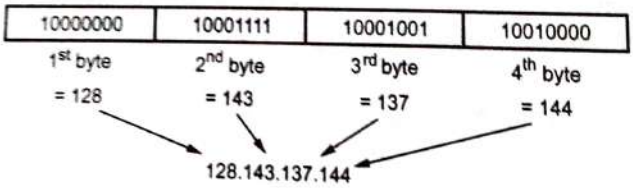


Fig. 3.4.2

- The IP address consists of a pair of numbers : **IP address = <network number><host number>**
- Example : Suppose IP address of technicalpublication.org is **129.144.136.146**, then network address is 129.144.0.0 and host number is 136.146.

Classes of IP Address

- IP address is divided into five classes : A, B, C, D, and E. Fig. 3.4.3 shows IP address classes with net ID and host ID.

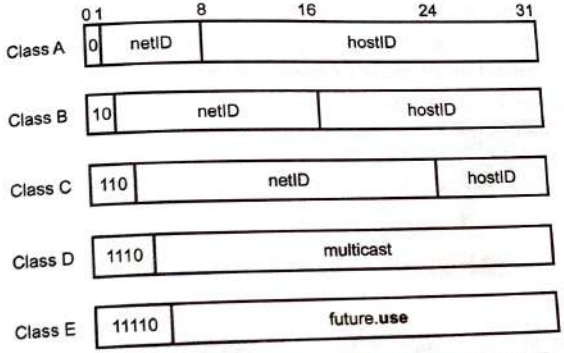


Fig. 3.4.3 : IP address classes with net ID and host ID

- The class of an IP address is identified in the most significant few bits. If the first bit is 0, it is a class A address.
- If the first bit is 1 and second is 0, it is a class B address. If the first two bits are 1 and third is 0, it is a class C address.
- Class D is used for what is known as multicasting, transmitting data to multiple computers or routers.
- Class E was reserved for future use, but this has given way to IPv6 instead.
- **Range of IP address :**

Class	Starting address	Last Address
Class A	0.0.0.0	127.255.255.255
Class B	128.0.0.0	191.255.255.255
Class C	192.0.0.0	223.255.255.255
Class D	224.0.0.0	239.255.255.255
Class E	240.0.0.0	255.255.255.255

- **Loopback Testing** : The 127 network number isn't used by hosts as a ***logical IP address***. Instead, this network is used for ***loopback IP addresses***, which allow for testing.
- Address 0.0.0.0, 127.0.0.1 and 255.255.255.255 carries special meaning.

3.4.3 Public and Private Addresses

- IPv4 addresses are further classified as either public or private. Public IP addresses are ones that are exposed to the Internet. Any other computers on the Internet can potentially communicate with them.
- Private IP addresses are hidden from the Internet and any other networks. They are usually behind an IP proxy or firewall device.
- Private Addresses are as follows :

Class	Starting Address	End of range
Class A	10.0.0.0	10.255.255.255
Class B	172.16.0.0	172.31.255.255
Class C	192.168.0.0	192.168.255.255

Subnetting

- Subnetting is the subdivision of logical IP network. By default, all computers are on one subnet or network with no divisions involved.
- Subnet mask of class A, B and C are as follows :

IP address Class	Subnet Mask
A	255.0.0.0
B	255.255.0.0
C	255.255.255.0

3.5 IPv6

- IPv4 provides the host to host communication between systems in the Internet.
- IPv4 has played a central role in the internetworking environment for many years. It has proved flexible enough to work on many different networking technologies.
- In the early 1990 the IETF began to work on the successor of IPv4 that would solve the address exhaustion problem and other scalability problems.
- IPv6 addresses are 128 bits in length. Addresses are assigned to individual interface on nodes, not to the node themselves.
- IPv6 addresses are assigned to interfaces, rather than to nodes, in recognition that a node can have more than one interface.
- A single interface may have multiple unique unicast addresses. The first field of any IPv6 address is the variable length format prefix, which identifies various categories of addresses.

- A new notation has been devised for writing 16-byte addresses. They are written as eight groups of four hexadecimal digits with colons between the groups, like this

  8000 : 0000 : 0000 : 0000 : 0123 : 4567 : 89AB : CDEF

Optimization

1. Leading zeros within a group can be omitted so 0123 can be written as 123.
2. One or more groups of 16 zero bits can be replaced by a pair of colons. The address new becomes

   8000 : : 123 : 4567 : 89AB : CDEF

Address Types

- IPv6 allows three types of addresses : Unicast, Anycast and Multicast.
- **Unicast** : An identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that address.
- **Anycast** : An identifier for a set of interfaces. A packet sent to an anycast address is delivered to one of the interfaces identified by the address.
- **Multicast** : An identifier for a set of interfaces. A packet sent to a multicast address is delivered to all interfaces identified by that address.
- The first field of any IPv6 address is the variable-length format prefix, which identifies various categories of address.

3.5.1 Packet Format

- Fig. 3.5.1 the IPv6 datagram header format. Each packet is composed of a mandatory base header followed by the payload.
- The payload consists of two parts : Optional and data.
  1. Versions : This 4 bits field defines the version number of the IP. The value is 6 for IPv6.
  2. Priority : The 4 bits priority field defines the priority of the packet with respect to traffic congestion.
  3. Flow label : It is 24 bits field that is designed to provide special handling for a particular flow of data.
  4. Payload length : The 16 bits payload length field defines the length of the IP datagram excluding the base header.
  5. Next header : It is an 8 bits field defining the header that follows the base header in the datagram.

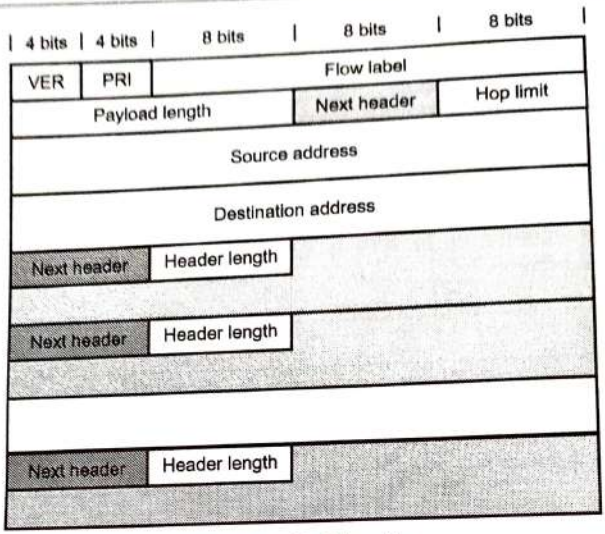


Fig. 3.5.1 : IPv6 header

6. Hop limit : This 8 bits hop limit field serves the same purpose as the TTL field in IPv4.
7. Source address : The source address field is a 128 bits internet address that identifies the original.
8. Destination address : It is 128 bits Internet address that usually identifies the final destination of the datagram.

Priority :

- The priority field defines the priority of each packet with respect to other packets from the same source. IPv6 divides traffic into two broad categories
  1. Congestion controlled
  2. Noncongestion controlled
- If a source adapts itself to traffic slowdown when there is congestion, the traffic is referred to as congestion controlled traffic congestion controlled data are assigned priorities from 0 to 7.
- A priority of 0 is the lowest; a priority of 7 is the highest.
- Noncongestion controlled traffic refers to a type of traffic that excepts minimum delay. Discarding of packets is not desirable. Retransmission in most cases is impossible. Real time audio and video are examples of this type of traffic.
- Priority numbers from 8 to 15 are assigned to noncongestion controlled traffic.

3.5.2 Difference between IPv4 and IPv6

Sr. No.	IPv4	IPv6
1	Header size is 32 bits.	Header size is 128 bits.
2	It cannot support autoconfiguration.	Supports autoconfiguration.
3	Cannot support real time application.	Supports real time application.
4	No security at network layer.	Provides security at network layer.
5	Throughput and delay is more.	Throughput and delay is less.

3.5.3 Advantages of IPv6

1. Larger address space
2. Better header format
3. Security capabilities
4. Support for resource allocation
5. New options
6. Allowance for extension

3.6 Transport Protocol

3.6.1 Bluetooth Low Energy

- Bluetooth Low Energy (BLE) is a low power wireless communication technology that can be used over a short distance to enable smart devices to communicate. It is also called Bluetooth Smart.
- The BLE technology provides an easy and a reliable interface, which is highly appreciated by consumer electronics manufacturers, mobile application developers and engineers.
- BLE uses 2.4 GHz ISM frequency band either in dual mode or single mode. Dual mode supports both bluetooth classic and low energy peripherals.
- All BLE devices use the GATT profile (Generic Attribute Profile). The GATT protocol provides series of commands for the client to discover information about BLE server.
- BLE device is acting in either a central or peripheral role and is sometimes also referred to as a client or server.

1. Central (Client) : A device that initiates commands and requests, and accepts responses. Examples : Computer, Smartphone
2. Peripheral (Server) : A device that receives commands and requests, and returns responses. Examples : A temperature sensor, Heart rate monitor

- Bluetooth device is identified by the Bluetooth device address. This address is 48-bit long. There are two types of device addresses :
  a. Public device address : This is a fixed, factory-programmed device address. It must be registered with the IEEE Registration Authority and will never get change during its entire lifetime.
  b. Random device address : This address can be pre-programmed or dynamically generated during the runtime. It has many practical uses in BLE.

Features

1. The lowest power consumption
2. Cost efficient and compatible
3. Ease of use and integration
4. Robustness, security, and reliability
5. License free
6. Support multibrand mobile

- Fig. 3.6.1 shows BLE connection.

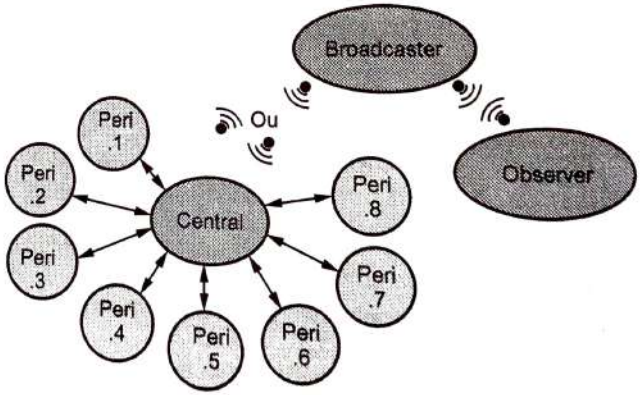


Fig. 3.6.1 : BLE connection

- A BLE connected item may have up to 4 different functions :

1. The "Broadcaster" shall be used as a server. Thus, its purpose is to transfer data to a device on a regular basis, but it does not support any incoming connection.
2. The "Observer" : In a second step, the device may only monitor and read the data sent by a "broadcaster". In such a case, the object is not able to send any connection to the server.
3. The "Central" usually consists of a smartphone or tablet. This device provides two different types of connection : either in advertising mode or in connected mode. It is leading the overall process as it triggers data transfer.

4. The "Peripheral" device allows connections and data transfer with the "Central" on a periodical basis. This system's goal is to ensure universal data transmission by using the standard process, so that other devices also may read and understand the data.

3.6.2 Components of BLE

- From architectural perspective, BLE has three components : application block, host and controller.
- Fig 3.6.2 shows BLE protocol stack.

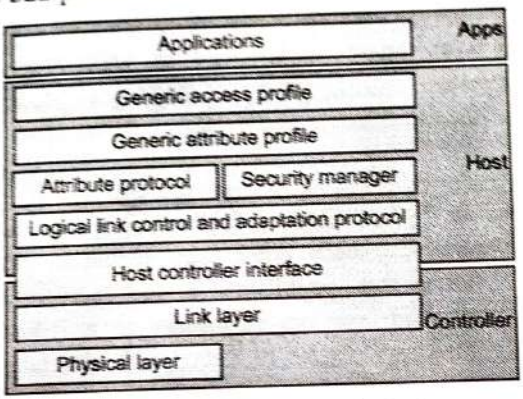


Fig. 3.6.2 : BLE protocol stack

1. **Controller**

- The controller includes the following layers :

a) Host Controller Interface (HCI) - Controller side

b) Link Layer

c) Physical Layer (PHY)

a. **Host-Controller Interface (HCI) :** It provides communication between controller and host through standard interface types. This HCI layer can be implemented either using API or by interfaces such as UART/SPI/USB. Standard HCI commands and events are defined in the bluetooth specifications.

b. **Link Layer**

- Link Layer (LL) directly interfaces with the physical layer (PHY), and is usually implemented as a combination of custom hardware and software.
- It is responsible for advertising, scanning, and creating/maintaining connections. The role of BLE devices changes in peer to peer (i.e. Unicast) or broadcast modes. The common roles are Advertiser/Scanner (Initiator), Slave/Master or Broadcaster/Observer.

c. **Physical Layer**

- The physical (PHY) layer handles the operation involves analog communications. Specifically, it defines the modulation and demodulation of analog signals and applies source coding to transform the signals into digital symbols.
- The transmitter uses GFSK modulation and operates at unlicensed 2.4 GHz frequency band. Using this PHY layer, BLE offers data rates of 1 Mbps (Bluetooth v4.2)/2 Mbps (Bluetooth v5.0).
- It uses frequency hopping transceiver. Two modulation schemes are specified to deliver 1 Msym/s and 2 Msym/s.
- Two PHY layer variants are specified viz. uncoded and coded.
- A Time Division Duplex (TDD) topology is employed in both of the PHY modes.

2. Host

a. **The Generic Access Profile (GAP) :** It provides a framework that defines how BLE devices interact with each other. This includes : Roles of BLE devices, Advertisements, Connection establishment and Security. This layer directly interfaces with application layer and/or profiles on it. It handles device discovery and connection related services for BLE device. It also takes care of initiation of security features.

b. **The Generic Attribute Profile (GATT)** defines the format of the data exposed by a BLE device. It also defines the procedures needed to access the data exposed by a device. This layer is service framework which specifies sub-procedures to use ATT. Data communications between two BLE devices are handled through these sub-procedures. The applications and/or profiles will use GATT directly.

- There are two Roles within GATT : Server and Client. The Server is the device that exposes the data it controls or contains, and possibly some other aspects of its behaviour that other devices may be able to control. A Client, on the other hand, is the device that interfaces with the Server with the purpose of reading the Server's exposed data and/or controlling the Server's behaviour.

c. **Attribute Protocol (ATT) :** This layer allows BLE device to expose certain pieces of data or attributes.

d. **Security Manager :** This security Manager layer provides methods for device pairing and key distributions. It offers services to other protocol stack layers in order to securely connect and exchange data between BLE devices.

e. **Logical Link Control & Adaptation Protocol (L2CAP) :** This layer offers data encapsulation services to upper layers. This allows logical end to end data communication.

3. Application Layer :

- The BLE protocol stack layers interact with applications and profiles as desired. Application interoperability in the Bluetooth system is accomplished by Bluetooth profiles.
- The profile defines the vertical interactions between the layers as well as the peer-to-peer interactions of specific layers between devices.
- A profile composed of one or more services to address particular use case. A service consists of characteristics or references to other services.
- Any profiles/applications run on top of GAP/GATT layers of BLE protocol stack. It handles device discovery and connection related services for the BLE device.

3.6.3 BLE Topology

- Communication between BLE device and outside world is performed in two ways : broadcasting and connections.

1. **Broadcasting :** It consists of two devices i.e., broadcaster and observer. Broadcaster send data to all observer.

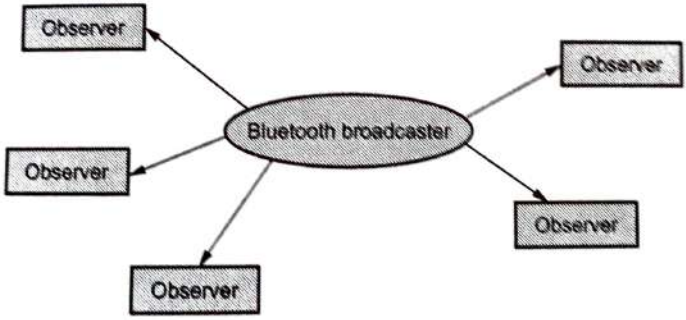


Fig. 3.6.3 : Broadcasting

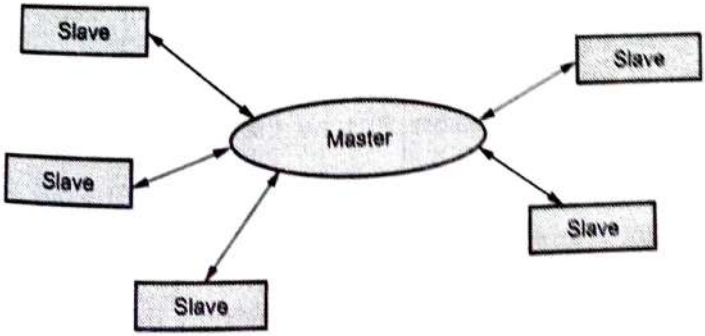


Fig. 3.6.4 : Connections

2. **Connections** : Connection is permanent and it send periodically data between two devices. It uses master-slave configuration. (See Fig. 3.6.4 on previous page.)

3.6.4 Comparison between Classic Bluetooth and Bluetooth Low Energy

Parameters	Classic Bluetooth	Bluetooth Low Energy
Distance/Range	100 m	50 m
Data Rate	1 - 3 Mbit/s	1 Mbit/s
Throughput	0.7 - 2.1 Mbit/s	0.27 Mbit/s
Number of Active Slaves	7	Not defined; implementation dependent
Security	56 / 128-bit and application layer user defined	128-bit AES with Counter Mode CBC-MAC and application layer user defined
Robustness	Adaptive fast frequency hopping, FEC, fast ACK	Adaptive frequency hopping, 24-bit CRC, Lazy Acknowledgement, 32-bit Message Integrity Check
Latency	100 ms	6 ms
Voice capable	Yes	No
Network topology	Scatternet	Star-bus
Peak current Consumption	Less than 30 mA	Less than 20 mA

3.6.5 Light Fidelity

- Light Fidelity (Li-Fi) is bidirectional wireless system that transmit data via LED or infrared light.
- Li-Fi potentially offers 10x the efficiency than traditional Wi-Fi, facilitating high-speed data communication of up to 1 Gbps.
- The technology has become an international standard for wireless communication in its first version in November 2011 by the International Telecommunications Standardisation Committee.
- The LEDs can be used like lasers in optical telecommunication in order to transfer data. LED light sources present in our surroundings can therefore be used for lighting but also used to transfer digital data.
- Li-Fi uses Light-Emitting Diodes (LEDs) bulbs, which flicker on and off at a very high rate not noticeable to the human eye, as a medium to deliver high speed

communication. Because LiFi uses light, it cannot penetrate walls and has limited working distance of typically a few meters.

- Fig. 3.6.5 shows working of Li-Fi.

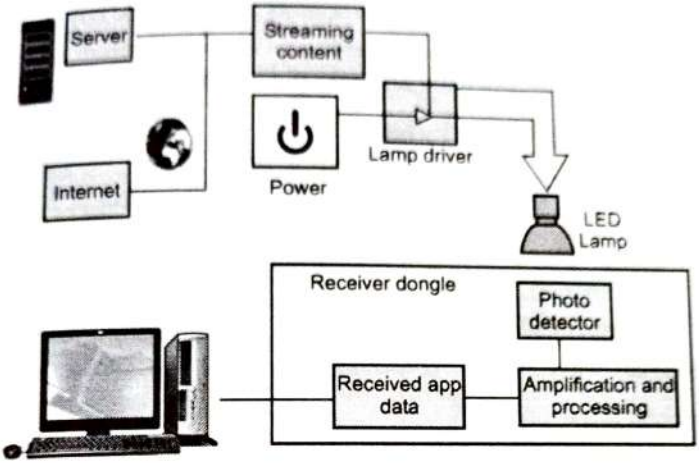


Fig. 3.6.5 : Working of Li-Fi

- Li-Fi is nothing but a Visible Light Communication (VLC) system which transmits data. It has two compsonents :
  a. At least one device with a photodiode which will be able to receive light signals and
  b. Light source equipped with a signal processing unit.
- Li-Fi technology requires LED a semiconductor light source, so that it can amplify light intensity and switch rapidly. Also, LED cells can modulate thousands of signals without the human eye ever noticing.
- The changes in light intensity from the LED light source are interpreted and converted as electrical current by the receiving photodiode device.
- Once the electronic signal is demodulated, it is converted into a continuous stream of binary data comprising of audio, video, web, and application information to be consumed by any internet-enabled device.
- Advantage
  1. No electromagnetic interference
  2. Fast and efficient
  3. Consumes less power
  4. Highly secure
  5. Low power consumption

6. Unlimited bandwidth

- Disadvantages
  1. Cannot penetrate through walls
  2. Initial cost is high
  3. Shorter range
  4. Interferences from external light sources like sun, light, normal bulbs etc

3.6.6 Difference between Li-Fi and Wi-Fi

Li-Fi	Wi-Fi
Li-Fi transmits data using light with the help of LED bulbs.	Wi-Fi transmits data using radio waves with the help of WiFi router.
Cannot penetrate through walls	Wi-Fi can penetrate through walls
It covers distance of about 10 meters	Wi-Fi has a range of 30 meters
Power consumption is less	Power consumption is more
Speed of 1 Gbps in use commercially	Speed up to 2 Gbps can be achieved commercially
It uses standard IEEE 802.15.7	It uses standard IEEE 802.11
Uses visible light of electromagnetic spectrum	Uses radio waves of electromagnetic spectrum
Works in high dense environment	Works in less dense environment due to interference related issues

3.7 Application Layer Protocols

- Internet of Things solutions employ some kind of messaging protocol for each individual IoT device to communicate in the system. These messaging protocols are used to transmit device telemetry or messages from the IoT devices to the IoT Messaging Hub.
- Messaging protocols are the rules, formats, and functions for messages sent between machines. Essentially, everyone has agreed on the types of information to include with data packets and the way of formatting that information so everyone can read it.

3.7.1 CoAP

- Constrained Application Protocol (CoAP) is a specialized web transfer protocol for use with constrained nodes and constrained (e.g., low-power, lossy) networks.

- CoAP is designed for simplicity, low overhead and multicast support in resource-constrained environments.
- CoAP is a web protocol that runs over the UDP for IoT. Datagram Transport Layer Security (DTLS) is used to protect CoAP transmission.
- The protocol is designed for machine-to-machine (M2M) applications such as smart energy and building automation.
- CoAP provides a request/response interaction model between application endpoints, supports built-in discovery of services and resources, and includes key concepts of the Web such as URIs and Internet media types.
- CoAP is designed to easily interface with HTTP for integration with the Web while meeting specialized requirements such as multicast support, very low overhead, and simplicity for constrained environments.
- CoAP is particularly targeted for small low power sensors, switches, valves and similar components that need to be controlled or supervised remotely, through standard Internet networks.
- CoAP is an application layer protocol that is intended for use in resource-constrained internet devices, such as Wireless Sensory Network (WSN) nodes.
- The key features of CoAP are :
  1. CoAP is a RESTful protocol.
  2. Four methods similar to HTTP : Get, Put, Post and Delete.
  3. Four different message types : Confirmable, Non-Confirmable, Acknowledgement and Reset (Nack).
  4. It support synchronous message exchange.
  5. Easy to proxy to and from HTTP.
  6. Constrained web protocol fulfilling M2M requirements.
  7. UDP binding with optional reliability supporting unicast and multicast requests. Confirmable and Acknowledgement / Reset messages to provide optional reliability when required. Low header overhead and reduced parsing complexity.
  8. Simple proxy and caching capabilities.
- A stateless HTTP mapping, allowing proxies to be built providing access to CoAP resources via HTTP in a uniform way or for HTTP simple interfaces to be realized alternatively over CoAP.

- CoAP provides a request/response interaction model between application endpoints, supports built-in discovery of services and resources, and includes key concepts of the Web such as URIs and Internet media types.
- CoAP is designed to easily interface with HTTP for integration with the Web, while meeting specialized requirements such as multicast support, very low overhead, and simplicity for constrained environments.
- CoAP is designed to interoperate with HTTP and the RESTful web through simple proxies, making it natively compatible to the Internet. CoAP is based on REST architecture, which is a general design for accessing Internet resources.
- Fig. 3.7.1 shows CoAP protocol stack.
- CoAP is based on the exchange of compact messages that, by default, are transmitted over UDP. Message of CoAP uses simple binary format.
- Message Layer supports 4 types message : CON (confirmable), NON (non-confirmable), ACK (Acknowledgement), RST (Reset)

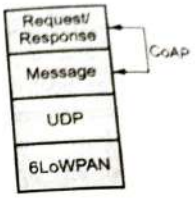


Fig. 3.7.1 CoAP protocol stack

- Reliable message transport : Keep retransmission until get ACK with the same message ID. Using default time out and decreasing counting time exponentially when transmitting CON. If recipient fail to process message, it responses by replacing ACK with RST.
- Unreliable message transport : transporting with NON type message. It doesn't need to be ACKed, but has to contain message ID for supervising in case of retransmission. If recipient fail to process message, server replies RST.
- Piggy-backed : Client sends request using CON type or NON type message and receives response ACK with confirmable message immediately. For successful response, ACK contain response message (identify by using token), for failure response, ACK contain failure response code.
- Separate response : If server receive a CON type message but not able to response this request immediately, it will send an empty ACK in case of client resend this message.
- When servers ready to response this request, it will send a new CON to client and client reply a confirmable message with acknowledgment. ACK is just to confirm CON message, no matter CON message carry request or response
- Fig. 3.7.2 shows CoAP message format.
- **Version (V) :** A 2-bit unsigned integer indicating the CoAP version number. Current version is 1. Other values are reserved for future versions.

1 byte			1 byte	2 bytes	TKL bytes	Variable	1 byte	Variable
V	T	TKL	Code	Message ID	Token (if any)	Options (if any)	0xFF (if payload)	Payload (if any)
2	2	4 bits						

Fig. 3.7.2 CoAP message format

- **Type (T)** : A 2-bit unsigned integer indicating if this message is of type Confirmable (0), Non-Confirmable (1), Achnowledgement (2) or Reset (3).
- **Token Length (TKL)** : A 4-bit unsigned integer indicating the length of the variable-length Token field (0-8 bytes). Lengths 9-15 are reserved.
- **Code** : An 8-bit unsigned integer indicating if the message carries a request (1-31) or a response (64-191), or is empty (0). (All other code values are reserved). In case of a request, the Code field indicates the Request Method (1 : GET; 2 : POST; 3 : PUT; 4 : DELETE); in case of a response a Response Code. Possible values are maintained in the CoAP Code Registry.
- **Message ID** : A 16-bit unsigned integer in network byte order used for the detection of message duplication and to match messages of type Acknowledgment/Reset to messages of type Confirmable/Non-confirmable.

Advantages :

- It runs over UDP and avoids overhead of TCP.
- It is easy to do HTTP - CoAP translation.
- It is a lightweight application layer protocol designed for constrained devices and constrained networks.

Disadvantages :

- Constraints associated with DTLS.
- No standardized framework for authorization and access control for CoAP exists as of now.
- No explicit support for real-time IoT application at present.

3.7.2 MQTT

- Message Queue Telemetry Transport (MQTT) is Open Connectivity for Mobile, M2M and IoT.
- MQTT is designed for high latency, low-bandwidth or unreliable networks. The design principle minimizes the network bandwidth and device resource requirements.
- MQTT is a lightweight broker-based publish/subscribe messaging protocol designed to be open, simple, lightweight and easy to implement.

MQTT characteristics

1. Lightweight message queueing and transport protocol
2. Asynchronous communication model with messages (events)
3. Low overhead (2 bytes header) for low network bandwidth applications
4. Publish / Subscribe (PubSub) model
5. Decoupling of data producer (publisher) and data consumer (subscriber) through topics (message queues)
6. Simple protocol, aimed at low complexity, low power and low footprint implementations
7. Runs on connection-oriented transport (TCP).
8. MQTT caters for (wireless) network disruptions

- The MQTT protocol works by exchanging a series of MQTT control packets in a defined way. Each control packet has a specific purpose and every bit in the packet is carefully crafted to reduce the data transmitted over the network.

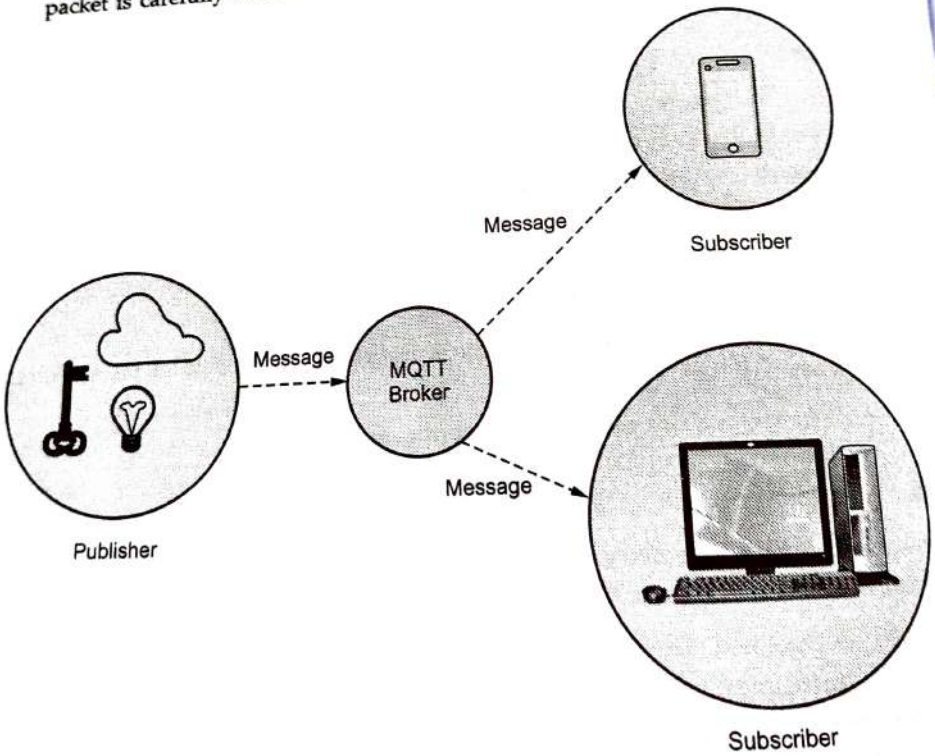


Fig. 3.7.3 MQTT publish/subscribe framework

- A MQTT topology has a MQTT server and a MQTT client. MQTT control packet headers are kept as small as possible.
- Having a small header overhead makes this protocol appropriate for IoT by lowering the amount of data transmitted over constrained networks.
- MQTT is the protocol built for M2M and IoT which is used to provide new and revolutionary performance
- Fig. 3.7.3 shows MQTT publish/subscribe framework.
- A producer publishes a message (publication) on a topic (subject). A consumer subscribes (makes a subscription) for messages on a topic (subject).
- A message server (called BROKER) matches publications to subscriptions.
- If none of them match the message is discarded after modifying the topic. If one or more matches the message is delivered to each matching consumer after modifying the topic
- Publish / Subscribe has three important characteristics :
  1. It decouples message senders and receivers, allowing for more flexible applications.
  2. It can take a single message and distribute it to many consumers.
  3. This collection of consumers can change over time, and vary based on the nature of the message.
- The MQTT messages are delivered asynchronously ("push") through publish subscribe architecture.
- The MQTT protocol works by exchanging a series of MQTT control packets in a defined way.
- Each control packet has a specific purpose and every bit in the packet is carefully crafted to reduce the data transmitted over the network.
- A MQTT topology has a MQTT server and a MQTT client. MQTT client and server communicate through different control packets. Table below briefly describes each of these control packets
- MQTT control packet headers are kept as small as possible. Each MQTT control packet consist of three parts, a fixed header, variable header and payload.
- Each MQTT control packet has a 2 byte Fixed header. Not all the control packet have the variable headers and payload.
- A variable header contains the packet identifier if used by the control packet. A payload up to 256 MB could be attached in the packets.
- Having a small header overhead makes this protocol appropriate for IoT by lowering the amount of data transmitted over constrained networks.

MQTT Quality of Service :

- There are the three levels of MQTT QoS.

1. **QoS 0 : AT MOST ONCE**
   - Guarantees that a particular message is only ever received by the subscriber a maximum of one time. This does mean that the message may never arrive.
   - The sender and the receiver will attempt to deliver the message, but if something fails and the message does not reach its destination the message may be lost.
   - This QoS has the least network traffic overhead and the least burden on the client and the broker and is often useful for telemetry data where it doesn't matter if some of the data is lost.
2. **QoS 1 : AT LEAST ONCE**
   - Guarantees that a message will reach its intended recipient one or more times. The sender will continue to send the message until it receives an acknowledgment from the recipient, confirming it has received the message.
   - The result of this QoS is that the recipient may receive the message multiple times, and also increases the network overhead than QoS 0.
   - In addition more burden is placed on the sender as it needs to store the message and retry should it fail to receive an ack in a reasonable time.
3. **QoS 2 : EXACTLY ONCE**
   - The most costly of the QoS, this QoS will ensure that the message is received by a recipient exactly one time.
   - This ensures that the receiver never gets any duplicate copies of the message and will eventually get it, but at the extra cost of network overhead and complexity required on the sender and receiver.

3.7.3 Difference between CoAP and MQTT

CoAP	MQTT
CoAP uses UDP protocol.	MQTT uses TCP protocol.
It uses Request/Response messaging.	It uses publish/subscribe messaging.
Communication model is One-to-one.	Communication model is Many-to-many.
Advantages : • Lightweight and fast • Low overhead • Support for multicasting	Advantages : • Simple management • Scalability • Robust communication
Weakness : Not as reliable as TCP based MQTT	Weakness : Higher overhead, no multicasting support
Security type is DTLS.	Security type is SSL/TLS.
Effectiveness in LLN is excellent.	Effectiveness in LLN is low.

3.7.4 Hypertext Transfer Protocol

- HTTP is an application layer protocol. The Web client and the Web server are application programs. Application layer programs do useful work like retrieving Web pages, sending and receiving email or transferring files. Lower layers take care of the communication details
- The client and server send messages and data without knowing anything about the communication network.
- HTTP is an asymmetric request-response client-server protocol. An HTTP client sends a request message to an HTTP server. The server, in turn, returns a response message. In other words, HTTP is a pull protocol; the client pulls information from the server.
- Fig. 3.7.4 shows HTTP request and response messages.

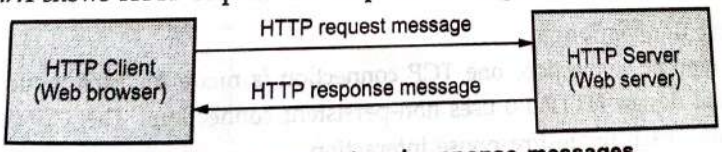


Fig. 3.7.4 HTTP request and response messages

- HTTP is core request-response protocol for web. It consists of four phases :
  1. Open connection : Based on URL
  2. Request : Client opens connection to server and sends request method, URL, HTTP version number, header information and terminated with blank line.

3. Response : Server processes request and sends HTTP protocol version and status code, header information, terminated by blank line and text (data).
4. Close connection

- All communication between clients and servers is based on HTTP. Servers listen on port 80. HTTP is a simple protocol; a client sends a request to a server and waits for a response.
- HTTP is stateless; it does not have any concept of open connection and does not require a server to maintain information on its clients.
- HTTP is based on TCP; whenever a client issues a request to a server, it first sets up a TCP connection and sends the message on that connection. The same connection is used for receiving the response. Fig. 3.7.5 shows working of HTTP server.

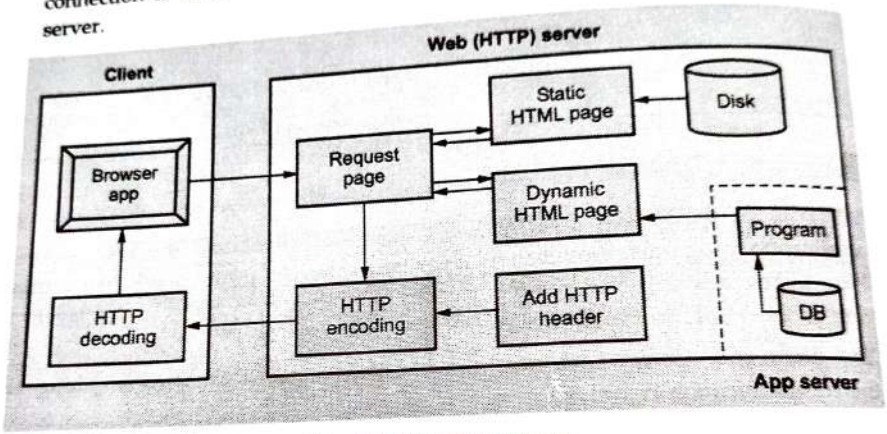


Fig. 3.7.5 HTTP working

- HTTP connections are of two types
  1. Non-Persistent HTTP
  2. Persistent HTTP

Non-Persistent Connection

- In this type of connection, one TCP connection is made for each request/response. The initial design HTTP 1.0 uses non-persistent connections. The TCP connection is closed after each request/response interaction.
- Each subsequent request from the same client to the same server involves the setting up and tearing down of an additional TCP connection. Fig. 3.7.6 shows non-persistent HTTP connection.

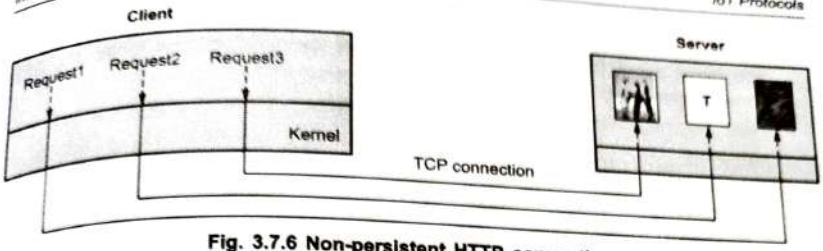


Fig. 3.7.6 Non-persistent HTTP connection

- Connection steps :
  1. Browser opens TCP connection to port 80 of server (handshake)
  2. Browser sends http request message
  3. Server receives request, locates object, sends response
  4. Server closes TCP connection
  5. Browser receives response, parses object
  6. Browser repeats steps 1-5 for each embedded object

Disadvantages of non-persistent

1. TCP processing and memory resource wasted in the server and the client.
2. It requires delay of 2 RTT associated with the transfer of each object.
3. Each TCP connection setup involves the exchange of three segments between client and server machines.

Persistent Connection

- HTTP 1.1 made persistent connections the default mode. The server now keeps the TCP connection open for a certain period of time after sending a response.

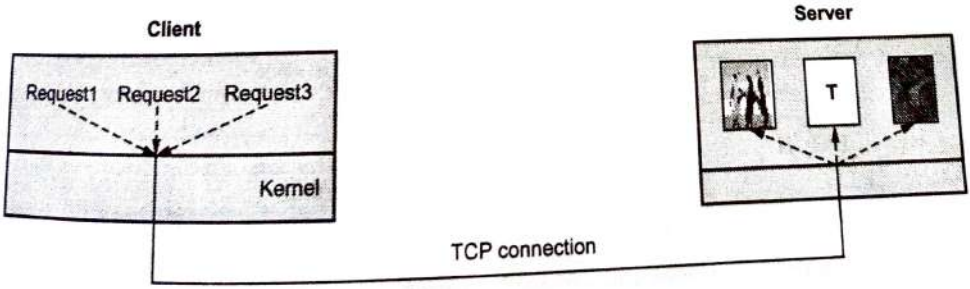


Fig. 3.7.7 Persistent HTTP connection

- This enables the client to make multiple requests over the same TCP connection and hence avoid the inefficiency and delay of the non-persistent mode. Fig. 3.7.7 shows persistent HTTP connection.
- There are two versions of persistent connections : Without pipelining and with pipelining.
- Steps for persistent connection :
  1. Browser opens TCP connection to port 80 of server (handshake).
  2. Browser sends http request message.
  3. Server receives request, locates object, sends response.
  4. Browser receives response, parses object.
  5. Browser repeats steps 2-4 for each embedded object.
  6. TCP connection closes on demand or timeout.

Advantages of Persistent Connection

1. CPU time saved in routers and hosts.
2. HTTP requests and responses can be pipelined on a connection.
3. Network congestion is reduced.
4. Latency on subsequent requests is reduced.

3.7.4.1 HTTP Methods

- Request message defines several kinds of messages referred to as methods.

Sr. No.	Method	Purposes
1.	GET	Used when the client wants to reterieve a document from the server. Server responds with the contents of the document.
2.	HEAD	Used when client wants some information about a document but not the document itself.
3.	POST	Used by the client to provide some information to the server i.e. input to the server.
4.	PUT	Used by the client to provide a new or replacement document to be stored on the server.
5.	PATCH	Similar to PUT except that the request contains a list of differences that should be implemented in the existing file.
6.	DELETE	Removes a document on the server.
7.	COPY	Copies a files to another location. URL gives the location of the source file.

8.	MOVE	Move a file to another location.
9.	LINK	Creates a link or links from a document to another location.
10.	UNLINK	UNLINK method deleted links created by the LINK method.
11.	OPTION	This method is used by the client to ask the server about available options.

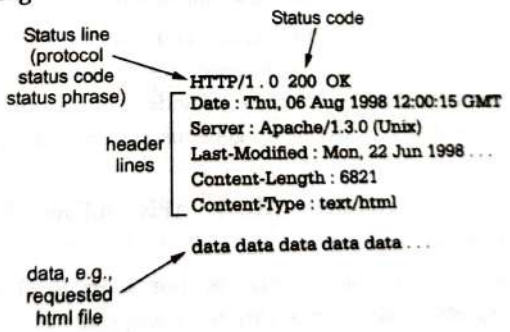
- The GET method requests the server to send the page suitably encoded in MIME. The HEAD method just asks for the message header, without the actual page.
- The PUT method is the reverse of GET : Instead of reading the page, it writes the page. This method makes it possible to build a collection of web pages on a remote server.
- The LINK and UNLINK methods allow connections to be established between existing pages or other resources.

HTTP Messages

HTTP messages are two types.

1. Request 2. Response

- Both message type use same format. Request message consists of a request line, headers and sometimes a body. An HTTP request must specify the host name (and possibly port) for which the request is intended.
- Response message format is shown below :



Status Code :

- The status code is a three-digit integer, and the first digit identifies the general category of response :
  a) **1xx** indicates an informational message.
  b) **2xx** indicates success of some kind.
  c) **3xx** redirects the client to another URL.
  d) **4xx** indicates an error on the client's part.

e) 5xx indicates an error on the server's part

3.7.4.2 Difference between Persistent and Non-persistent HTTP

Sr. No.	Persistent HTTP	Non-persistent HTTP
1.	Peristent version in 1.1	Non-persistent HTTP version is 1.0.
2.	IT uses on RTT.	It uses two RTT.
3.	TCP connection is not closed	TCP connection is closed after every request-response.
4.	Client make multiple request over the same TCP connection.	Client make multiple request over the multiple TCP connection.
5.	It is default mode.	It is not default mode.
6.	Request methods are GET, HEAD, POST, DELETE, TRACE and OPTIONS.	Request methods used are GET, POST and HEAD.

3.7.5 Systematic HTTP Access Methodology : REST API

- A RESTful API is an architectural style for an Application Program Interface (API) that uses HTTP requests to access and use data. That data can be used to GET, PUT, POST and DELETE data types, which refers to the reading, updating, creating and deleting of operations concerning resources.
- A large part of the interoperability, scale, and control for IoT can be achieved through API management. Standards-based design patterns for Web APIs, API management, and a RESTful architecture provide tremendous value in simplifying the task of interoperability across heterogeneous systems handling vast amounts of data.
- Representational State Transfer (REST) APIs follow the request-response communication model.
- REST is a set of architectural constraints, not a protocol or a standard. API developers can implement REST in a variety of ways.
- REST requires that a client make a request to the server in order to retrieve or modify data on the server. A request generally consists of :
  a) An HTTP verb, which defines what kind of operation to perform
  b) A header, which allows the client to pass along information about the request
  c) A path to a resource
  d) An optional message body containing data

- Important REST principles :
  a) Stateless : No client context stored on the server, each request is complete.
  b) Cacheable : Responses explicitly indicate their cacheability.
  c) Layered System : Client cannot tell if connected directly to the server (e.g. reverse proxies).
  d) URIs : Resources are identified using Uniform Resource Identifiers (URIs)
- When a client request is made via a RESTful API, it transfers a representation of the state of the resource to the requester or endpoint. This information, or representation, is delivered in one of several formats via HTTP : JSON, HTML, XLT, Python, PHP, or plain text.
- JSON is the most generally popular programming language to use because, despite its name, it's language-agnostic, as well as readable by both humans and machines.
- Something else to keep in mind : Headers and parameters are also important in the HTTP methods of a RESTful API HTTP request, as they contain important identifier information as to the request's metadata, authorization, Uniform Resource Identifier (URI), caching, cookies, and more. There are request headers and response headers, each with their own HTTP connection information and status codes.
- Applications conforming to the REST constraints can be called RESTful. RESTful systems typically communicate over HTTP with the same methods (GET, POST, PUT, DELETE etc) that browsers use to retrieve web pages and to send data to remote servers.

1. **Client-Server :** requires that a service offer one or more operations and that services wait for clients to request these operations.
2. **Stateless :** requires communication between service consumer (client) and service provider (server) to be stateless.
3. **Cache :** requires responses to be clearly labeled as cacheable or non-cacheable.
4. **Interface :** requires all service providers and consumers within a REST-compliant architecture to share a single common interface for all operations.
5. **Layered System :** requires the ability to add or remove intermediaries at runtime without disrupting the system.
6. **Code-on-Demand :** allows logic within clients (such as Web browsers) to be updated independently from server-side logic using executable code shipped from service providers to consumers.

- Each client request and server response is a message, and REST-compliant applications expect each message to be self-descriptive. That means each message must contain all the information necessary to complete the task. Other ways to describe this kind of message are "stateless" or "context-free." Each message passed between client and server can have a body and metadata.

3.7.6 Web Socket

- WebSocket support full-duplex, two-way communication between client and server.
- WebSocket APIs reduce the network traffic and latency as there is no overhead for connection setup and termination requests for each message.
- WebSocket uses a standard HTTP request-response sequence to establish a connection. When the connection is established, the WebSocket API provides a read and write interface for reading and writing data over the established connection in an asynchronous full duplex manner.
- WebSocket also provides an interface for asynchronously closing the connection from either side.
- **Features :**
  1. Small header size.
  2. WebSocket enables bi-directional communication over a single TCP connection.
  3. The WebSocket protocol uses port 80 for regular WebSocket connections and port 443 for WebSocket connections tunneled over Transport Layer Security (TLS).
  4. Protocol specifies six frame types and leaves ten reserved for future use.
  5. Client and server exchange the message after a successful handshake.
  6. It is an independent TCP-based protocol.
- Fig. 3.7.8 shows web socket communication. (Refer Fig. 3.7.8 on next page)
- The protocol works in the following sequence :
  1. The client sends an HTTP upgrade request to the server through Access Gateway to establish a communication channel between the client and the server. (WebSocket protocol handshake)
  2. The server sends an HTTP 101 response to the requesting client through Access Gateway. When the client receives the response, the HTTP connection is upgraded to WebSocket.
  3. Bidirectional data exchange happens between the server and the client over the WebSocket connection.

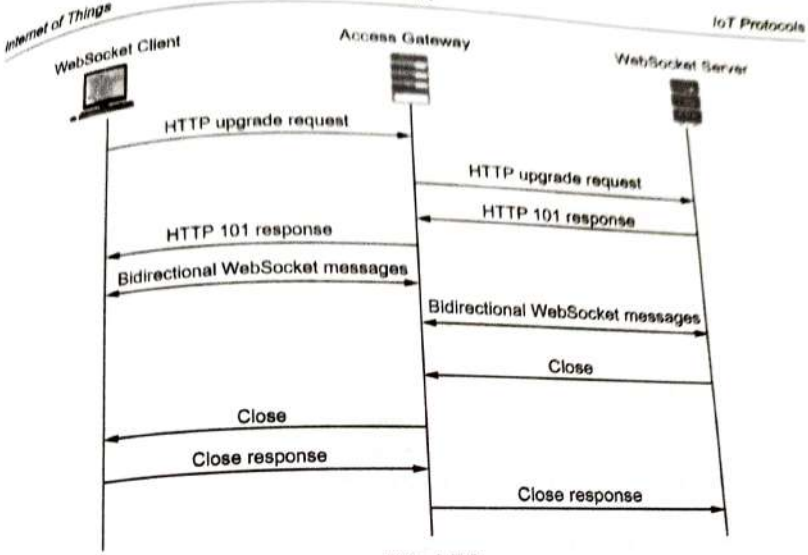


Fig. 3.7.8

4. Either of the participant in the data exchange requests to terminate the WebSocket connection. One participant sends a Close request to the other participant and the connection is terminated.

3.7.7 XMPP

- Extensible Messaging and Presence Protocol (XMPP) is an open XML technology for real-time communication. It is based on instant messaging and presence.
- XMPP is an open-source popular language which uses the markups. The markup means marking by some signs and characters or tags so specify the contents between the markups.
- It allows the exchange of data between two or more systems and supports presence and contact list maintenance.
- It uses client server architecture in which XMPP client communicates with XMPP server using TCP socket. It also works via HTTP using a websocket implementation.
- It also uses publish/subscribe mechanism for data sharing like MQTT protocol.
- XMPP is based on a decentralized client-server architecture. In this architecture, clients don't communicate directly with each other; instead, there's a decentralized server acting as the intermediary between them.

- XMPP allocates an XMPP address to every client on the XMPP network. This address works just like a standard email address with an IP address/domain name, an optional node, and a username for the resident server.
- Fig. 3.7.9 shows simple architecture of XMPP.

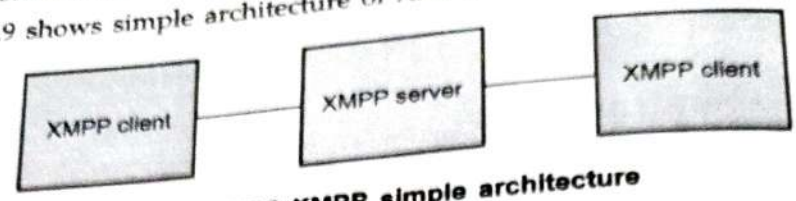


Fig. 3.7.9 XMPP simple architecture

- In a simple XMPP architecture consisting of a server and two clients, a client with a unique name communicates through an associated XMPP server with another client using a unique name.
- Each client on the XMPP network implements the client form of the protocol with the XMPP server providing routing capability. The architecture may include XMPP gateways which are often used to translate between foreign messaging domains and IM protocols.
- The XMPP gateways permit the termination of a given client-to-server session as well as the initiation of a new client-to-server session to the target endpoint protocol, along with the necessary protocol translation.
- XMPP uses the Transmission Control Protocol as its original and "native" transport protocol for web applications and firewalls.
- XMPP protocol is used for all the following applications.
  a) Instant messaging apps (Google Talk, WhatsApp)
  b) Presence status
  c) Message delivery
  d) Conferencing (Multi-party chat)
  e) Roster management
  f) Voice and video calls
  g) Online gaming
  h) News websites
  i) VoIP apps
- Advantages of XMPP protocol
  1. Supports HTTP transport protocol.
  2. It offers persistent connection.
  3. It is decentralized in nature as no central XMPP servers are needed.

4. It allows servers with different architectures to communicate
5. Utilizes a decentralized client-server architecture.
6. It uses TLS and SASL to provide secured end to end connection.

- Disadvantages of XMPP protocol
  1. It does not have QoS mechanism as used by MQTT protocol.
  2. Streaming XML has overhead due to text-based communication compare to binary based communication.
  3. XML content transports asynchronously.
  4. Server may overload with presence and instant messaging.

3.7.8 DDS

- The Object Management Group (OMG) Data Distribution Service for Real-Time Systems is a standard for data-centric Publish/Subscribe system.
- DDS is an Object Management Group (OMG) standard that defines a system, Application Programming Interface (API) and wire protocol for type-safe network communications.
- It is an IoT protocol developed for M2M Communication by OMG (Object Management Group).
- It enables data exchange via publish-subscribe methodology.
- DDS makes use of brokerless architecture unlike MQTT and CoAP protocols.
- It uses multicasting to bring high quality QoS to the applications.
- DDS protocol can be deployed from low footprint devices to cloud.
- The built-in authentication plug-in uses Public Key Infrastructure (PKI) with a trusted identity certificate authority. Each DDS domain participant is certified by the certificate authority.
- Fig. 3.7.10 shows components in the OMG DDS standards.
- The key abstraction at the foundation of DDS is a fully distributed Global Data Space (GDS). The DDS specification requires a fully distributed implementation of the GDS to avoid single points of failure or single points of contention. Publishers and subscribers can join or leave the GDS at any point in time as they are dynamically discovered.
- It consists of the DDS v1.2 API and the Data Distribution Service Interoperability Wire Protocol (DDSI). DDS V1.2 API standard is language independent, OS and HW architecture independent.

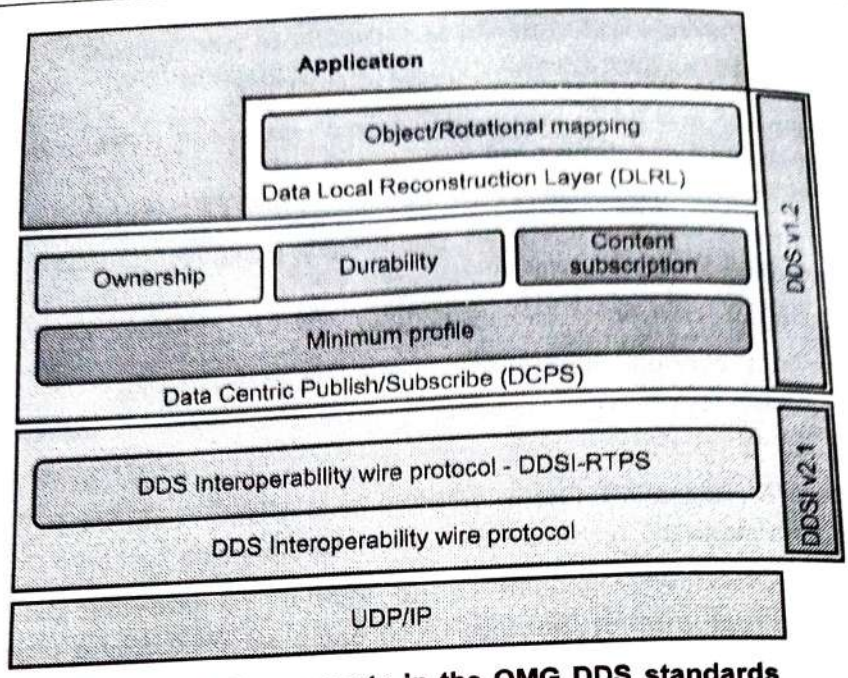


Fig. 3.7.10 Components in the OMG DDS standards

- The DDS API standard ensures source code portability across different vendor implementations, while the DDSI Standard ensures on the wire interoperability across DDS implementations from different vendors.
- The DDS API standard shown in figure also defines several profiles that enhance real time P/S with content filtering, persistence, automatic fail-over, and transparent integration into object oriented languages.
- Data Centric Publish Subscribe (DCPS) layer delivers information to subscribers. DCPS is a standard API for data centric, topic based, real time publish/subscribe layer.
- DLRL Data Local Reconstruction Layer (DLRL) provides interface to DCPS functionalities. This enables sharing of distributed data among devices which are IoT enabled. DLRL is standard API for creating object views out of collection of topics.
- The DDS standard was formally adopted by the OMG in 2004. It quickly became the established P/S technology for distributing high volumes of data dependably and with predictable low latencies in applications such as radar processors, flying and land drones, combat management systems, air traffic control and management, high performance telemetry, supervisory control and data acquisition systems, and automated stocks and options trading.

- Along with wide commercial adoption, the DDS standard has been mandated as the technology for real-time data distribution by organization worldwide, including the US Navy, the Department of Defence (DoD), Information Technology Standards Registry the UK Mistry of Defence (MoD), the Military Vehicle Association (MILVA), and EUROCAE - the European organization that regulates standards in Air Traffic Control and Management.

3.7.9 AMQP

- Advanced Message Queuing Protocol (AMQP) is a software layer protocol for message-oriented middleware environments. AMQP is an open protocol for asynchronous message queuing.
- AMQP is an open standard, binary application layer protocol designed for message-oriented middleware i.e., AMQP protocol standardizes messaging using Producers, Brokers and Consumers and messaging increases loose coupling and scalability.
- A protocol to communicate between clients and messaging middleware servers (brokers). The Broker is the AMQP Server.
- AMQP supports both publish-subscribe model and point-to-point communication, routing and queuing.
- AMQP divides the brokering task between exchanges and message queues, where the first is a router that accepts incoming messages and decides which queues to route the messages to, and the message queue stores messages and sends them to message consumers.
- AMQP supports username and password authentication as well as SASL authorization. It also supports TLS encryption. Fig. 3.7.11 shows AMQP architecture.

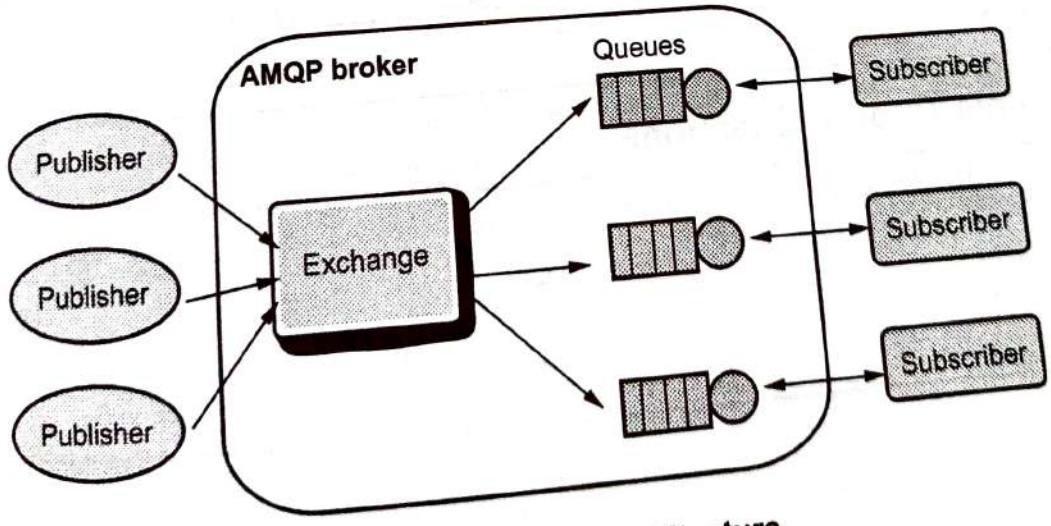


Fig. 3.7.11 AMQP architecture

- Exchange : Receives messages from publisher primarily based programs and routes them to message queues.
- Message Queue : Stores messages until they may thoroughly process via the eating client software.
- Binding : States the connection between the message queue and the change.

3.8 Fill in the Blanks

Q.1 10BASET is _____ MHz Ethernet running over UTP cable.

Q.2 AMQP is an open protocol for _____ message queuing.

Q.3 XMPP is based on a _____ client-server architecture.

Q.4 XMPP uses the _____ as its original and "native" transport protocol for web applications and firewalls.

Q.5 WebSocket support full-duplex, _____ communication between client and server.

Q.6 Representational State Transfer APIs follow the request-response _____ model.

Q.7 _____ is a 'request-reply' protocol.

Q.8 RESTful web service is a web API implemented using _____ and _____ principles.

Q.9 IEEE 802.15.4 is a collection of standards for _____ wireless personal area networks.

Q.10 CoAP stands for _____.

Q.11 MQTT is a light weight messaging protocol based on the _____ model.

Q.12 An IP address is made up of _____ bits of information.

Q.13 Packets in the IPv4 layer are called _____.

Q.14 CoAP is a web protocol that runs over the _____ for IoT.

Q.15 XMPP uses _____.

3.9 Multiple Choice Questions

Q.1 IPv6 addresses are _____ bits in length.

a 32 b 64

c 128 d 255

Q.2 XMPP stands for _____.

a Extensible Messaging and Presence Protocol
b Extensible Mail and Presence Protocol
c Extensible Messaging and Packet Protocol
d Extensible Messaging and Publish Protocol

Q.3 _____ is a software layer protocol for message-oriented middleware environments.

a XMPP
b AMQP
c CoAP
d TCP

Q.4 AMQP stands for _____.

a Advanced Message Queuing Packet
b Advanced Mail Queuing Protocol
c Application Message Queuing Protocol
d Advanced Message Queuing Protocol

Q.5 WebSocket APIs allows bi-directional, _____ communication between clients and servers.

a simplex
b half duplex
c full duplex
d all of these

Q.6 In the context of IoT, _____ allows real time communication between IoT devices.

a TCP
b UDP
c XMPP
d MQTT

Q.7 The _____ layer protocols provide end-to-end message transfer capability independent of the underlying network.

a Application
b Network
c Internet
d Transport

Q.8 _____ is a lightweight broker-based publish/subscribe messaging protocol designed to be open, simple, lightweight and easy to implement.

a XMPP
b AMQP
c MQTT
d DDS

Q.9 _____ is a web protocol that runs over the UDP for IoT.

a UPnP
b CoAP
c MQTT
d HTTP

Q.10 CoAP is based on _____ model.

a client-server
b push-pull
c REST API
d None

Q.11 BLE stands for _____.

a Bluetooth Low Energy
b Bluetooth Layer Energy
c Bluetooth Low Encryption
d Bluetooth Light Energy

Q.12 MQTT follows the _____ pattern.

a push-pull
b client server
c socket
d publish/subscribe

Q.13 Li-Fi uses _____ as a medium to transfer or transport the data.

a light
b copper
c signal
d IR

Q.14 An Internet Protocol (IP) address has a fixed length of _____.

a 4 bits
b 8 bits
c 16 bits
d 32 bits

Q.15 _____ addresses are used for multicast services that allow a host to send information to a group of hosts simultaneously.

a Class A
b Class B
c Class C
d Class D

Q.16 The header length of an IPv6 datagram is _____.

a. 10 bytes
b. 25 bytes
c. 30 bytes
d. 40 bytes

Q.17 Subnet mask 255.0.0.0 belongs to _____.

a. Class A
b. Class B
c. Class C
d. Class D

Q.18 IPv6 addresses are _____ bits in length.

a. 32
b. 64
c. 128
d. 256

Q.19 IPv6 does not use _____ type of address.

a. broadcast
b. multicast
c. anycast
d. unicast

Q.20 What is subnet mask of class B ?

a. 255.0.0.0
b. 255.255.0.0
c. 255.255.255.0
d. 255.255.255.255

Q.21 BLE uses _____ ISM frequency band either in dual mode or single mode.

a. 1 GHz
b. 1.4 GHz
c. 2.4 GHz
d. 2.4 Hz

Q.22 A packet sent to _____ address is delivered to all interfaces identified by that address.

a. unicast
b. anycast
c. broadcast
d. multicast

Answer Keys for Fill in the Blanks

Q.1	10	Q.2	asynchronous	Q.3	decentralized
Q.4	TCP	Q.5	two-way	Q.6	communication
Q.7	HTTP	Q.8	HTTP, REST	Q.9	low rate

Q.10	Constrained Application Protocol	Q.11	publish / subscribe	Q.12	32
Q.13	datagrams	Q.14	UDP	Q.15	XML

Answer Keys for Multiple Choice Questions

Q.1	c	Q.2	a	Q.3	b	Q.4	d
Q.5	c	Q.6	c	Q.7	d	Q.8	c
Q.9	b	Q.10	c	Q.11	a	Q.12	d
Q.13	a	Q.14	d	Q.15	d	Q.16	d
Q.17	a	Q.18	c	Q.19	a	Q.20	b
Q.21	c	Q.22	d				

□□□

4

Syllabus

IOT Security,
Update Man

Content

4.1 IoT S
4.2 Assi
4.3 Sec
4.4 Ke
4.5 Up
4.6 C
4.7 F
4.8

4 IoT Security and Challenges

Syllabus

IOT Security, Dangers, Assigning values to Information, Security Components, Key Management, Update Management, Challenges in IoT security.

Contents

4.1 IoT Security Issues and Need

4.2 Assigning Values to Information

4.3 Security Components

4.4 Key Management

4.5 Update Management

4.6 Challenges in IoT Security

4.7 Fill in the Blanks

4.8 Multiple Choice Questions

4.1 IoT Security Issues and Need

- The Internet of Things (IoT) refers to a concept of connected objects and devices of all types over the Internet wired or wireless. The popularity of IoT or the Internet of Things has increased rapidly, as these technologies are used for various purposes, including communication, transportation, education, and business development.
- The unconscious use, not changing passwords and the lack of device updates have increased cybersecurity risks and access to malicious applications to the IoT systems sensitive data.
- Most of the security professionals consider IoT as the vulnerable point for cyber-attacks due to weak security protocols and policies. Even though several security mechanisms were developed to protect IoT devices from cyber-attacks, security guidelines are not appropriately documented.
- IoT enabled devices have been used in industrial applications and for multiple business purposes. The apps help these businesses to attain a competitive edge over their competitors.
- However, due to the excessive adoption of various smart devices with data sharing and integration, the privacy and data breach becomes a significant concern to most businesses, as it interrupts the flow of work, activities, and network services.
- IoT system functionalities :
  1. Security patch must be upload time to time in microprocessor firmware.
  2. Monitor the access and usage of public network.
  3. User authentication is necessary.
  4. Only after authentication can the controller direct commands for things control that are present in the system.
- The Internet of Things (IoT) has become a ubiquitous term to describe the tens of billions of devices that have sensing or actuation capabilities and are connected to each other via the Internet.

Risks :

- The IoT includes everything from wearable fitness bands and smart home appliances to factory control devices, medical devices and even automobiles. Security has not been a high priority for these devices until now.
- The security of the Internet of Things, the following principles can be established.
  a) **Identity** : Trust is always tied to an identity. Therefore every device needs a unique identity that can't be changed. The device must also be able to prove its identity at all times.

b) **Positive intention** : The device and linked service have positive intentions.

c) **Predictability and transparency** : The functional scope of the service provided by devices is known to its full extent. There are no undocumented (secret) functions. The behaviour of the system can be checked at any time by independent third parties.

d) **Reputation** : An increasing number of positive interactions between the things gradually form a reputation based intelligent network.

4.1.1 Security Architecture

- Fig. 4.1.1 shows IoT security architecture. (See Fig. 4.1.1 on next page.)
- IoT systems are often highly complex, requiring end-to-end security solutions that span cloud and connectivity layers, and support resource-constrained IoT devices that often aren't powerful enough to support traditional security solutions.
- Application layer support user services. This layer helps users access IoT through the interface using PC, mobile equipment etc. This layer also support secure communication protocol and authentication protocols.
- Network layer support wired and wireless communication protocol and technology. This layer is responsible for dependable broadcast of data and information from the below layer.
- Sensors are the monitors that pick up data and relay it for further analysis. Actuators are devices that act as robotic controls. Many IoT attacks have used actuators, such as printers, as launch points into a business's network.
- An IoT security architecture is a blueprint that illustrates all components of the IoT infrastructure for all IoT projects and details how to secure each component.
- In both cases, it is imperative to ensure device access is controlled via settable passwords, encrypt any data stored locally and monitor and contain any executable code run by the device.
- Physical layer gathers all types of information with the help of physical equipment. IoT devices face many threats, including malicious data that can be sent over authenticated connections, exploiting vulnerabilities and/or misconfigurations.
- Such attacks frequently exploit many weaknesses, including but not limited to
  a) Failure to use code signature verification and secure boot,
  b) Poorly implemented verification models which can be bypassed.

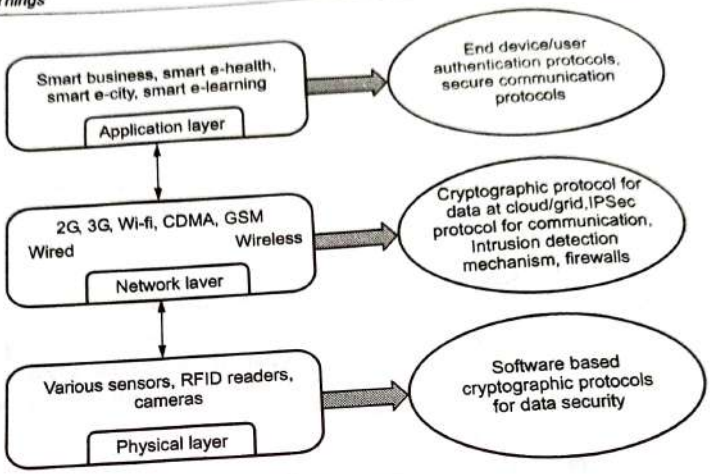


Fig. 4.1.1

- Attackers often use those weaknesses to install backdoors, sniffers, data collection software, file transfer capabilities to extract sensitive information from the system, and sometimes even Command & Control (C & C) infrastructure to manipulate system behaviour.

4.1.2 Security Requirement

- The key requirements for any IoT security solution are :
  1. Device and data security, including authentication of devices and confidentiality and integrity of data.
  2. Implementing and running security operations at IoT scale.
  3. Meeting compliance requirements and requests.
  4. Meeting performance requirements as per the use case.
- Application layer : Verification and user's confidentiality
- Support layer : Various encryption algorithms
- Network layer : Distributed denial of service attack
- Physical layer : Authentication.

4.2 Assigning Values to Information

- An information asset can be described as information or data that is of value to the organization, including such information as patient records, intellectual property, or customer information. These assets can exist in physical form (on paper) or electronically (stored on databases, in files, on PC).
- An information asset container is where information assets are stored, transported, or processed.
- Identification, valuation and categorization of information systems assets are critical tasks of the process to properly develop and deploy the required security control for the specified IT assets.
- Quantitative measurement of risk impact is implemented based on the following formula :

  Risk impact = Potential risk × Probability of occurrence
- Potential risk is a product of total asset value, severity of vulnerability and severity of threat :

  Potential risk = Total asset value × Severity of vulnerability × Severity of threat
- Asset valuation : This is a method of assessing the worth of the organization's information system assets based on its Confidentiality, Integrity and Availability (CIA) security.

  Total asset value = Asset value × Weight of asset
- Assumptions for asset valuation include :
  a) The value of an asset depends on the sensitivity of data inside the container and their potential impact on CIA.
  b) CIA of information will have a minimum value of 1 for each.
  c) The value of levels for CIA are as follows: A rating of 3 is high, 2 is medium and 1 is low.
  d) The value of the information asset is determined by the sum of the three (C + I + A) attributes.

Weight of Asset

- The actual value of an asset is determined by the sensitivity value of data in the container. The reason is that all similar containers are not equally important to the organization, and the value of a container is determined by the data it holds, processes or transfers.
- For example, servers with equal capacity, technology and cost may have different weights due to the data they hold, process or transfer.

- A database containing employee information may have less value than one containing customer transactions. Equally, data on prominent customers may have more value than data on ordinary/walk-in customers, based on business/organizational objectives.

4.2.1 Risk Assessment

- Risk assessment basically involves the calculation of the magnitude of potential consequences (levels of impacts) and the likelihood (levels of probability) of these consequences to occur. Essentially, the higher the probability of a "worse" effect occurring, the greater the level of risk.

  Risk = Threats × Vulnerabilities

- This is a common formula that is used to determine the likelihood of risk. It's a good way to approach finding risk because it addresses the key factors in a cyber security threat.

  Risk = (Likelihood of occurrence of vulnerability) × (Value of the information asset) – (% of risk mitigated by current controls) + (Uncertainty of current knowledge of vulnerability).

- Fig. 4.2.1 shows risk assessment steps.

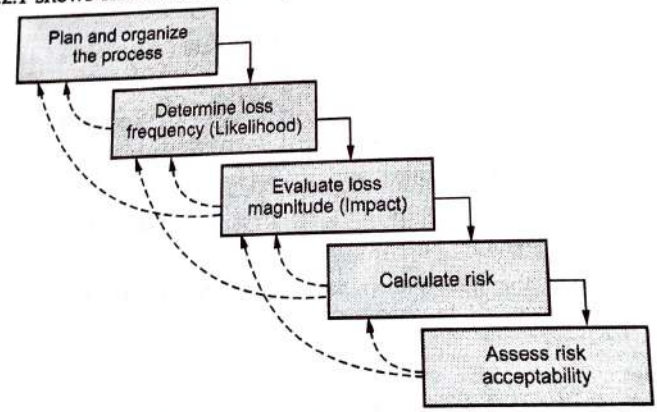


Fig. 4.2.1 Risk assessment steps

Step 1 : Identify the hazards

Step 2 : Decide who might be harmed and how

Step 3 : Evaluate the risks and decide on precautions

Step 4 : Record your findings and implement them

Step 5 : Review your assessment and update if necessary

- The objective of risk assessment is to determine the level of risk exposure on the information asset by analyzing the vulnerability factor (extent of impact), threat likelihood (likelihood of exploitation) and the risk occurrence rate. There are two sub-processes for conducting the risk assessment - Risk analysis and Risk evaluation.
- Risk analysis can be undertaken either qualitatively or quantitatively or semi-quantitatively. In this procedure, qualitative information security risk assessment method has been adopted.
- Risk evaluation : Risk evaluation uses the understanding of risk obtained by risk analysis to make decisions about future action
- Risk assessment is a term used to describe the overall process or method where you:
  1. Identify hazards and risk factors that have the potential to cause harm (hazard identification).
  2. Analyze and evaluate the risk associated with that hazard (risk analysis, and risk evaluation).
  3. Determine appropriate ways to eliminate the hazard, or control the risk when the hazard cannot be eliminated (risk control).
- A risk assessment is a thorough look at your workplace to identify those things, situations, processes, etc. that may cause harm, particularly to people. After identification is made, you analyze and evaluate how likely and severe the risk is. When this determination is made, you can next, decide what measures should be in place to effectively eliminate or control the harm from happening.

Identify Possible Controls

- **Residual risk** is the risk to the information asset that remains even after the application of controls. Either there are no control measures that could prevent it, or they would be disproportionate to the level or risk presented.
- General categories of controls are policies, programs, and technologies.
- Policies are documents that specify an organization's approach to security. Various types of security policies are general security, program security, issue-specific and systems-specific policies.
  1. General security policy : These policies are an executive-level document that outlines the organization's approach and attitude toward information security and relates the strategic value of information security within the organization.
  2. Program security policy : This policy contains planning documents. It is an outline the process of implementing security in the organization. This policy act as a blueprint for the analysis, design, and implementation of security.

3. Issue-specific policies : This policy gives idea about specific implementations of policy. These policies are typically developed to provide detailed instructions and restrictions associated with security issues. Examples include policies for Internet use, e-mail, and access to the building.
4. Systems-specific policies address the particular use of certain systems

4.3 Security Components

- How to protect the valuable assets ? It is necessary to keep in safe place like a bank to protect the valuable assets. But bank is not a safe place now a day. There are so many examples where bank robbery in our country.
- Bank robbery is the crime of stealing from a bank during opening hours. Protecting assets was difficult and not always effective.
- Now a day, protection is easier because many factors working against the potential criminal. Very sophisticated alarm and camera systems silently protect secure places like banks.
- Traditionally information security provided by physical i.e. rugged filing cabinets with locks and administrative mechanisms i.e. personnel screening procedures during hiring process.
- Asset protection systems are designed to recover stolen cash and high value assets, apprehend criminals and deter crime. The system has the capacity to track, protect and manage critical assets in real-time.
- The techniques of criminal investigation have become so effective that a person can be identified by genetic material, voice, retinal pattern, fingerprints etc.
- Use of networks and communications links requires measures to protect data during transmission.
- Data security is the science and study of methods of protecting data from unauthorized disclosure and modification.
- Data and information security is about enabling collaboration while managing risk with an approach that balances availability versus the confidentiality of data.
- Computer security : Generic name for the collection of tools designed to protect data and to hackers.
- Network security : Measures to protect data during their transmission.
- Internet security : Measures to protect data during their transmission over a collection of interconnected networks.
- Physical security : To protect physical items, objects, or areas from unauthorized access and misuse

- Personnel security : To protect the individual or group of individuals who are authorized to access the organization and its operations
- Operations security : To protect the details of a particular operation or series of activities
- The Committee on National Security Systems (CNSS) defines information security as the protection of information and its critical elements, including the systems and hardware that use, store, and transmit that information.
- Fig. 4.3.1 shows components of information security.

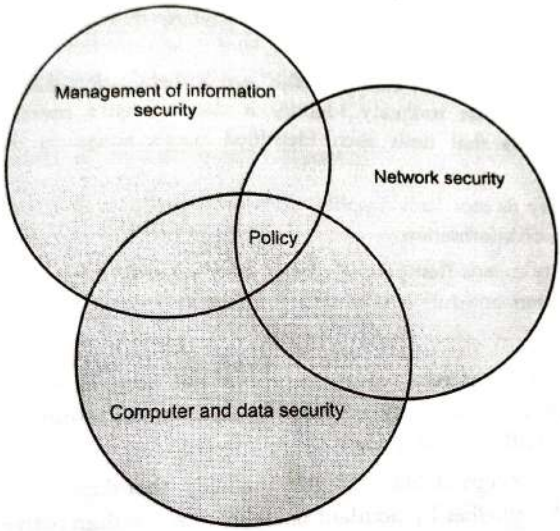


Fig. 4.3.1 Components of information security

- The CNSS model of information security evolved from a concept developed by the computer security industry called the C.I.A triangle.
- The CIA triad is a widely used information security model that can guide an organization's efforts and policies aimed at keeping its data secure. The model has nothing to do with the U.S. Central Intelligence Agency; rather, the initials stand for the three principles on which infosec rests :
- Confidentiality : Only authorized users and processes should be able to access or modify data
- Integrity : Data should be maintained in a correct state and nobody should be able to improperly modify it, either accidentally or maliciously

- Availability : Authorized users should be able to access data whenever they need to do so.

Security Goals :

- **Security goals are as follows :** 1. Confidentially 2. Integrity 3. Availability

1. Confidentiality

- Confidentiality ensures that no one can read the message except intended receiver.
- Confidentiality refers to limiting information access and disclosure to authorized users and preventing access by or disclosure to unauthorized ones.
- Sensitive information should be kept secret from individuals who are not authorized to see the information.
- Underpinning the goal of confidentiality are authentication methods like user-IDs and passwords that uniquely identify a data system's users, and supporting control methods that limit each identified user's access to the data system's resources.
- Confidentiality is not only applied to storage of data but also applies to the transmission of information.
- Confidentiality means that people cannot read sensitive information, either while it is on a computer or while it is traveling across a network.

2. Integrity

- Integrity ensures that received message has not been altered in any way from origin. It refers to the trustworthiness of information resources. Integrity should not be altered without detection.
- It includes the concept of "data integrity" namely, that data have not been changed inappropriately, whether by accident or deliberately malign activity.
- It also includes "origin" or "source integrity" that is, that the data actually came from the person or entity you think it did, rather than an imposter.
- Integrity ensures that information is not changed or altered in transit. Under certain attack models, an adversary may not have to power to impersonate an authenticated party or understand a confidential communication, but may have the ability to change the information being transmitted.
- On a more restrictive view, however, integrity of an information system includes only preservation without corruption of whatever was transmitted or entered into the system, right or wrong.

3. Availability

- Availability refers, to the availability of information resources. An information system that is not available when you need it is at least as bad as none at all.
- Availability means that people who are authorized to use information are not prevented from doing so. It may be much worse, depending on how reliant the organization has become on a functioning computer and communications infrastructure.
- Almost all modern organizations are highly dependent on functioning information systems. Many literally could not operate without them.
- Availability, like other aspects of security, may be affected by purely technical issues (e.g. a malfunctioning part of a computer or communications device), natural phenomena (e.g. wind or water), or human causes (accidental or deliberate).

4.3.1 Components of an Information System

- An Information System can be defined technically as a set of interrelated components that collect (or retrieve), process, store and distribute information to support decision making and control in an organization
- An information system is essentially made up of five components hardware, software, database, network and people. These five components integrate to perform input, process, output, feedback and control.
- Fig. 4.3.2 shows components of information system.

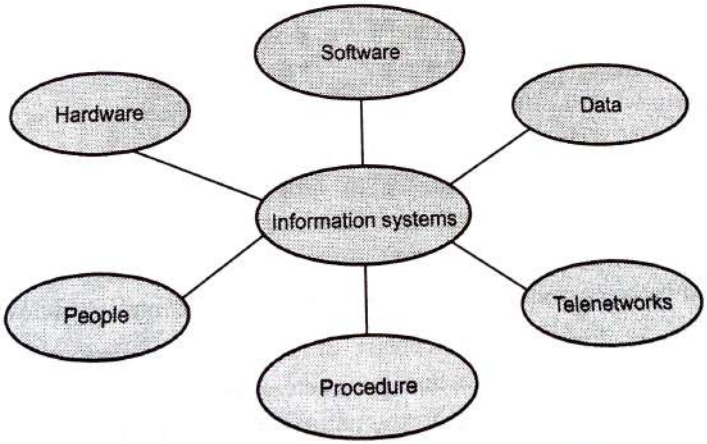


Fig. 4.3.2 Components of information system

- Information systems activities are Input, Processing and Output. 'Input' consists of acquisition of the 'raw data', which is transformed into more meaningful packets of 'Information' by means of 'Processing'.

- The processed information now flows to the users or activities also called as 'Output'. The shortcomings are analyzed and the information is sent back to the appropriate members of the organization to help them evaluate and refine the input. This is termed as 'feedback'.
- Hardware consists of input/output device, processor, operating system and media devices. Software consists of various programs and procedures.
- Database consists of data organized in the required structure. Network consists of hubs, communication media and network devices. People consist of device operators, network administrators and system specialist.
- Information processing consists of input; data process, data storage, output and control. During input stage data instructions are fed to the systems which during process stage are worked upon by software programs and other queries. During output stage, data is presented in structured format and reports.

4.4 Key Management

- Management and handling of the pieces of secret information is generally referred to as **key management.**
- Activities of key management includes selection, exchange, storage, certification, revocation, changing, expiration and transmission of the key.
- Key management is the set of processes and mechanisms which support key establishment and maintenance of ongoing keying relationship between parties, including replacing older key with new keys.
- Two major issues in key management are :
  1. Key life time
  2. Key exposure
- Key life time - Limit of use which can be measured as a duration of time.
- Key management is a set of techniques to initialize, register, update, and recover keys for cryptographic mechanisms in order to ensure privacy, integrity, and authentication among authorized parties.
- Key management may also be called a process to revoke keys from compromised or detected malicious nodes and update keys for non-compromised ones.
- The contents of the key may consist of public/private key pairs, secret keys, non-secret parameters, initialization parameters, and supporting key management in various instances, depending on the nature of the scheme being used.
- Symmetric key cryptography also known as shared key ciphers. It is type of cryptography in which the same key is used by both the sender and receiver for the encryption and decryption of plaintext and ciphertext, respectively.

- Asymmetric key cryptography uses two types in a pair called as public key and private key in a pair. It is also called as public key cryptography. The private key is used to decrypt the cipher text and generate the digital signature, whereas the public key is used to encrypt plaintext and to verify the digital signature.
- Trusted third party is certification authority that grants a digital certificate. The certificate is usually the public key of that organization to whom this certificate is issued. A third trust party acceptable by both sender and receiver is performing the task of guarantor.
- Dynamic Key Management Schemes (DKM) : In dynamic key management schemes, different keys are assigned for different sessions. Once the communication session terminated or finished between the sender and receiver, the keys for the next session will be dynamically assigned to nodes without any revocation or updating command.
- Contributory / Distributed key management schemes : Contributory/Distributive schemes are symmetric cryptographic based solutions characterized by the lack of a trusted third party which is normally responsible for the generation and distribution of the cryptographic keys.
- Centralized key management schemes : These schemes require centralized Trusted Authority (TA) which is designated to generate and distribute a unique session key for all concerned group members in the Internet of Things. The key in Internet of Things update is difficult to manage because of its dynamic topology and its connection is varied with multiarchitecture clients/nodes.
- In static key management schemes, the key is created for the overall lifetime of nodes by either mutual agreement, symmetric cryptography, or centralized certification authority, in asymmetric cryptography.

4.5 Update Management

- Update management represents another term for patch management.
- Patch management is the process that helps acquire, test and install multiple patches (code changes) on existing applications and software tools on a computer, enabling systems to stay updated on existing patches and determining which patches are the appropriate ones. Managing patches thus becomes easy and simple.
- Patch Management is mostly done by software companies as part of their internal efforts to fix problems with the different versions of software programs and also to help analyze existing software programs and detect any potential lack of security features or other upgrades.

- Patching is a process to repair a vulnerability or a flaw that is identified after the release of an application or a software. Newly released patches can fix a bug or a security flaw, can help to enhance applications with new features, fix security vulnerability.
- Unpatched software can make the device a vulnerable target of exploits. Patching a software as and when the patch is released is critical to deny malware access.
- Software vendors release patches to fix vulnerabilities identified after the release of a software or application. Patch Management enables patch testing and deployment which is a critical aspect of cyber security. Quick and instant responses to patch updates would mitigate the chances of data breaches that can cause due to unpatched software.

4.6 Challenges in IoT Security

- The security challenges are as follows :
  a. **Devices are not reachable** : Most of the time a device is not connected.
  b. **Devices can be lost and stolen** : Makes security difficult when the device is not connected.
  c. **Devices are not crypto-engines** : Strong security difficult without processing power.
  d. **Devices have finite life** : Credentials need to be tied to lifetime.
  e. **Devices are transportable** : Will cross borders.
- IoT system has a cloud database that is connected to all your devices. These devices are connected to the internet and it could be accessed by the cybercriminals and hackers. As the number of connected devices increases, chances for hackers to breach the security system gets increased.

4.7 Fill in the Blanks

Q.1 _____ is the process of encoding a plain text message into non-readable form.

Q.2 _____ is a process of transferring an encrypted message back into its normal form.

Q.3 Update management represents another term for _____ management

Q.4 Symmetric key cryptography also known as _____ key ciphers

Q.5 In _____ key management schemes, the key is created for the overall lifetime of nodes by either mutual agreement, symmetric cryptography, in asymmetric cryptography

Q.6 _____ key cryptography uses two types in a pair called as public key and private key in a pair

Q.7 Confidentiality ensures that no one can _____ the message except intended receiver.

4.8 Multiple Choice Questions

Q.1 List the components of information security.

a Network security

b Computer and data security

c Management of information security

d All of these

Q.2 The _____ model of information security evolved from a concept developed by the computer security industry called the C.I.A triangle.

a NSS

b CNSS

c McCumber

d None of these

Q.3 _____ refers to the security flaws in a system that allows an attack to be successful.

a Vulnerability

b Availability

c Integrity

d Confidential

Q.4 When an entire message is encrypted for _____, using either symmetric or asymmetric encryption, the security of the scheme generally depends on the bit length of the key.

a integrity

b non-repudiation

c availability

d confidentiality

Q.5 Components of information system are _____

a software

b network

c database

d all of these

Q.6 The original message is called as _____.

a ciphertext

b plaintext

c cryptography

d encryption

Q.7 The process of converting plaintext to ciphertext is called as _____.

a encryption
b decryption
c substitution
d transposition

Q.8 Information has _____ when it is protected from disclosure or exposure to unauthorized individuals or systems.

a integrity
b confidentiality
c availability
d authentication

Q.9 Secret key cryptography is also known as _____.

a symmetric key cryptography
b asymmetric key cryptography
c private key cryptography
d quantum cryptography

Q.10 The _____ is a widely used information security model that can guide an organization's efforts and policies aimed at keeping its data secure.

a CIA triad
b ARPANET
c NIST
d None

Q.11 A loss of _____ is the unauthorized disclosure of information.

a integrity
b availability
c authentication
d confidentiality

Q.12 Which of the following are the security requirements triad ?

a Confidentiality
b Integrity
c Availability
d All of these

Q.13 _____ prevents either sender or receiver from denying a transmitted message.

a Nonrepudiation
b Replay
c Fabrication
d Masquerade

Answer Keys for Fill in the Blanks

Q.1	Encryption	Q.2	Decryption	Q.3	patch
Q.4	shared	Q.5	static	Q.6	Asymmetric
Q.7	read				

Answer Keys for Multiple Choice Questions

Q.1	d	Q.2	b	Q.3	a	Q.4	d
Q.5	d	Q.6	b	Q.7	a	Q.8	b
Q.9	a	Q.10	a	Q.11	d	Q.12	d
Q.13	a						

❑❑❑

5 IoT Applications and Case Study

Syllabus

Broad categories of IoT applications : Consumer IoT, Commercial IoT, Industrial IoT, Infrastructure IoT, Military Things (IoMT)
IoT Case studies : Home automation with IoT, River water pollution monitoring, Smart city street light control and monitoring, Health care monitoring, Voice Apps on IoT device.

Contents

5.1 Broad Categories of IoT Applications
5.2 Home Automation with IoT
5.3 River Water Pollution Monitoring
5.4 Smart City Street Light Control and Monitoring
5.5 Health Care Monitoring
5.6 Voice Apps on IoT Device

5.1 Broad Categories of IoT Applications

- With the advent of the Internet of Things era, homes, cities and almost everything is becoming smart. Those smart objects can sense the physical world by obtaining data from sensors, affecting the sensed world by triggering actions using actuators, engage users by interacting with them whenever necessary, and process gathered data to provide useful information to us.
- An increasing number of physical objects are being connected to the Internet at an unprecedented rate realizing the idea of the Internet of Things.
- Examples of such objects include thermostats and Heating, Ventilation, and Air Conditioning (HVAC) monitoring and control systems that enable smart homes. There are also other domains and environments in which the IoT can play a remarkable role and improve the quality of our lives.
- These applications include : Transportation, health care, industrial automation, and emergency response to natural and man-made disasters where human decision making is difficult.
- Currently, the IoT vision is mainly focused on the technological and infrastructure aspect, as well as on the management and analysis of the huge amount of generated data.
- Consumer devices are always a public concern, but there are currently five types of IoT applications :
  a) Consumer IoT : Such as light fixtures, home appliances, and voice assistance for the elderly.
  b) Commercial IoT : IoT applications in the healthcare and transport industries, such as smart pacemakers, monitoring systems, and vehicle to vehicle communication (V2V).
  c) The Industrial Internet of Things : IIoT includes digital control systems, statistical evaluation, smart agriculture, and big industrial data.
  d) Infrastructure IoT enables the connectivity of smart cities through infrastructure sensors, management systems, and user-friendly user apps.
  e) Military Things (IoMT) applying IoT technologies in the military field, such as robots for surveillance and human-wearable biometrics for combat.

5.1.1 Consumer IoT

- Consumer IoT refers to the billions of physical personal devices, such as smartphones, wearables, fashion items and the growing number of smart home appliances, that are now. Connected to the internet, collecting and sharing data.

- Consumers want IoT products that fit seamlessly into their daily lives and simplify routine tasks. Finding keys, locking the front door, turning lights on and off, these tasks can be controlled with wireless technologies.
- Fig. 5.1.1 shows consumer IoT.

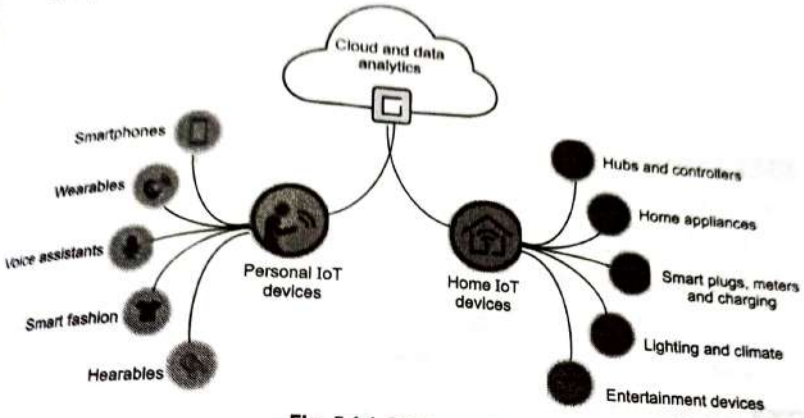


Fig. 5.1.1 Consumer IoT

- Consumer smart assistants, such as Amazon Alexa, Google Home and Apple Homepod, are the hubs for consumer IoT. They build on stable technologies : Bluetooth (BLE,) Wi-Fi and derivatives of ARM mobile processors. On the software side, they all rely on Natural Language Processing (NLP) and acoustics that are fairly mature.
- The real-time data collected from connected devices and smart connected consumer products can generate insights and provide personalized information and services to the consumers.
- Smart homes, connected vehicles, wellness and fitness through wearables, smart retail and smart farming are a few examples of how businesses can achieve their short - term and long - term goals.
- Business-to-consumer value : Using product data to inform better customer experience, personalization, reliability, preemptive support, convenience, security, product/service improvement, etc.
- Business-to-business value : Using product data to inform operational efficiencies, improve sales conversion, marketing decisions, security, inventory optimization, etc.

- Business-to-ecosystem value : Using product data to inform partner strategies, integrations, APIs, content needs, supply chain efficiencies, etc.
- Consumer IoT has long held a reputation for poor security standards. Homes today are filled with connected devices. It's not just smart speakers, it's app-enabled espresso machines and wifi - connected security cameras.
- The majority of IoT devices purchased for the home are relatively cheap and little effort is made to protect them at a hardware or software level at this end of the spectrum by manufacturers.

5.1.2 Commercial IoT

- Commercial IoT lies between consumer IoT and industrial IoT applications.
- Commercial IoT refers to devices and systems for business and enterprise use. These can vary broadly by sector. IoT is increasingly leveraged in healthcare, event venues, office buildings, and more.
- Specifically, when we talk about commercial IoT some of the key use cases that we see are intelligent asset tracking, smart office and buildings, connected lighting, sensing and monitoring of all types and location services.

5.1.3 Industrial IoT

- Industrial IoT (IIoT) infrastructure should be protected by a comprehensive security solution (device-to-cloud) that does not disrupt operations, service reliability or profitability.
- Industrial IoT refers to devices and systems used in manufacturing or industrial purposes. Examples include supply chain asset tracking and fleet tracking.
- Fig. 5.1.2 shows industrial IoT system architecture. (Refer Fig. 5.1.2 on next page)
- The common requirements of the IoT are technical requirements independent of any specific application domain.
- IoT non-functional requirements refer to the requirements related to the implementation and operation of the IoT itself , i.e. Interoperability, scalability, reliability, high availability, adaptability, manageability.
- IoT functional requirements refer to the requirements related to the IoT actors, these requirements have been categorized as,
  1. Application support requirements
  2. Service requirements
  3. Communication requirements
  4. Device requirements

- Business-to-ecosystem value : Using product data to inform partner strategies, integrations, APIs, content needs, supply chain efficiencies, etc.
- Consumer IoT has long held a reputation for poor security standards. Homes today are filled with connected devices. It's not just smart speakers, it's app - enabled espresso machines and wifi - connected security cameras.
- The majority of IoT devices purchased for the home are relatively cheap and little effort is made to protect them at a hardware or software level at this end of the spectrum by manufacturers.

5.1.2 Commercial IoT

- Commercial IoT lies between consumer IoT and industrial IoT applications.
- Commercial IoT refers to devices and systems for business and enterprise use. These can vary broadly by sector. IoT is increasingly leveraged in healthcare, event venues, office buildings, and more.
- Specifically, when we talk about commercial IoT some of the key use cases that we see are intelligent asset tracking, smart office and buildings, connected lighting, sensing and monitoring of all types and location services.

5.1.3 Industrial IoT

- Industrial IoT (IIoT) infrastructure should be protected by a comprehensive security solution (device-to-cloud) that does not disrupt operations, service reliability or profitability.
- Industrial IoT refers to devices and systems used in manufacturing or industrial purposes. Examples include supply chain asset tracking and fleet tracking.
- Fig. 5.1.2 shows industrial IoT system architecture. (Refer Fig. 5.1.2 on next page)
- The common requirements of the IoT are technical requirements independent of any specific application domain.
- IoT non-functional requirements refer to the requirements related to the implementation and operation of the IoT itself , i.e. Interoperability, scalability, reliability, high availability, adaptability, manageability.
- IoT functional requirements refer to the requirements related to the IoT actors, these requirements have been categorized as,
  1. Application support requirements
  2. Service requirements
  3. Communication requirements
  4. Device requirements

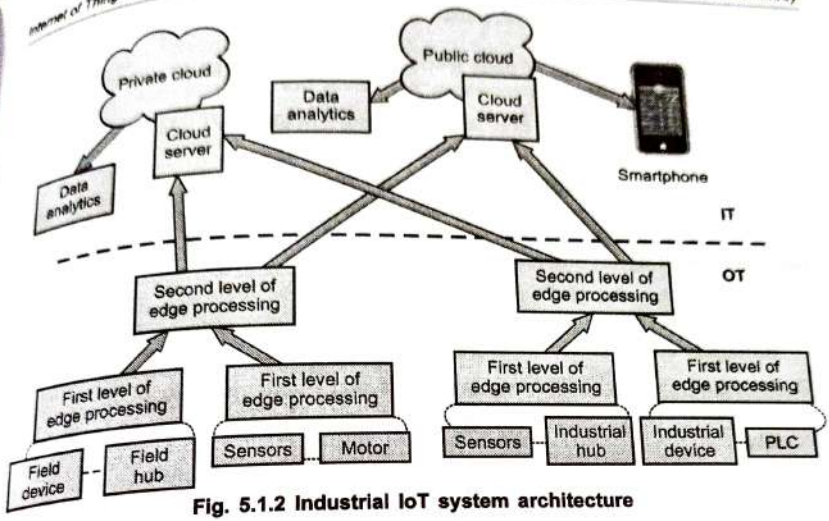


Fig. 5.1.2 Industrial IoT system architecture

5. Data management requirements
6. Security and privacy protection requirements.

- The IoT application capabilities for industrial application should meet requirements such as :
  1. Reliability : Reliable IoT devices and systems should allow a continuous operation of industrial processes and perform on-site activities
  2. Robustness : The IoT application and devices should be robust and adapted to the task and hard working conditions
  3. Simple to use
  4. Low maintains cost
  5. Support optimal and adaptive set of features
  6. It must have reach sensing and data capabilities
  7. Industry grade support and services. The IoT applications should be supported over years in operation by a set of rich tools and continuously updated services
  8. Support standardization
  9. Security and safety.

- The industrial internet of things will be shaped by the appearance of three features :
  1. The enhancement of basic mechanical devices through sensors and other data producing devices ("smartness");
  2. The possibilities of ever faster and more flexible data allocation, transfer and processing (computing capacities);
  3. The ever increasing digital interconnectivity between the engaged devices and computational capacities (digital integration).

Challenges faced by IoT industry applications

1. Security. As the IoT connects more devices together, it provides more decentralized entry points for malware. Less expensive devices that are in physically compromised locales are more subject to tampering.
2. Trust and Privacy. With remote sensors and monitoring a core use case for the IoT, there will be heightened sensitivity to controlling access and ownership of data.
3. Complexity, confusion and integration issues. With multiple platforms, numerous protocols and large numbers of APIs, IoT systems integration and testing will be a challenge to say the least. The confusion around evolving standards is almost sure to slow adoption.
4. Evolving architectures competing standards. With so many players involved with the IoT, there are bound to be ongoing turf wars as legacy companies seek to protect their proprietary systems advantages and open systems proponents try to set new standards. There may be multiple standards that evolve based on different requirements determined by device class, power requirements, capabilities and uses. This presents opportunities for platform vendors and open source advocates to contribute and influence future standards.
5. Concrete use cases and compelling value propositions. Lack of clear use cases or strong ROI examples will slow down adoption of the IoT. Although technical specifications, theoretical uses and future concepts may suffice for some early adopters, mainstream adoption of IoT will require well-grounded, customer-oriented communications and messaging around "what's in it for me."
6. There are several wireless standards which can be used to connect devices to a network, most are still developing. That means delays as products play catch - up with new networking standards.

5.1.4 Infrastructure IoT

- Five things are making up an IoT infrastructure are IoT platforms, access technologies, data storage and processing, data analytics, and security. An essential part of any internet - connected device, these five pillars are what enable growth for future IoT solutions.

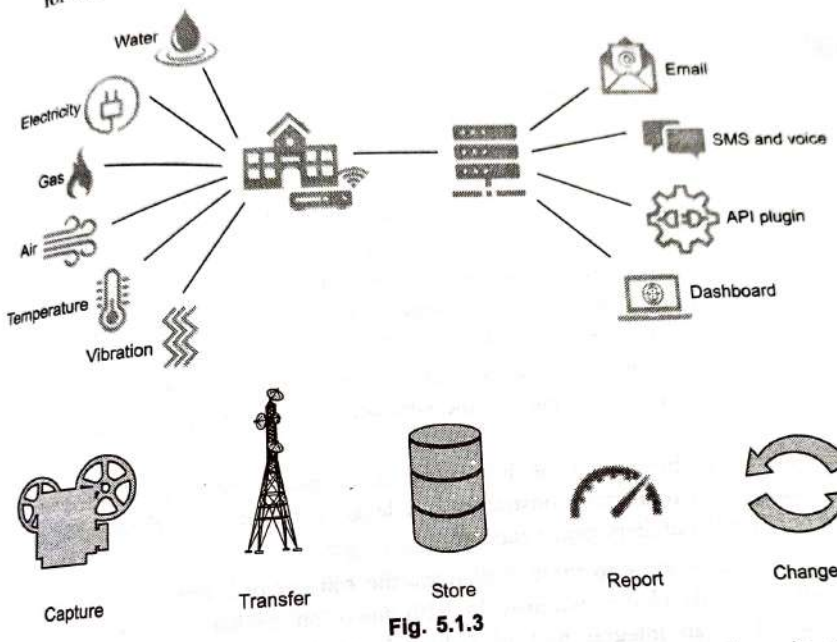


Fig. 5.1.3

- Smart infrastructure also includes implementing charging stations in parking systems, city fleets, shopping malls and buildings, airports, and bus stations across the city. Electronic Vehicle (EV) charging platforms can be integrated with IoT to streamline the operations of EV charging and addresses the impact of the power grid.
- A simple IoT infrastructure can be broken down into five key areas, capture, transfer, store, report and change.
- Capture covers the type of data source and the way the information is captured. This could be temperature or power consumption using digital, analogue or Modbus inputs. This data can then be transferred directly to the cloud using LoRaWAN for example or it can be transmitted to a gateway first using RF 868 mHz and then from the gateway to the cloud using GPRS 2G/3G.

- The data could be stored on the IoT supplier's cloud or sent via an API to the user's own cloud. From the cloud the data can be Reported and ingested in several ways. This could be in the form of a dashboard, an email alert or via an API into existing reporting tools.
- Once the information has been ingested it can be used to influence some form of change may that be a physical input or an automated response back via the IoT infrastructure or another system.

5.1.5 Military Things (IoMT)

- The IoMT is the application of Internet of Things (IoT) technologies and concepts to the military domain. Military IoT adoption is still in its infancy, however defence companies and the armed forces are eager to prepare, understand and leverage the IoT.
- There are clear military benefits from the use of IoT devices for the armed forces, ranging from vehicle maintenance to personnel monitoring to stock control.
- In today's scenarios military commanders may have only several minutes to be presented with an assessment of the situation in a way that is quickly assimilated, to assess potential courses of action and to make their decision.
- It needs to draw upon all possible sources to ensure that the most complete and relevant picture can be created of the situation, and the implications of different decisions understood.
- However, the integration of heterogeneous sensors and systems diverse in technology, environmental constraints and level of fidelity is a challenging issue not only for the military organizations.
- To provide a response to these challenges, the concept of Internet of Things (IoT) is extensively developed world-wide with focus on civilian applications. IoT is considered as an integral part of Future Internet and could be defined as a **dynamic global network infrastructure** with self-configuring capabilities based on standard and interoperable communications protocols where physical and virtual things have identities, physical attributes and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network.
- The concept of IoT developed for civilian market does not fit the military requirements. Highly dynamic and disruptive environment, fragile nature of wireless communications, especially at lower tactical levels, specific requirements for speed and fidelity with which information must be exchanged and analysed requires robust and effective mechanisms, as well as specific applications that present IoT solutions do not provide.

- The Internet of military things refers to a network of sensors, wearables, and other devices that use cloud and edge computing to give advantage to a fighting force.
- Long-Term Evolution (LTE) is a standard for wireless broadband communication for mobile devices that increases the capacity and speed of data transfer and provides several add - on facilities.
- IoMT is a type of Internet of Things designed for combat operations and warfare based upon smart technology and artificial intelligence. It comprises a complex network of interconnected entities in the military domain that continually communicate with each other to accomplish a broad range of activities in a more efficient and informed manner.

5.2 Home Automation with IoT

- Home automation is the automatic control of electronic devices in your home. These devices are connected to the Internet, which allows them to be controlled remotely.
- Interconnected devices enable to intelligently monitor and control smart homes in a future Internet of Things.
- Energy saving applications, for example, control indoor climate and electricity usage by employing context information to switch off appliances (e.g., lights, computers), reduce room temperature, close windows, or stop warm water circulation.
- Home automation works on three levels :
  1. **Monitoring :** Monitoring means that users can check in on their devices remotely through an app. For example, someone could view their live feed from a smart security camera.
  2. **Control :** Control means that the user can control these devices remotely, like planning a security camera to see more of a living space.
  3. **Automation :** Finally, automation means setting up devices to trigger one another, like having a smart siren go off whenever an armed security camera detects motion.

5.2.1 Smart Lighting

- Smart control the lights with automation signal system to save energy. Smart, connected lighting is the next - generation energy - efficient LED products with additional sensors to sense things such as occupancy and temperature.

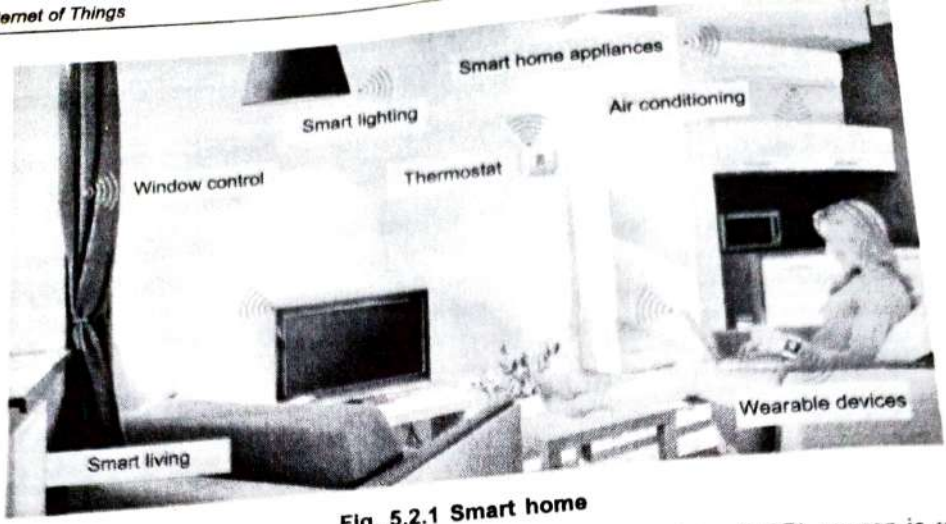


Fig. 5.2.1 Smart home

- In automatic light control system, Light Dependent Resistor (LDR) sensor is used to detect bright /medium /dim /dark conditions.
- It is simple enough to envision the addition of sensors and communications to create that initial concept of smarter, more adaptive lighting. If people are present, turn the lights on; if not, turn them off. Or use your smart phone to connect to the lighting system and tune it to the desired brightness level or to a particular color.
- Smart lighting is considered the one of the main solutions for energy reduction by means of controlling lighting level according to desired need with minimum energy consumption.
- Smart - lighting systems utilize motion and light sensors for performing the control algorithms.
- The system uses motion and light sensors for detecting the surrounding environment. There are lamps controlled with the specific lighting level in order to supply the adequate amount of lighting required without affecting the user visibility.
- Certainly the required lighting level is strongly dependent on the weather conditions. In clear weather at night might require more luminance than cloudy one, due to the reflection from the clouds.
- While during mist and foggy weathers require the highest possible lighting level, as the visibility reaches its lowest. On snowy weather it might require an intermediate level between clear and foggy.

- During night it requires high lighting levels, while at day it needs just fade level to provide guidance or turn off if the weather is clear. The lighting concentration in the yard is affected by the above conditions.

5.2.2 Smart Appliances

- The role and scope of smart appliances in the home (Washer, dryer, refrigerator, dishwasher, fridge, freezer, air conditioner, vacuum cleaner and so on) is on the increase with the market being estimated to have a year on year compound growth of slightly over 15 %.
- Connecting everyday objects to the internet is an essential element of the IoT. Some appliance suppliers use a low power wireless network to communicate over such as bluetooth, whilst others utilise the existing higher powered Wi-Fi network used for a tablet or computer wireless connectivity. Once a network is in place objects can populate the home environment and communicate with the user and each other.
- The ability of an object to respond to remote commands and change its behaviour makes it an active device, such as the new Hive heating thermostat or a Sky+ box.
- Where the remote object has no ability to respond to remote control requests then it is considered passive as with some fixed cameras, microphones or temperature sensors.
- IR Sensor : It will be activated in the automated mode to detect person entering or coming out of the room and set a counter based on that. If the counter show there is a person inside it will light up the room automatically and turn on the AC depending upon the temperature reading.
- Sensors provide data about motion, occupancy, glass breakage, door and window openings, water leaks, light intensity, temperature, energy consumption, camera, and even appliance plug insertion or removal.
- Controllers turn power on and off or adjust settings on appliances, furnaces, air conditioners, space heaters, fans, pool pumps, water heaters, lighting, home theatres, music, motorized blinds, door locks, and plug loads.
- To be deemed intelligent, an appliance's sensors and controllers should use internet protocol communication.
- Smart refrigerators can keep track of the items stored and send updates to the users when an item is low on stock.
- Smart TV allow user to search video and movies from the Internet on a local storage drive, search TV schedule, fetch news and other things from the Internet.
- OpenRemote is the professional open source middle-ware for an Internet of Things. Integrate any device or protocol, and design your own user interface and

automation. Use our online designer, sync to the controller, and control with this app.

- OpenRemote is a state of the art open source software platform for building control and automation.
- OpenRemote allows for designing a fully customizable building and home control solution without the need to actually write code.

5.2.3 Intrusion Detection

- Intrusion Detection System (IDS) includes both hardware and software mechanisms and IDS is responsible for identifying malicious activities by monitoring network environment and system.
- The purpose of home intrusion detection system is to detect intrusions using sensors and raise alerts, if necessary.
- With the help of Light dependent resistor and PIR motion sensor, it detect the motions in the room. If a motion is detected, system capture the image with the help of a webCam and store locally. Now the alerts are sent to the user with the captured image.
- Fig. 5.2.2 shows block diagram of intrusion detection.

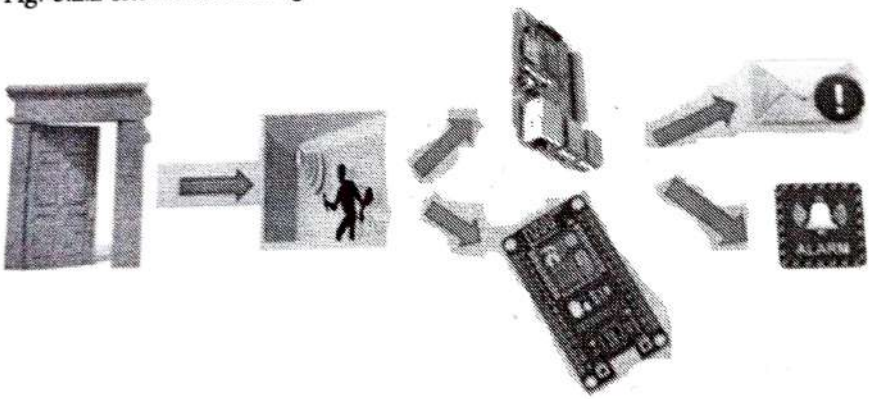


Fig. 5.2.2 Block diagram of intrusion detection

- To detect any form of intrusion in restricted areas and report it immediately, following concept is used.
  1. A PIR sensor is required to detect the presence of any human being in the room.
  2. An RFID is required to validate the presence of the person in the room by tallying his identity with those in the database.

3. A camera is required to click the picture of the room and send it via email as an alarm.
4. An internet connection is required to register all these movements on a website so that it can be accessed from any place and any device.

- The different input / output devices are controlled using TCP/IP over the IEEE 802.11 standard protocol. Data being gathered from sensors, such as PIR sensors, temperature sensors, IR transmitter and receiver is being processed on micro - controller as a server.
- Passive Infrared Sensor (PIR) Sensor : PIR sensor is an electronic sensing device that senses infrared (IR) light emitted from entities in its field of view and used to detect motion in its range. It is activated only in the security mode to detect any unwanted motion at the entrance. If any unwanted movement is detected then it will signal the microcontroller to take necessary steps.
- Alarm : It will only be activated in the security mode when some intruder is detected by the PIR motion sensor.
- Cloud controlled intrusion detection is possible by using location aware services. Here geo - location of each node is independently detected and stored in the cloud.
- Some intrusion detection system uses UPnP technology. It is based on image processing to recognize the intrusion.

5.2.4 Smoke for Gas Detection

- Smoke or gas detector sensor which detects the smoke and turns on the buzzer alarm and all these are update on the web page.
- MQ2 is a semiconductor type sensor, which can appropriately sense the presence of smoke, LPG, methane, butane, propane and other hydrocarbon.
- When it comes in contact with the gas to be monitored, the electrical resistance of the sensor decreases; enabling the microcontroller to respond to the situation.
- When it detects the concentration of combustible gas in the air it outputs its reading as an analog voltage. The sensor can measure concentrations of flammable gas of 300 to 10,000 ppm. The sensor can operate at temperatures from – 20 to 50 °C and consumes less than 150 mA at 5 V.
- The MQ-2 smoke sensor reports smoke by the voltage level as output. The more smoke is there, the greater the voltage output. The MQ-2 also has a built-in potentiometer to adjust the sensitivity to smoke.
- By adjusting the potentiometer, one can change how sensitive it is to smoke, so there is a form of calibrating it to adjust how much voltage it will give in relation to the smoke it is exposed to.

5.3 River Water Pollution Monitoring

- In conventional systems, the monitoring process involves the manual collection of sample water from various regions, followed by laboratory testing and analysis. This process is ineffective, as this process is arduous and time-consuming and it does not provide real - time results.
- The quality of water should be monitored continuously, to ensure the safe supply of water from any water bodies and water resources. Hence, the design and development of a low - cost system for real-time monitoring of water quality using the Internet of Things (IoT) is essential.
- Monitoring water quality in water bodies using Internet of Things helps in combating environmental issues and improving the health and living standards of all living things.
- The proposed system monitors the quality of water relentlessly with the help of IoT devices, such as, NodeMCU. The in-built Wi-Fi module is attached in NodeMCU which enables internet connectivity transfers the measured data from sensors to the Cloud.
- Fig. 5.3.1 shows water monitoring system.

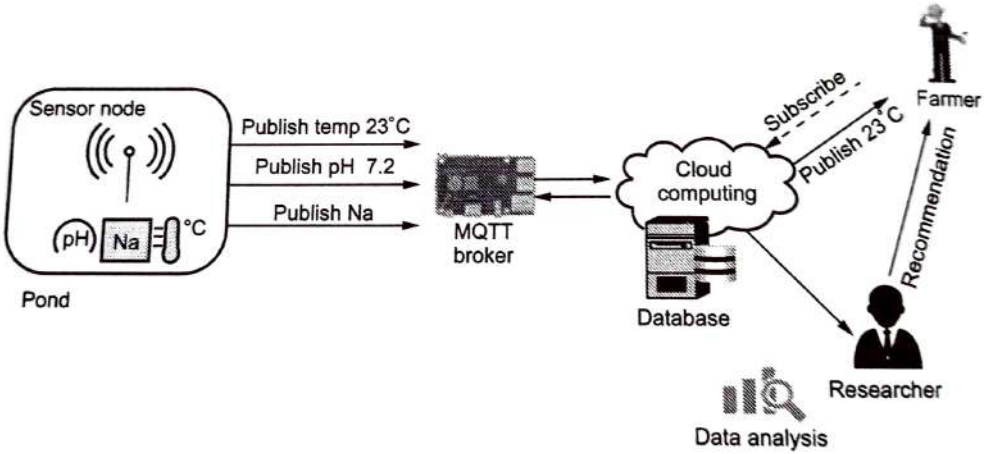


Fig. 5.3.1 Water monitoring system

- The system mainly consists of sensor node as publishers, and Raspberry pi MQTT broker, and mobile client devices as subscribers. The sensor nodes are built with small embedded devices, LoRa wireless interface, and water quality sensors, i.e. water temperature sensor, pH sensor, and salinity sensor.
- **Water quality sensor :** The pH meter is used for the quality check if water is safe for use.

- **Temperature sensor** : A device which gives temperature measurement as an electrical signal is called as temperature sensor. This electrical signal will be in the form of electrical voltage and is proportional to the temperature measurement.
- **Water level sensor** : This sensor will help us decide if we have enough quantity of water to be supplied. An ultrasonic wave is triggered from the sensor and distance to target is determined by calculating the time required after the echo is returned. The sensor emits a high-frequency pulse, generally in the 20 kHz to 200 kHz range, and then listens for the echo.
- Sensor node technical specification is listed below :

Microcontroller	Arduino MEGA 2560
Wireless Interface	LoRa Shield with 915 MHz Antenna
Sensors	Water Temperature, Salinity, pH
Battery	12 V 18AH Rechargeable Sealed Lead Acid
Solar Cell	20 WP 12 V
Packet Size	17 bytes
Transmission Interval	60 seconds

5.4 Smart City Street Light Control and Monitoring

- The number of urban residents is growing by nearly 60 million every year. In addition, more than 60 percent of the world's population will be living in cities by 2050.
- As a result, people occupying just 2 percent of the world's land will consume about three-quarters of its resources. Moreover, more than 100 cities of 1 million people will be built in the next 10 years.
- Over the past decade, the city of Amsterdam, the Netherlands, has developed a vision for collaborating, envisioning, developing, and testing numerous connected solutions that could pave the way to a smarter, greener urban environment.
- Fig. 5.4.1 shows concept of smart city. (Refer Fig. 5.4.1 on next page.)
- Smart city includes :
  1. Smarter management of city infrastructure using big data analytics
  2. Collaboration across multiple and disparate agencies using cloud technologies
  3. Real - time data collection, enabling quick response using mobile technologies
  4. Enhanced security : Improved public safety and law enforcement, and more efficient emergency response
  5. Better city planning improved schematics, project management and delivery

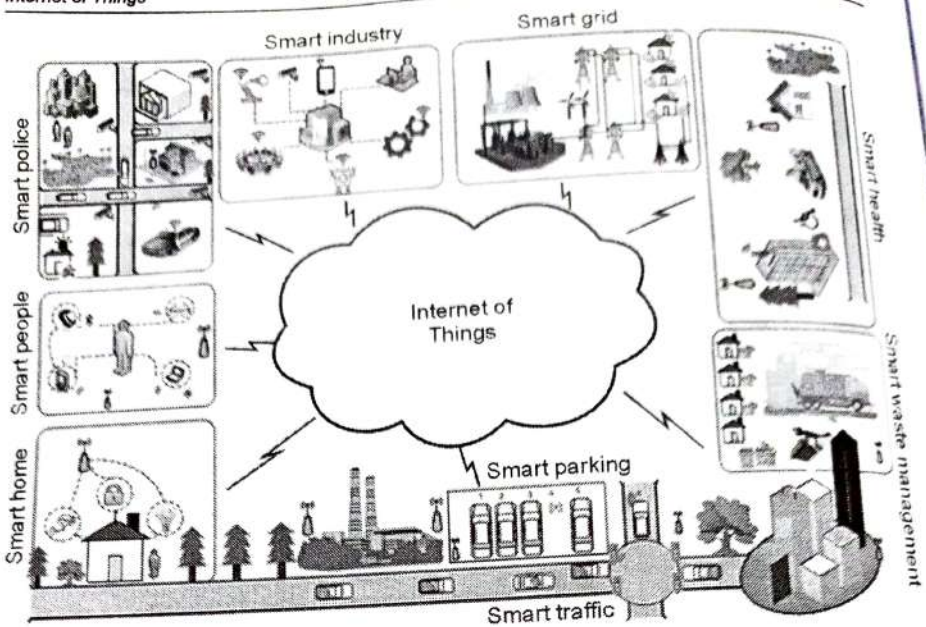


Fig. 5.4.1 Smart city

6. Networked utilities smart metering and grid management
7. Building developments more automation, and better management and security.

- With smart city applications producing continuous large data from heterogeneous sources, existing relational database technologies are inadequate to handle such huge amounts of data given the limited processing speed and the significant storage expansion cost.
- To address this problem, big data processing technologies, which are based on distributed data management and parallel processing, have provided enabling platforms for data repositories, distributed processing, and interactive data visualization.

5.4.1 Smart Parking

- Traffic congestion is major problem in big cities. Searching for a parking space is a routine (and often frustrating) activity for many people in cities around the world.

- After finding parking space to the driver, he parks the vehicle, it maybe spend small amount of time to looking for a city council parking attendant to pay the parking fees.
- The smart parking system is designed by making use of some IOT supportable hardware's such as raspberry pi, auridino boards etc.
- Smart parking systems typically obtains information about available parking spaces in a particular geographic area and process is real - time to place vehicles at available positions.
- It involves using low-cost sensors, real-time data collection, and mobile-phone-enabled automated payment systems that allow people to reserve parking in advance or very accurately predict where they will likely find a spot.
- When deployed as a system, smart parking thus reduces car emissions in urban centers by reducing the need for people to needlessly circle city blocks searching for parking.
- It also permits cities to carefully manage their parking supply smart parking helps one of the biggest problems on driving in urban areas; finding empty parking spaces and controlling illegal parking.

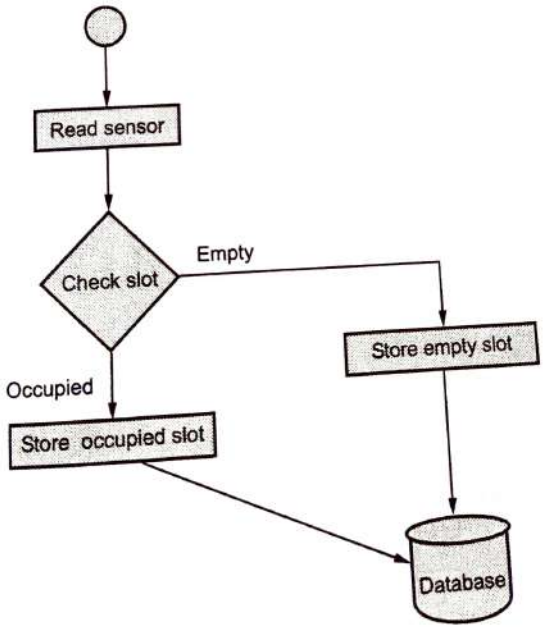


Fig. 5.4.2 Process specification for smart parking IoT system

- Smart parking application can be accessed by drivers from smart phones, tables. Sensor is used for each parking slot, to detect whether the slot is empty or occupied.
- Local controller collect the information and send to server using Internet. Fig. 5.4.2 shows process specification for smart parking IoT system.
- Each parking slot contains the sensor and it reads at regular intervals. Sensor sends the status information to local processing centre.
- Fig. 5.4.3 includes four layers : a sensing, networking, middleware and application layer. (See Fig. 5.4.3 on next page).
- Sensing layer defines a platform where sensor devices are embedded into the parking lot to detect car presence/absence, and RFID devices located at the parking gates and strategic points of the parking are used to identify cars based on a unique mapping between RFID tags and car.
- Networking Layer : TCP/IP over Ethernet for connecting the gateway to the parking server and database and Internet access for remote access to the smart parking system from outside.
- Middleware layer hosts different databases and associated servers and manages all of the software intelligence provided by the smart parking system to provide smart services to users by enabling communication between the application layer where services are requested and the lower layers where smart devices are embedded into the parking lot to provide smart services.
- The application layer is the layer where the different services are defined and provided to different users. Client devices have been connected via the TCP/IP protocol to a parking database.
- Parking availability status by integrating into the car detection system sources of light on parking spots, which are controlled by actuators to inform of the status of a parking spot : e.g., red for occupied, green for empty, yellow for reserved and blue for out of service.
- Remote availability checking using the Internet and/or the GSM network to check in real time the availability of the smart parking system.
- The data of smart parking lots are able to provide profits for both customers and merchant's daily lives in the smart cities. This service works based on road sensors and intelligent displays which lead drivers to the best path for parking in the city.

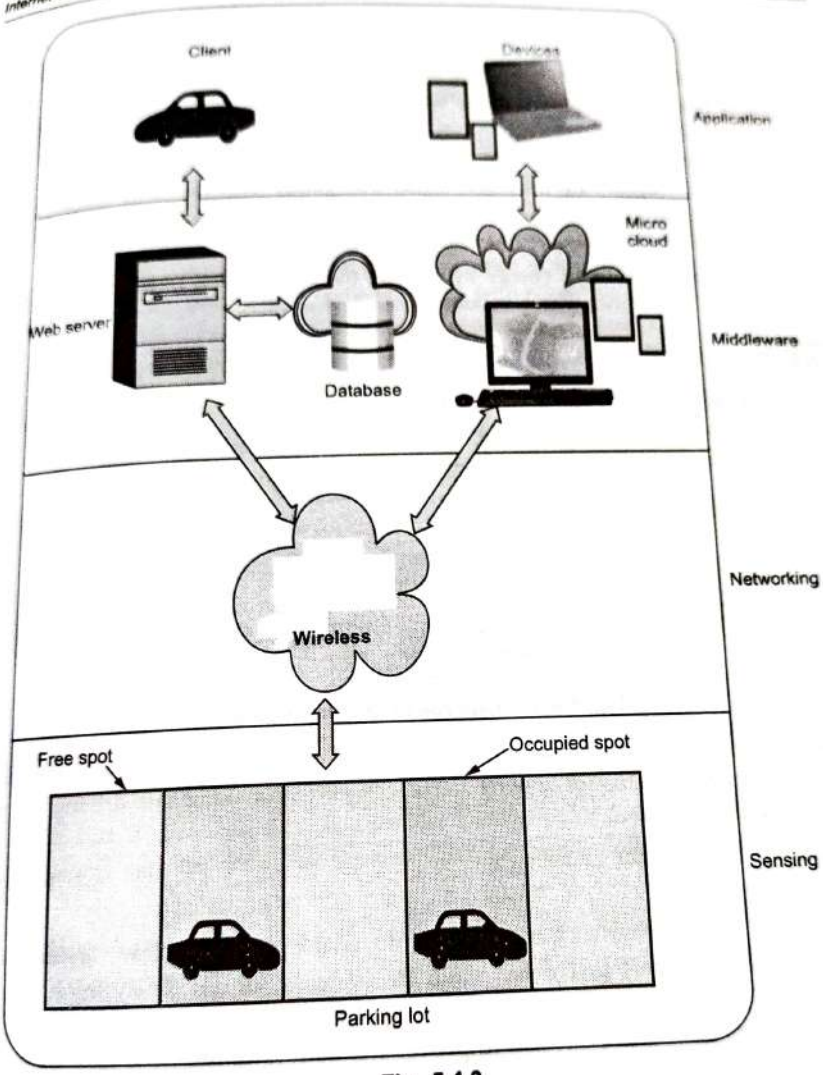


Fig. 5.4.3

5.4.2 Smart Lighting

- The street lighting is one of the largest energy expenses for a city. The street light section comprises of all the light lamps in an area with current sensors and RF

module. 'N' street lights of this section communicates with local controller unit wirelessly through RF module (Zigbee). 'N' local controller unit communicates with main server through IoT due to its global coverage area.

- Smart light infrastructure is the backbone of the IoT in smart cities. Smart and wireless street light luminaries can act as service gateways for other street level IoT devices.
- Smart street lights are intelligent lights that gather dynamic data i.e. data that keep changing dynamically by time, through some sensors and generate required information for the request claimed by a citizen on road.
- Smart street light saves energy by sensing the surrounding through their sensors expecting some other sensor in some other device.

5.4.3 Smart Roads

- Sensor is installed on road to provides road traffic condition, travel time estimation, congestion and accident.

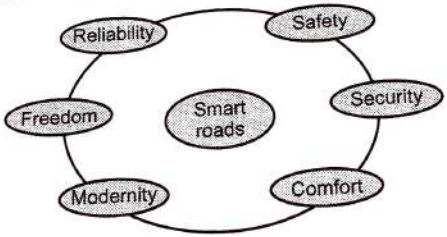


Fig. 5.4.4 Smart roads characteristics

- Sensor collect this information and stored on the central database using cloud. This information helps for solving traffic congestion, making safe driving, keeping road condition upto date.
- User can access the information from the cloud. User also get real time information.
- Real time traffic maps can be obtained to enable smooth flow. Traffic can be reduced with systems that detect alternate routes. User get timely information so they can locate a traffic free road, saving time and fuel. This information can reduce traffic jams and pollution improves the quality of life.

5.5 Health Care Monitoring

- The World Health Organization (WHO) defines E - health as : E - health is the transfer of health resources and health care by electronic means. It encompasses

three main areas : The delivery of health information, for health professionals and health consumers, through the internet and telecommunications.

- E - health provides a new method for using health resources - such as information, money, and medicines and in time should help to improve efficient use of these resources.
- E - health brings special characteristics. The monitoring device's environment is a patient; a living and breathing human being. This changes some of the dynamics of the situation. Human interaction with the device means batteries could be changed, problems could be called in to technical support and possibly be resolved over the phone rather than some type of service call. In most cases, the devices on the patient are mobile not static with regard to location.
- Fig. 5.5.1 shows high level E - health ecosystem architecture.

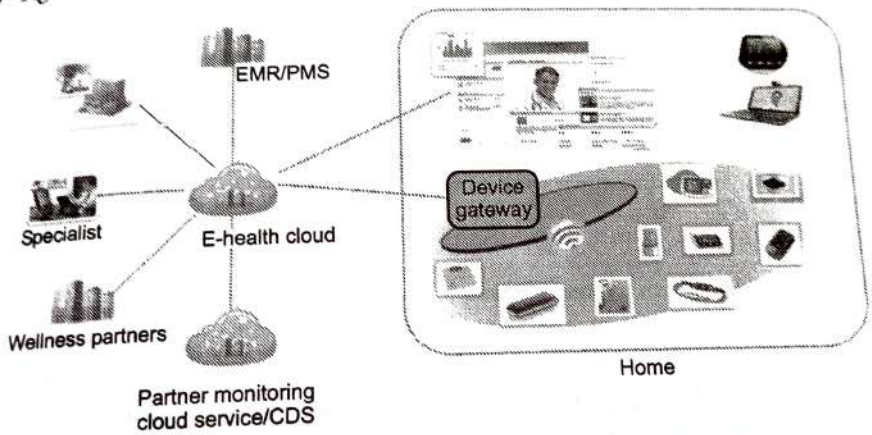
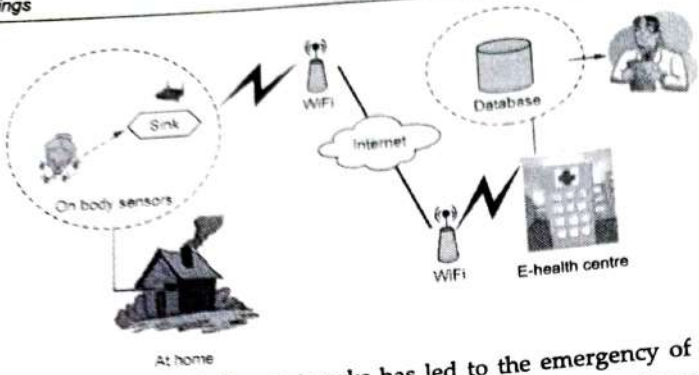


Fig. 5.5.1 High level E - health ecosystem architecture

- The data flow architecture focuses on the source of the data, the destination the data and path the data. The source of the data is typically the sensor.
- The data can be either locally cached or is sent to the upstream systems without storing in the sensor. The path taken by the data includes a gateway, which can also cache some of the data and do distributed processing.
- Intermediate hubs can also store and process the data to filter out or make certain decisions. A distributed rules engine is used to make distributed decisions at the closest point of care. This enables data traffic to be filtered and processed efficiently without having every data being processed by the cloud service



- The development of wireless networks has led to the emergency of a new type of E - healthcare system, providing expert-based medical treatment remotely on time.
- With the E - healthcare system, wearable sensors and portable wireless devices can automatically monitor individuals' health status and forward them to the hospitals, doctors and related people.
- The system offers great conveniences to both patients and health care providers. For the patients, the foremost advantage is to reduce the waiting time of diagnosis and medical treatment, since they can deliver the emergent accident information to their doctors even if they are far away from the hospital or they don't notice their health condition.
- In addition, E - health system causes little interruption to patient's daily activities. For the health care providers, after receiving the abnormal signals from the patients, appropriate treatment can be made, which saves medical resources.
- Furthermore, without direct contact with medical facilities, medical personnel or other patients, the patients are unlikely to be infected with other diseases.
- However, to ensure the security and privacy of patient's medical records encounters a lot of challenges :
  1. How to achieve the confidentiality and integrity of patients' information,
  2. The security of wireless body area network,
  3. The privacy and unlink ability of patients' health status,
  4. The undeniability and unlinkability of doctors' treatment,
  5. The location privacy of patients, the fine - grained access control of patient's medical record, the mutual authentication between patients and hospitals, etc.
- It would be useful to create an up-to-date bibliography on secure E - healthcare systems.

5.6 Voice Apps on IoT Device

There will be three types of voice communication in IoT environments :

1. Bi - directional voice communication
2. Mono - directional voice communication
3. Voice recognition.

- Reasons that voice is suited to a range of IoT applications :
  1. Speech is the natural mode of communication for humans. It is both intuitive and easier to convey commands verbally.
  2. Voice recognition is particularly appealing when the human's hands or eyes are otherwise occupied.
  3. Voice telephony is an efficient means of bi - directional voice communication with machines that can listen, and respond without the need for complex commands.
  4. Cost saving factors : Voice integration could potentially challenge the need for a touch screen on many devices, as it reduces the cost for devices that will be dormant for the majority of the time.
- The IoT market is broad and encompasses a range of consumer, commercial and industrial applications where voice can play a role. There are significant differences between the drivers for implementing voice into consumer products and from those that drive the same technology in the consumer market.

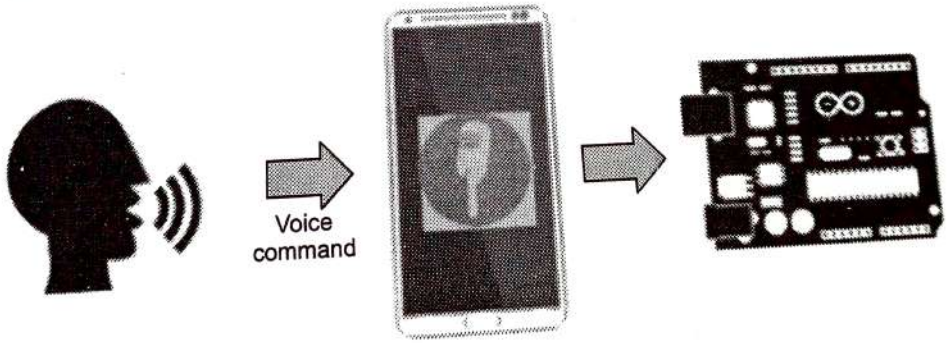


Fig. 5.6.1

- Voice is a feature that does not need to make any consideration for infrastructure, other than the need for an Internet connection.
- Consumer applications for voice include virtual assistants on smartphones as well as devices that do not include integrated telephony functions, such as wearabl devices with minimal screen real - estate.

- Devices in this category include smartwatches and fitness wearables that can offer hands - free voice activation of a multitude of functions, through to smart televisions and games consoles.

Alexa Voice Service (AVS) Integration for AWS IoT

- Alexa Voice Service (AVS) Integration for AWS IoT is a new feature that cost - effectively brings Alexa Voice to any connected device without incurring messaging costs.
- AVS for AWS IoT has three components :
  1. A set of reserved MQTT topics to transfer audio messages between Alexa enabled devices and AVS.
  2. A virtual Alexa enabled device in the cloud that shifts tasks related to media retrieval, audio decoding, audio mixing, and state management from the physical device to the virtual device.
  3. A set of APIs that support receiving and sending messages over the reserved topics, interfacing with the device microphone and speaker, and managing device state.

❑❑❑