Unit III: WSN & Cloud Computing

Contents:

WSN: introduction to WSN technology, Basic components of WSN, Characteristic features of WSNs, challenges, Application of WSN in: smart homes, healthcare, intelligent transportation, agriculture, etc

Cloud Computing: Cloud architecture standards and interoperability, Business concerns in the cloud, characteristics, Cloud types; IaaS, PaaS, SaaS, Public cloud, Private cloud, Benefits and challenges of cloud computing, Development environments for service development: Amazon, Azure, Thing speak, Google App-cloud platform in industry (Features and services provided).

Wireless Sensor Networks (WSNs)

Introduction

Wireless Sensor Networks (WSNs) can be defined as a self-configured and infrastructure less wireless networks to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants and to cooperatively pass their data through the network to a main location or sink where the data can be observed and analyzed. A sink or base station acts like an interface between users and the network. One can retrieve required information from the network by injecting queries and gathering results from the sink. Typically a wireless sensor network contains hundreds of thousands of sensor nodes. The sensor nodes can communicate among themselves using radio signals. A wireless sensor node is equipped with sensing and computing devices, radio transceivers and power components. The individual nodes in a wireless sensor network (WSN) are inherently resource constrained: they have limited processing speed, storage capacity, and communication bandwidth. After the sensor nodes are deployed, they are responsible for self-organizing an appropriate network infrastructure often with multi-hop communication with them. Then the onboard sensors start collecting information of interest. Wireless sensor devices also respond to queries sent from a ―control site‖ to perform specific instructions or provide sensing samples.

Definition of WSN: A Wireless Sensor Network is a self-configuring network of small sensor nodes communicating among themselves using radio signals, and deployed in quantity to sense, monitor and understand the physical world.

What is a Wireless Sensor Network?

A Wireless Sensor Network is one kind of wireless network which includes a large number of circulating, self-directed, minute, low powered devices named sensor nodes called motes. These networks certainly cover a huge number of spatially distributed, little, battery-operated, embedded devices that are networked to caringly collect, process, and transfer data to the operators, and it has controlled the capabilities of computing & processing. Nodes are the tiny computers, which work jointly to form the networks.

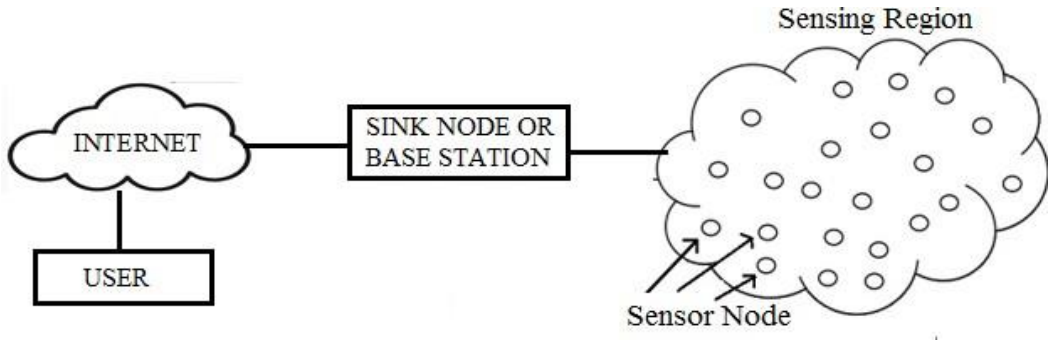


Figure 1: A typical Wireless Sensor Network

A wireless sensor network (WSN) consists of three main components: nodes, gateways, and software. The spatially distributed measurement nodes interface with sensors to monitor assets or their environment. The acquired data is wirelessly transmitted to the gateway, which provides a connection to the wired world where you can collect, process, analyze, and present your measurement data using software.

Definition - What does Wireless Sensor Network (WSN) mean?

Wireless sensor network (WSN) refers to a group of spatially dispersed and dedicated sensors for monitoring and recording the physical conditions of the environment and organizing the collected data at a central location. WSNs measure environmental conditions like temperature, sound, pollution levels, humidity, wind speed and direction, pressure, etc.

What are Wireless Sensor Networks?

Wireless sensor network (WSN) is a collective term to specify a rather independent set of tiny computers with the main target of sensing some physical property of their environment such as vibration, humidity, or temperature. They consist of a few to thousands of *sensor nodes*, often also referred to as nodes or sensors, which are connected to each other via wireless communications. Typically, there is also at least one special node, called the *sink* or the *base station*, which connects the sensor network to the outside world.

History of WSN

To understand the tradeoffs in today‘s WSNs, it is helpful to briefly examine their history. Like many advanced technologies, the origin of WSNs can be seen in military and heavy industrial applications, far removed from the light industrial and consumer WSN applications that are widespread today. The first wireless network that bore any real resemblance to a modern WSN is the Sound Surveillance System (SOSUS), developed by the United States Military in the 1950s to detect and track Soviet submarines. This network used submerged acoustic sensors – hydrophones – distributed in the Atlantic and Pacific oceans. This sensing technology is still in service today, albeit serving more peaceful functions of monitoring undersea wildlife and volcanic activity.

Echoing the investments made in the 1960s and 1970s to develop the hardware for today‘s Internet, the United States Defense Advanced Research Projects Agency (DARPA) started the Distributed Sensor Network (DSN) program in 1980 to formally explore the challenges in implementing distributed/wireless sensor networks. With the birth of DSN and its progression

into academia through partnering universities such as Carnegie Mellon University and the Massachusetts Institute of Technology Lincoln Labs, WSN technology soon found a home in academia and civilian scientific research.

Governments and universities eventually began using WSNs in applications such as air quality monitoring, forest fire detection, natural disaster prevention, weather stations and structural monitoring. Then as engineering students made their way into the corporate world of technology giants of the day, such as IBM and Bell Labs, they began promoting the use of WSNs in heavy industrial applications such as power distribution, waste-water treatment and specialized factory automation.

While the market demand for WSNs was strong, moving beyond these limited applications proved to be a challenge. The military, science/technology and heavy industrial applications of previous decades were all based on bulky, expensive sensors and proprietary networking protocols. These WSNs placed a premium on functionality and performance, while other factors such as hardware and deployment costs, networking standards, power consumption and scalability fell to the wayside. The combination of high cost and low volume prevented the widespread adoption and deployment of WSNs into a broader range of applications.

WSN Technology Transitions:

Although the technology for large-volume industrial and consumer applications did not exist in the 20th century, both academia and industry recognized the potential for such networks and formed joint efforts to solve the engineering challenges. Examples of these academic/industrial initiatives include:

- UCLA Wireless Integrated Network Sensors (1993)
- University of California at Berkeley PicoRadio program (1999)
- μ Adaptive Multi-domain Power Aware Sensors program at MIT (2000)
- NASA Sensor Webs (2001)
- ZigBee Alliance (2002)
- Center for Embedded Network Sensing (2002).

The goal of many of these initiatives and standards organizations is to enable high-volume deployment of WSNs in light industrial and consumer applications by reducing the cost and energy per sensor, while simplifying development and maintenance tasks.

Characteristics of a WSN

The main characteristics of a WSN include

- Power consumption constrains for nodes using batteries or energy harvesting
- Ability to cope with node failures
- Mobility of nodes
- Dynamic network topology
- Communication failures
- Heterogeneity of nodes
- Scalability to large scale of deployment
- Ability to withstand harsh environmental conditions
- Unattended operation
- Power consumption

- Ease of use
- Cross-layer design

Basic components of WSN/Architecture of WSN

When a large number of sensor nodes are deployed in a large area to co-operatively monitor a physical environment, the networking of these sensor node is equally important. A sensor node in a WSN not only communicates with other sensor nodes but also with a Base Station (BS) using wireless communication.

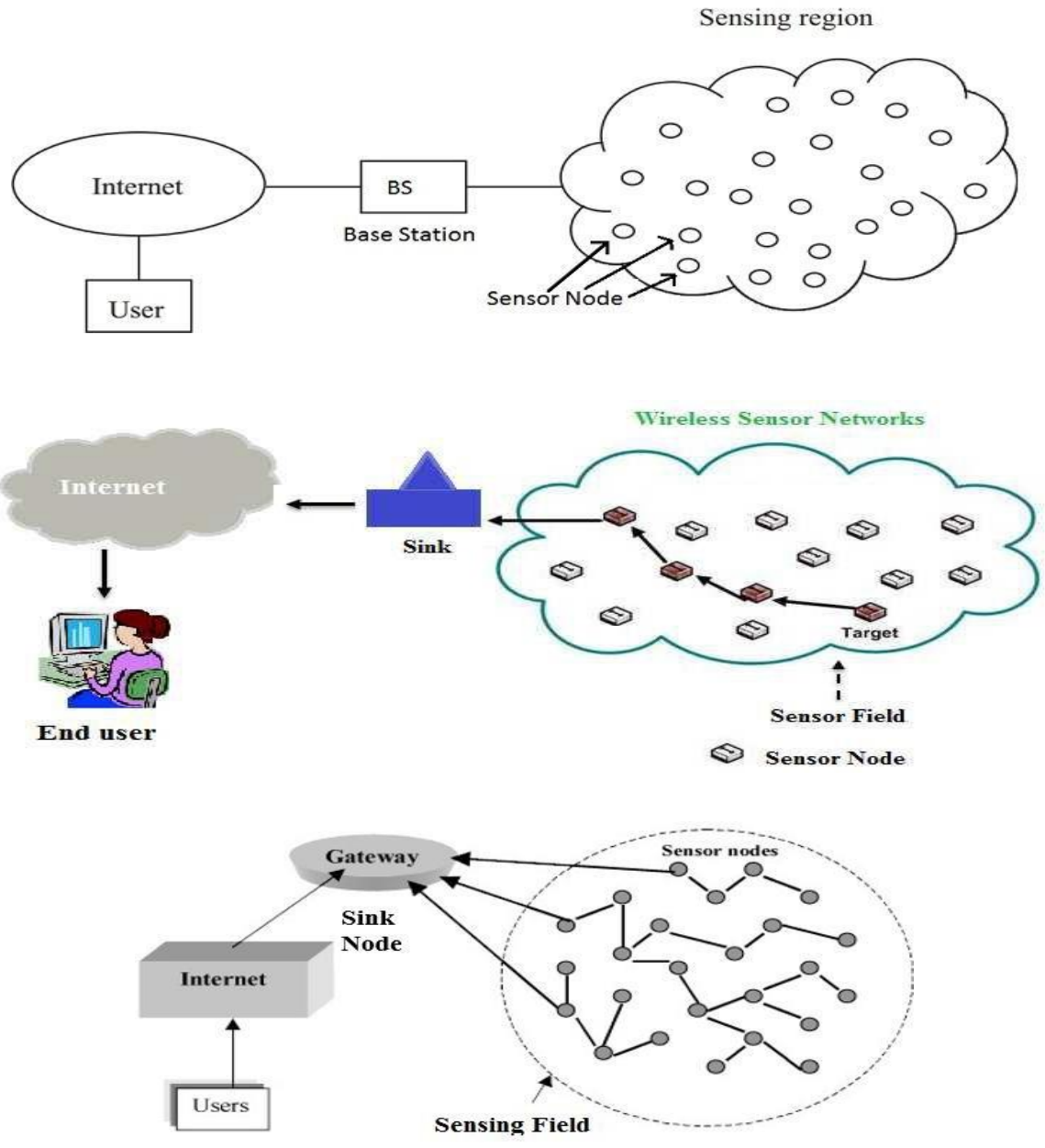


Figure: Basic components of WSN/Architecture of WSN

The base station sends commands to the sensor nodes and the sensor node perform the task by collaborating with each other. After collecting the necessary data, the sensor nodes send the data back to the base station.

A base station also acts as a gateway to other networks through the internet. After receiving the data from the sensor nodes, a base station performs simple data processing and sends the updated information to the user using internet.

If each sensor node is connected to the base station, it is known as Single-hop network architecture. Although long distance transmission is possible, the energy consumption for communication will be significantly higher than data collection and computation.

A WSN is a network consisting of numerous sensor nodes with sensing, wireless communications and computing capabilities. These sensor nodes are scattered in an unattended environment (i.e. sensing field) to sense the physical world. The sensed data can be collected by a few sink nodes which have accesses to infrastructure networks like the Internet. Finally, an end user can remotely fetch the sensed data by accessing infrastructure networks. Fig. shows the operation sketch map of WSNs. In Fig. , two kinds of network topologies are shown. The sensor nodes either form a flat network topology where sensor nodes also act as routers and transfer data to a sink through multi-hop routing, or a hierarchical network topology where more powerful fixed or mobile relays are used to collect and route the sensor data to a sink.

Three main components of WSN

1. Base station (Gateway)
2. Sensor node
3. Software

As shown in the WSN architecture, WSN will have three main components as mentioned above viz. distributed sensor nodes, sink nodes (Base station or Gateway) and Software. Each sensor node usually will have single omnidirectional antenna and transceiver, power supply and DSP embedded. Gateway is located at the boundary of the monitoring coverage area. The gateway helps connect micro-sensor network with the outside world such as internet or similar other WSN.

Performance metrics in WSN :

The design of SN is a challenge because many influencing factors such as fault tolerance, scalability, production cost, operating environment, network topology, hardware constraints, transmission media, power consumption and others have to be considered. The performance of the network is then measured based on quantifiable parameters called performance metrics. ***Network Lifetime****:* Network lifetime is defined as the number of data aggregation rounds till x % of sensors die where x is specified by the system designer. For instance, in applications where the time that all nodes operate together is vital, lifetime is defined as the number of rounds until the first sensor is drained of its energy.

Data accuracy: The definition of data accuracy depends on the specific application for which the sensor network is designed. For instance, in a target localization problem, the estimate of target location at the sink determines the data accuracy.

Latency: Latency is defined as the delay involved in data transmission, routing and data aggregation. It can be measured as the time delay between the data packets received at the sink and the data generated at the source nodes.

Energy Efficiency*:* The functionality of the sensor network should be extended as long as possible. In an ideal data aggregation scheme, each sensor should have expended the same amount of energy in each data gathering round. A data aggregation scheme is energy efficient if it maximizes the functionality of the network. If we assume that all sensors are equally important, we should minimize the energy consumption of each sensor. This idea is captured by the network lifetime which quantifies the energy efficiency of the network.

Bandwidth*, **Capacity** and **Throughput****:* These indicate the capacity of data which can be sent over a link within a given time, however since the data size is very small bandwidth rarely matters.

Hop Count: No of hop in communication determine the cost of path, and eventually the energy consumed in the process.

Signal Strength: SNR as an indication for the link quality and the distance between two nodes is helpful to compute and determine the nodes and their reach ability during the communication process.

Design Issues/Challenges in Wireless Sensor Networks:

In order to design good applications for wireless micro-sensor networks, it is essential to understand factors important to the sensor network applications. Although WSNs share some commonalities with existing wireless ad-hoc networks they pose a number of technical challenges different from traditional wireless ad-hoc networks. The protocols and algorithms that have been proposed for traditional wireless ad-hoc networks are therefore not well suited for the application requirements of the sensor networks. To illustrate this point, differences between sensor networks and traditional networks are outlined below:

i. Energy

The sensor nodes are generally inaccessible after deployment and normally they have a finite source of energy that must be optimally used for processing and communication to extend their lifetime. It is a well-known fact that communication requires significant energy. In order to make optimal use of energy, therefore communication should be minimized as much as possible.

ii. Redundancy

Due to the frequent node failures and inaccessibility of failed nodes, WSNs are required to have high redundancy of nodes so that the failure of few nodes can be negligible.

iii. System lifetime

The WSNs should function as long as possible. Their system lifetime can be measured by using generic parameters such as time until the nodes die or by using application specific parameters like time until the sensor network is no longer providing acceptable quality results.

iv. Scalability

In WSNs, each sensor node obtains a specific view of the environment. A given sensor's view of the environment is limited both in range and accuracy; it can only cover a limited physical area of the environment. The WSNs therefore, deploys sensor nodes that have a short transmission distance in large numbers to monitor the entire area.

v. Adaptability

The WSN system should be adaptable to changes such as addition of more nodes, failure of nodes, environmental conditions and thus unlike traditional networks, where the focus is on maximizing channel throughput or minimizing node deployment, the major consideration in a sensor network is to extend the network lifetime besides system robustness.

vi. Application awareness

A WSN is not a general purpose network. In order to deploy it for specific application, the WSN protocols should consider application-specific trade-offs in terms of complexity, resource usage and communication patterns to improve network efficiency.

vii. Lack of global identification

Due to large number of sensor nodes in a sensor network the global identification (GID) is generally not possible. Although in some cases, the Global Positioning System (GPS) provides positioning information to sensor nodes but it requires line of sight to several satellites, which is generally not available inside of buildings, beneath dense foliage, underwater, when jammed by an enemy or during MARS exploration etc.

viii. Storage, search and retrieval

The sensor network can produce a large volume of raw data such as continuous time series of observations over all points in space covered by the network. Since the data source is continuous traditional databases are not suitable for WSNs.

ix. Data centric processing

The naming schemes in WSNs are often data-oriented for example an environmental monitoring system may requests temperature readings through a query like "collect temperature readings in the region bounded by the rectangle (x1,y1,x2,y2)", instead of a query "collect temperature readings from a set of nodes having addresses x, y and z."

x. Production cost

The cost of a single node is very important to justify overall cost of the network; since the sensor networks consist of a large number of sensor nodes therefore cost of each sensor node has to be kept low.

xi. Node deployment

Node deployment is application dependent and affects performance of the protocol. The deployment is either deterministic or self-organizing. In deterministic situations, the sensors are manually placed and data is routed through pre-determined paths. However, in self organizing systems, the sensor nodes are scattered randomly creating an infrastructure in an ad-hoc manner.

xii. In-network processing

In general transport protocols used in wired and wireless networks have assumed end-to-end approach guaranteeing that data from the senders have not been modified by intermediate nodes until it reaches a receiver. However, in WSNs data can be modified or aggregated by intermediate nodes in order to remove redundancy of information. The previous solutions did not accommodate concept of in-network processing, called data aggregation or diffusion in WSNs.

xiii. Latency

Latency refers to delay from when a sender sends a packet until the packet is successfully received by the receiver. The sensor data has a temporal time interval in which it is valid, since the nature of the environment changes constantly, it is therefore important to receive the data in a timely manner.

xiv. Fault tolerance

Sensor nodes are fragile and they may fail due to depletion of batteries or destruction by an external event. Realizing a fault-tolerant operation is critical, for successful working of the WSN, since faulty components in a network leads to reduced throughput, thereby decreasing efficiency and performance of the network.

Advantages of the wireless sensor network

- **Flexible:** WSN is a flexible network and can adapt to the changes.
- **Additional of New Device**: WSN can accommodate new devices in the network any time with ease.
- **Save Cost:** Wireless sensor networks save a lot of wiring cost and sensors like PIR detectors are relatively cheaper then wires.
- **Useful to society:** Wireless sensor network are used in different fields like healthcare, defense, environment monitoring which is very beneficial to human welfare.
- Wireless sensor network can be deployed on a large scale.
- Wireless sensor networks can be developed according to the application.

- Wireless sensor networks can be /monitored or accessed with remote location.
- Wireless sensor networks can be easily implemented.
- Wireless sensor networks can be designed with 3 different topologies (Star, Tree & Mesh).
- Wireless sensor networks are made up of low cost, low power sensor nodes.
- Wireless sensor networks can accommodate with multiple node at a time.
- Wireless sensor networks supports to embedded systems (like CPLD / FPGA).
- It avoid a lot of wiring
- It can accommodate new devices at any time
- Its flexible to go through physical partitions
- It can be accessed through a centralized monitor

Disadvantages of wireless sensor networks

- Less secure because hackers can enter the access point and obtain all the information.
- Lower speed as compared to a wired network.
- More complicated to configure compared to a wired network.
- Easily troubled by surroundings (walls, microwave, large distances due to signal attenuation, etc).
- It is easy for hackers to hack it we couldn‘t control propagation of waves.
- Comparatively low speed of communication.
- Gets distracted by various elements like Bluetooth.
- **Security:** WSN networks are not secure as compared to wired networks. Hackers can easily hack the network.
- **Battery Issue:** Nodes need to be charged at regular intervals. Battery life of the nodes is very low.
- **Low Communication Speed:** Communication speed is comparatively low than the wired network.
- **Distraction**: Wireless sensor networks keep distracting by other wireless devices.
- Its damn easy for hackers to hack it as we cant control propagation of waves
- Comparatively low speed of communication
- Gets distracted by various elements like Blue-tooth
- Still Costly at large

Applications of sensor networks

- Physical security for military operations
- Indoor/Outdoor Environmental monitoring
- Seismic and structural monitoring
- Industrial automation
- Bio-medical applications
- Health and Wellness Monitoring
- Inventory Location Awareness
- Future consumer applications, including smart homes.
- Military Applications
- Health Applications
- Environmental Applications
- Home Applications
- Commercial Applications
- Area monitoring
- Health care monitoring
- Environmental/Earth sensings
- Air pollution monitoring
- Landslide detection
- Water quality monitoring
- Industrial monitoring

Requirements for WSNs

- **Fault tolerance**
  The network functionality must be maintained even though the built-in dynamic nature and failures of nodes due to harsh environment, depletion of batteries, or external interference make networks prone to errors.
- **Lifetime**
  The nodes are battery powered or the energy is scavenged from the environment and their maintenance is difficult. Thus, energy saving and load balancing must be taken into account in the design and implementation of WSN platforms, protocols, and applications.
- **Scalability**
  The number of nodes in WSN is typically high. Thus, the WSN protocols must deal with high densities and numbers of nodes.

- **Real time**
  WSNs are tightly related to the real world. Therefore, strict timing constraints for sensing, processing, and communication are present in WSNs.
- **Security**
  The need for security in WSNs is evident, especially in health care, security, and military applications. Most of the applications relay data that contain private or confidential information.
- **Production cost**
  The number of nodes in WSNs is high, and once nodes run out of batteries they are replaced by new ones. Further, WSNs are envisioned to be everywhere. Therefore, to make the deployments possible, the nodes should be extremely low cost.

1. Applications of WSN in Agricultural Environment Monitoring Systems

The Wireless Sensors Network (WSN) is nowadays widely used to build decision support systems to solve many real-world problems. One of the most interesting fields having an increasing need of decision support systems is agricultural environment monitoring. Agricultural environment monitoring has become an important field of control and protection, providing real-time system and control communication with the physical world. An intelligent and smart WSN system can collect and process large amount of data from the beginning of the monitoring and manage air quality, soil conditions, to weather situations. The proposed system collects and monitors information related to the growth environment of crops outside and inside greenhouses using WSN sensors and CCTV cameras. The temperature and humidity sensors are developed in-house and both sensors are very reliable. Furthermore, the system allows automatic control of greenhouse environment remotely and thus improves the productivity of crops. This paper presents hardware architecture, system architecture and software process control of the agriculture environment monitoring system.

Figure 1 shows a general WSN node architecture. Each sensor network node consists of a sensor module, processor module, communication module and power module using battery

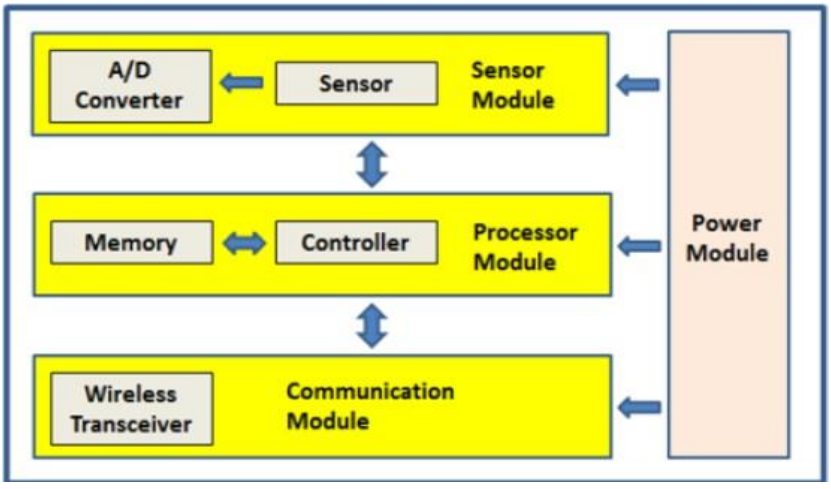


Fig. 1. WSN Node Architecture

Agricultural Environment Monitoring System Architecture

The agricultural environment monitoring system developed in this project measure and analyze external weather conditions such as temperature, humidity, rain fall, wind direction, wind speed and also internal weather conditions in a greenhouse such as temperature, pH, relative humidity, moisture, CO2, EC, and the intensity of light. The system also collects image information on the indoors and outdoors through CCTV cameras. The information collected in real-time is then converted and stored into a database through a web server, which provides critical information to users.

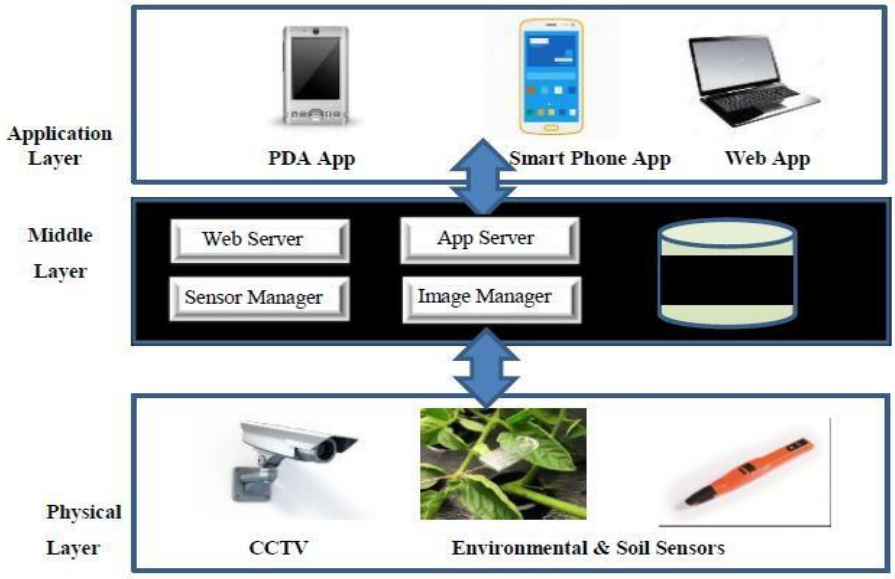


Fig. 4. Agricultural Environment Monitoring System Architecture

The system architecture of the Agricultural Environment Monitoring System consists of three layers as shown in Figure 4. The Physical Layer represents the physical sensors, actuators and CCTV cameras. The Middle Layer represents the WSN interface, CCTV interface, sensor and image manager, database, App server and Web server. Finally, the Application Layer represents the Graphical User Interface (GUI), monitoring services (sensor and image monitoring), alerting services and decision making services.

a. Physical Layer

The physical layer consists of wireless environmental, soil sensors (temperature, humidity, moisture, EC, pH and CO2), actuators and CCTV cameras. These sensors will collect environmental and soil information from indoor (e.g. greenhouses) and outdoor environment periodically and sent the data to sensor manager.

The measurement data include temperature, humidity, soil and water pH, soil moisture, rainfall quantity, EC, wind speed and direction, water level, water consumption, etc. The data is further analysed and converted into appropriate formats and finally sent to Middle Layer for storing in databases.

b. Middle Layer

The middle layer (ML) supports communication between physical layer and application layer. The ML converts the data received from the physical layer and store it in a database. The ML will also support the data request from the application layer. The Web server and App server allows users to monitor data processed by the components through a Web browser, PDA or smart phone at any time. The sensor manager converts and analyses this data and store it in a database. The web server will fetch the sensing data stored in the database at a fixed interval and send it to the GUI. The users can view the sensor data through the GUI in the Application Layer. The GUI will also display and send emails and SMS if there is any alerts and warnings. The CCTV cameras will collect real-time image information (real-time streaming) on the indoors and outdoors and send it to image manager. The image manager will store it in the database. The user can access the image data at anytime and anywhere through the GUI using PDAs or Smart Phones. The web server used to host sites and deliver web pages content to clients who can access them through the Internet. Another use of web server is to connect any component of the physical layer to browse any monitoring or actuating device in real time.

c. Application Layer

The application layer consists of various GUIs to provide suitable information and support user requests. The users can access agricultural environment information and status through various platforms such as smart phone, web and PDA. This layer consists of monitoring support, actuating support, alerting support and decision making support.

2. Application of WSN for Smart Homes

In smart home system, the current commonly used three kinds of short-range wireless communication technologies are: ZigBee, Bluetooth and Wi-Fi. These three kinds of short range wireless communication technology are compared in Table 1, in which the number of network nodes, the power consumption, the transmission rate, the transmission distance, frequency, device activation time and other technical indicators have been considered.

In the interpretation of network, smart home system is usually divided into three parts: external network, home gateway and internal network. Internal network is recognized by ZigBee nodes, which support user management platform in getting state of appliances and convey the commands from the user organization platform to appliances via ZigBee network. And the users outdoor or in remote areas need to attain the family network monitoring and management through an external network, using a computer, web or mobile phone, which enables the application surface of the system to be broaden. Home gateway plays the role of a manager for the internal network, which can control the nodes in home. For the external network, home gateway works as a protocol converter. The overall framework of smart home system is showed in Fig.

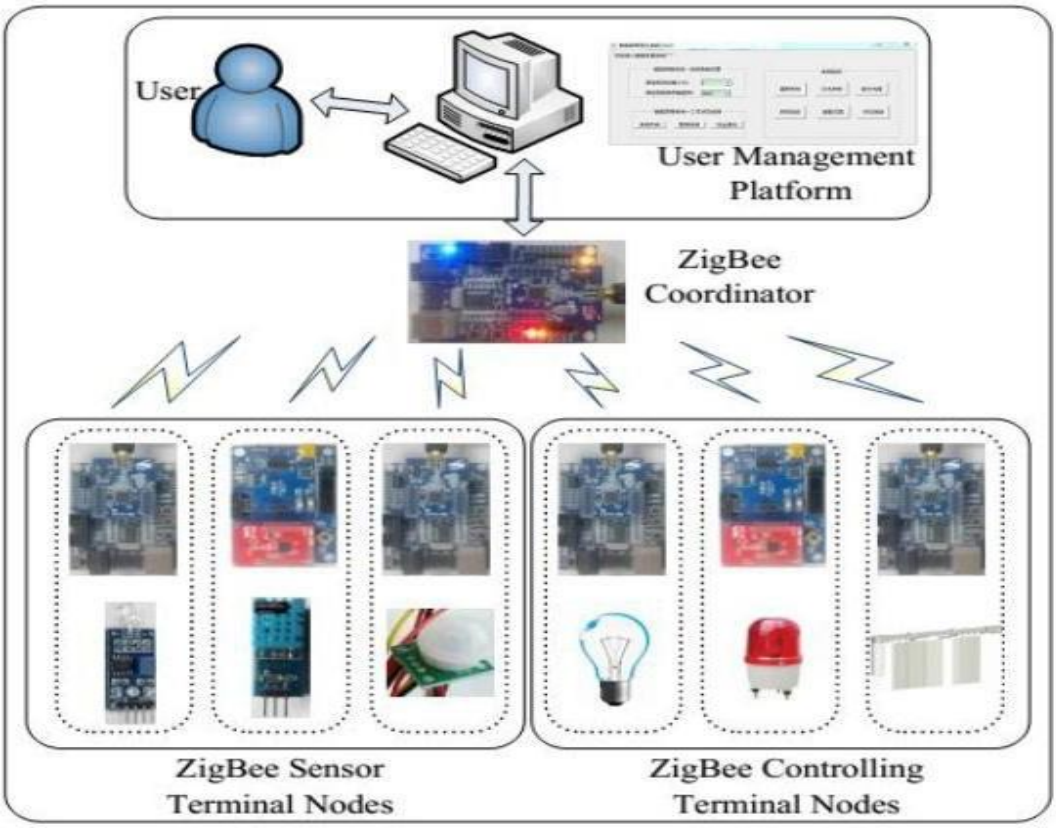


Figure: Application of WSN for Smart Homes

Design of Smart Home System

The smart home system intended in this paper uses the star network topology and emphases on the internal network, which mainly comprises the user management platform, ZigBee coordinator and ZigBee terminal nodes, as showed in Fig. Due to the smart home system based on ZigBee technology is reinforced by computer technology, sensor network technology and control technology, so the design of smart home can be divided into three blocks.

1) Smart home user management platform: Smart home user management platform is achieved on PC, which realizes the home network management and controlling through the simple man-machine interface.

2) Smart home sensor network: The sensor network is based on the ZigBee protocol stack. Users can control wireless nodes distributed indoor. The sensor network is primarily composed by the ZigBee coordinator, ZigBee sensor terminal nodes and ZigBee controlling terminal nodes.

3) Smart home control technology: CC2530 is the core chip of each node in this system. The control technology mainly contains the process of data collecting by ZigBee sensor terminal nodes and the management of the controlling terminal nodes by the coordinator.

Hardware Design

The design of a smart home system based on WSN mainly involves three parts - sensors acquiring data, wireless transceiver chips sending or receiving data and commands, and the master chip processing data. This system is mainly composed of two parts - master control and sensor terminals. The master control terminal is used to receive and analyse the data of the

sensor terminals, and the sensor terminal is used to acquire and transmit data. The functions of the master control are realised by the master control module and the wireless transceiver module, and the functions of sensor terminals are completed by the wireless transceiver module and the smart sensor module. The hardware design of the smart home system is shown in Figure

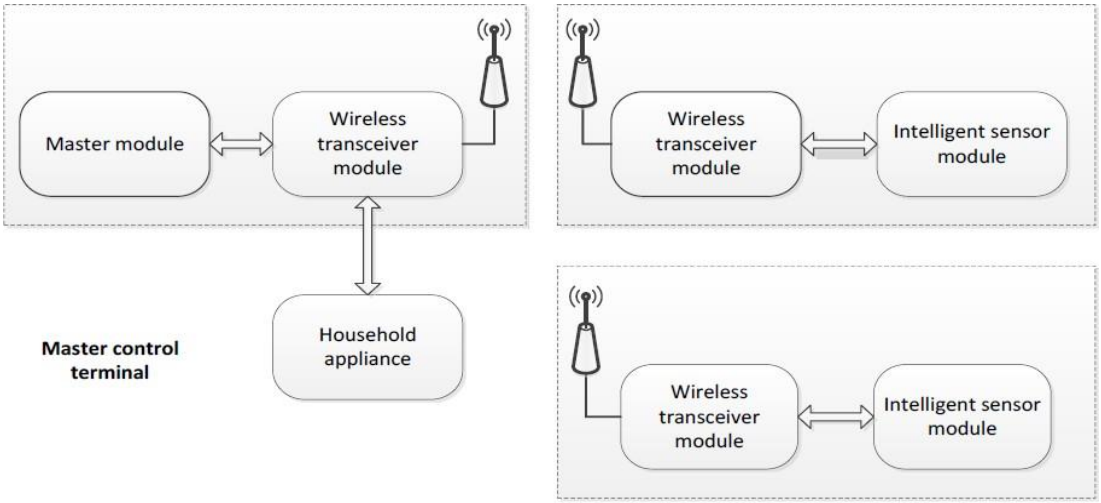


Fig. . Block diagram of the smart home system hardware

Master control design

The master control consists of the master control module and the wireless transceiver module. It is designed to receive data from each sensor terminal. After data analysis, the control of home appliances is completed. The master control unit can decide whether to add or delete other sensor terminals, and can also send, receive and analyse the processing instructions to or from the sensor terminals. The master control module coordinates other modules, so that the whole system can run in an orderly fashion, and also to analyse and process the acquired data. It can also connect with the computer, write the programme into the MCU with the help of the development environment, display the data and information acquired by sensors, and select the functions of the system. The data processing chip of the master control module is STM32.

Wireless transceiver module design

The wireless transceiver module is composed of a wireless transceiver chip and antenna, etc. It is designed to carry out data transmission with a single chip microcomputer, receive and transmit the commands of the single chip computer, send the commands to other sensor terminals, and acquire sensor terminal data. This module uses Nordic‘s monolithic RF transceiver n RF905 as the wireless transceiver chip to receive and transmit data or commands.

Sensor terminal design

The sensor terminal is composed of an intelligent sensor module and a wireless transceiver module. The sensor terminal is used for data acquisition and transmission. DS18B20 is used as the temperature sensor to collect temperature data. The MCU sends the collected data and displays them on the LCD screen. The humidity sensor HS1101 is used to collect humidity data. The MCU sends the collected data and displays them on the LCD screen.

3. Healthcare Monitoring System Using Wireless Sensor Network

At present day to make human life more comfortable Wireless Sensor Network (WSN) are used to understand the critical conditions of human body. It is smallest unit which have unique features. The wireless sensor supports reliability, mobility etc. The body sensor network helps to people providing healthcare services like medical data access, medical monitoring and communication with physician in emergency situations through SMS or GPRS. It also provides fully remote method to acquire and detect and monitor the physiological signals without any interruption in patient‘s normal life. The wireless sensor network improves life quality.

An important component of ubiquitous healthcare is wireless sensor network (WSN). WSNs are an emerging technology that is poised to transform healthcare. The WSNs promise to make life more comfortable by significantly improving and expanding the quality of care across a wide variety of settings and segments of the population.

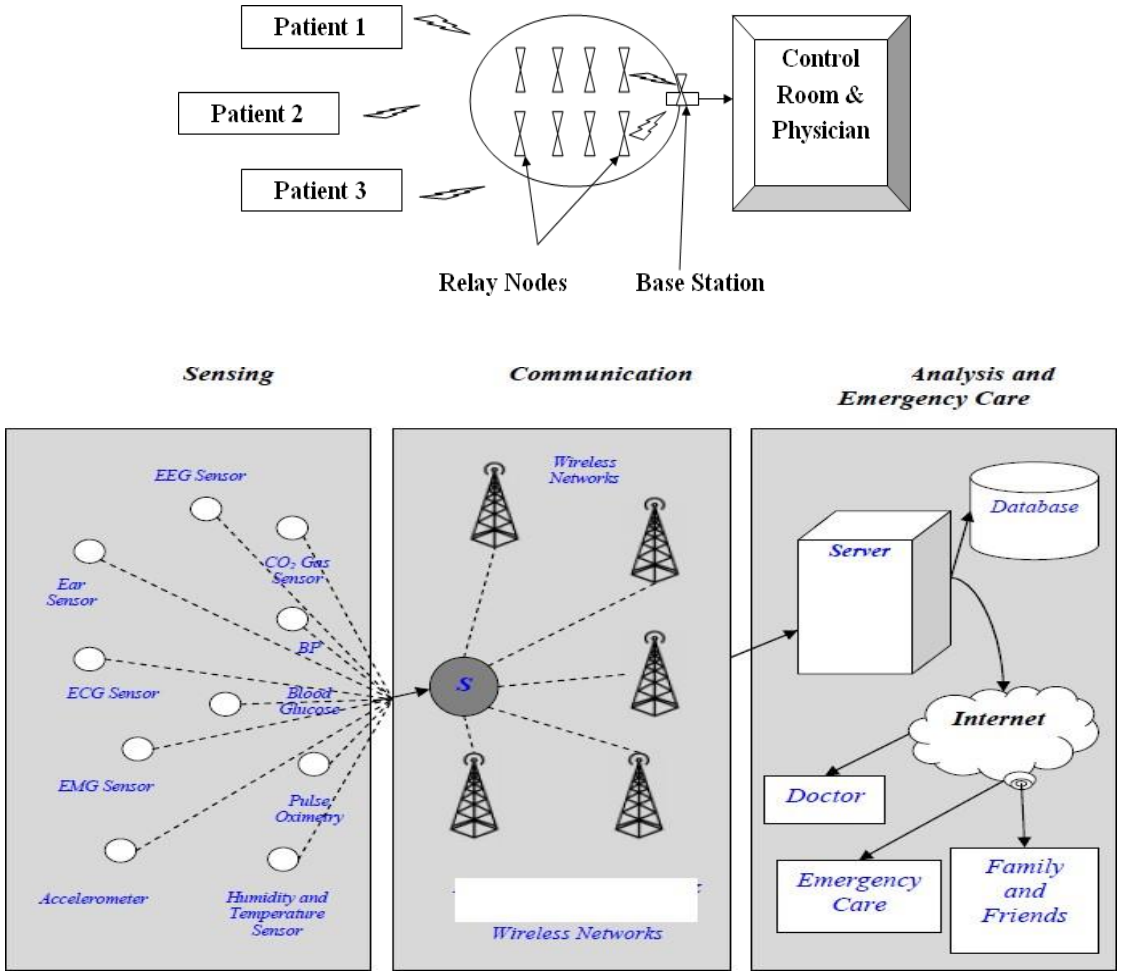


Fig. Architecture for healthcare system using WSN

Architecture for smart Healthcare using Wireless Sensor Network (WSN) is as shown in above figure. The system monitors the physiological parameters such as Blood Pressure (BP), ECG, Body Temperature and Respiration etc. The coordinator node has attached on body of patients for collecting the signal from wireless sensors. The wireless sensors send this signal to base station or control room of physician. This wireless sensors form wireless body sensor network (WBSN). Node of each WSN composed of health care sensors and RF tranreceiver which send data to back end sever. Sensors can choose in the range of WSNs, while RF tranreceiver is implemented as a coordinator which manages WSN other than forwards data. The sensing data of each patient are stored in back-end server with each having its own ID. The data analysis, database inquiry, data manning and the system management are processed on the web page of server. The system can detect abnormal condition of patients and send the SMS or e-mail to the physician. It advantageous to patient and associate relative of patient and others who may use the continuous remote health monitoring. By using wireless sensor network this system improves the quality of medical healthcare system.

Advantages of mobile WSN to Healthcare Monitoring System

The advantages of integrating WSN to healthcare monitoring system are as follows.
WSN in healthcare has the following benefits

- *Flexibility:* The system collects and communicates data wirelessly with minimal input from the patient. It is not required that the patient be limited to his bed.
- *Always-on*: The physiological and environment data can be monitored continuously allowing for real-time response by caregivers. The WSNs allow patients to be monitored and remain always under medical control.
- *Self-organization*: Physicians can change the mission of the network as medical needs change. *Low-cost:* Using WSNs in healthcare provides a low-cost communication infrastructure that is suitable for monitoring.

Due to these advantages, the mobile WSN are used to various healthcare applications as listed in the next section.

Applications of Healthcare Monitoring System

The applications of healthcare monitoring include

- Day-to-day activity monitoring applications
- Fall and movement detection applications
- Location tracking applications
- Medication intake monitoring applications

4. Intelligent transportation System Using Wireless Sensor Network

Intelligent Transportation Systems (ITS) aims to improve transport safety and mobility, as well as to increase people‘s productivity and reducing the harmful effects of traffic. This improvement is achieved through the integration of communication technologies in vehicles and infrastructure. ITS is not only proposed to improve vehicle traffic conditions but also intends to make the transportation safer, more sustainable, and efficient, avoiding the inconvenience caused by traffic congestion and the effects of climate problems on traffic. To this end, the focus is on improving the management of cities‘ resources and increasing the convenience of people using information and alert services. This improvement, therefore, helps to ease the flow in the city, reducing the time spent on congestion and consequently reducing fuel consumption, CO2 emissions and monetary losses. In the following sections, the central concepts related to ITS will be presented.

As the city road networks is growing day-by-day, the question of how to obtain information about the roads is becoming more and more challenging. In such an era Intelligent Transportation Systems (ITSs) has emerged as a key candidate that is benefited from the unique features and capabilities of Wireless Sensor Networks (WSNs) and Bluetooth technology. WSNs are composed of small tiny devices that work in autonomous manner to sense the surroundings. The Bluetooth protocol can be used for inter-vehicle communication among vehicles equipped with Bluetooth devices. ITSs are a state of art combination of transportation infrastructure and computer and information technology. ITSs can also resolve severe situations like traffic congestion and cope with emergency conditions like major accidents. The architecture that will increase the safety of road travel using the concepts of WSN and the Bluetooth protocol.

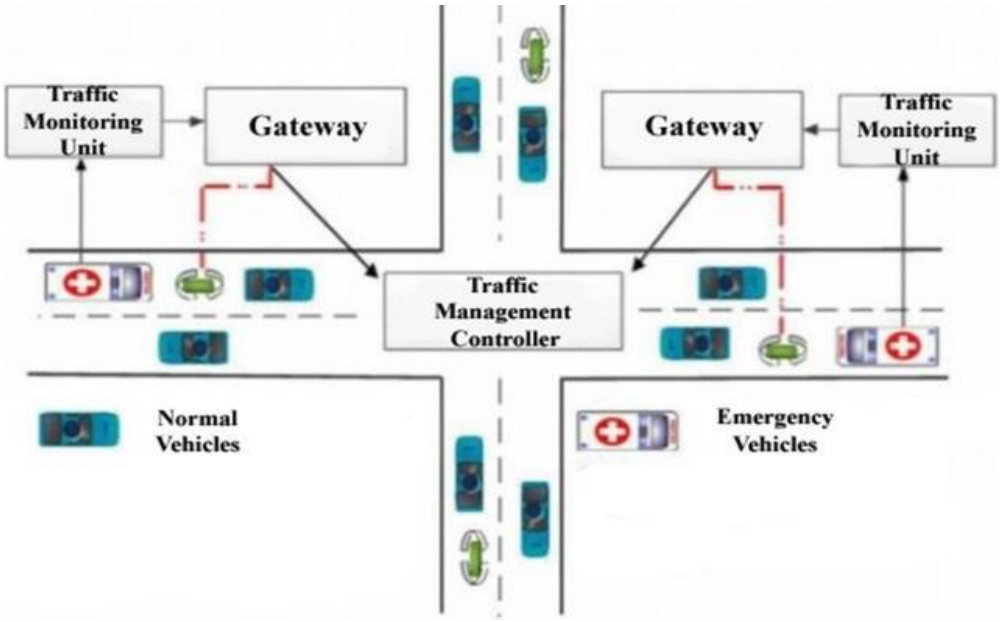


Fig. Intelligent Transportation system Using WSN

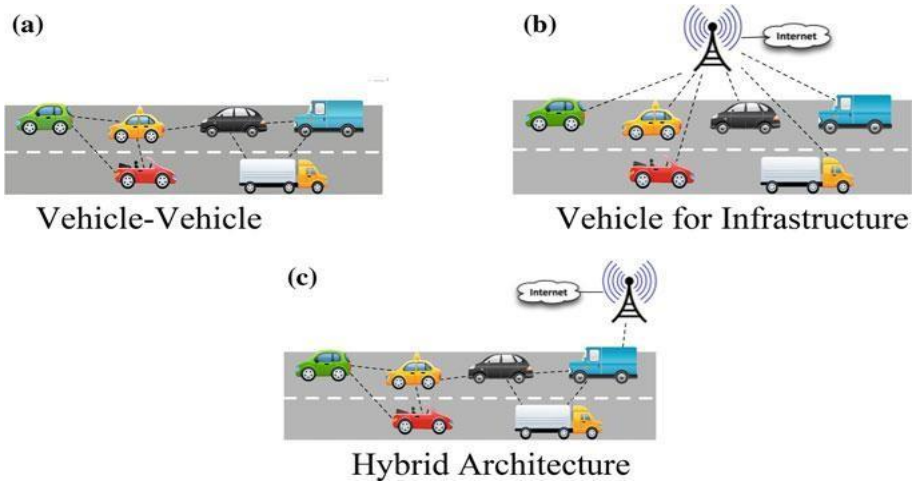
Low power standards like ZigBee are not suitable for ITS because vehicles are running in speed. Sensor equipped in vehicles does not suffer from problem of power as they can be charged by

vehicle battery. As described base stations are deployed at important places only, they get information from the vehicles running on the road. They will optimize information by efficient data aggregation and serve it to the vehicle in need. ITS is formed with the help of WSN and ad-hoc network formed by Bluetooth devices attached to vehicles.

If we consider the scenario shown in Figure, base stations will have the information about road conditions that will previously provided by passing vehicles. In this way, cars A, B and E can get information from the one BS and store some information to the other BS at the same time. BS are supposed to monitor up to 200 vehicles and sense 100 m. BS are powered with solar power. In the following subsection we describe proposed algorithm. Vehicles running on the road first search for the base station, if it find the BS it communicate with it otherwise it seeks the vehicles equipped with Bluetooth devices and communicate. In this way, there are two types of communications to be considered between WSN-BS and vehicles running on the roads as shown in above Figure

Vehicular Networks to Intelligent Transportation Systems

Vehicles participating in the network are equipped with an onboard system with computing capability, communication interfaces, sensors, and user interfaces. The system supports a range of applications to improve transport security and also provide services to users.



- **Vehicle-to-Vehicle (V2V):** It allows a direct communication of vehicles without relying on fixed infrastructure support. In this type of communication, vehicles can exchange data of the conditions of the highway, detect the presence of other vehicles, and even information about vehicles in unsafe movement.
- **Infrastructure-to-Vehicle (V2I):** It allows a vehicle to communicate with the road infrastructure. In this way, the vehicle can receive from the road infrastructure information about obstacles and the presence of pedestrians, road conditions data, advertisements, and safety information.
- **Hybrid Architecture:** It combinesV2Vand V2I solutions. In this case, a vehicle can communicate with the road infrastructure in a single or multiple hops according to its location about the point of connection with the infrastructure for different purposes.

Cloud Computing

What is cloud Computing?

Simply put, cloud computing is the delivery of computing services—including servers, storage, databases, networking, software, analytics, and intelligence—over the Internet (‑the cloud‖) to offer faster innovation, flexible resources, and economies of scale. You typically pay only for cloud services you use, helping lower your operating costs, run your infrastructure more efficiently and scale as your business needs change.

Cloud computing is a growing idea in the world of IT, born out of the necessity for computing on the go. It brings the user access to data, applications and storage that are not stored on their computer. For a very simple cloud computing overview, it can be understood as a delivery system that delivers computing the same way a power grid delivers electricity. To the average computer user it offers the advantage of delivering IT without the user having to have an in depth knowledge of the technology. Similar to the way a consumer can access electricity without being an electrician.

What Is Cloud Computing?

Cloud computing is the delivery of different services through the Internet. These resources include tools and applications like data storage, servers, databases, networking, and software.

Rather than keeping files on a proprietary hard drive or local storage device, cloud-based storage makes it possible to save them to a remote database. As long as an electronic device has access to the web, it has access to the data and the software programs to run it.

Cloud computing is a popular option for people and businesses for a number of reasons including cost savings, increased productivity, speed and efficiency, performance, and security.

What is cloud computing?

Cloud computing is a general term for various hosted services delivered over the internet. It is the use of remote, rented servers to store and manage data, rather than the use of a local, privately maintained server.

Understanding Cloud Computing

Cloud computing is named as such because the information being accessed is found remotely in the cloud or a virtual space. Companies that provide cloud services enable users to store files and applications on remote servers and then access all the data via the Internet. This means the user is not required to be in a specific place to gain access to it, allowing the user to work remotely.

A simple definition of cloud computing involves delivering different types of services over the Internet. From software and analytics to secure and safe data storage and networking resources, everything can be delivered via the cloud.

Why Cloud Computing?

With increase in computer and Mobile user‘s, data storage has become a priority in all fields. Large and small scale businesses today thrive on their data & they spent a huge amount of

money to maintain this data. It requires a strong IT support and a storage hub. Not all businesses can afford high cost of in-house IT infrastructure and back up support services. For them Cloud Computing is a cheaper solution. Perhaps its efficiency in storing data, computation and less maintenance cost has succeeded to attract even bigger businesses as well.

Cloud computing decreases the hardware and software demand from the user‘s side. The only thing that user must be able to run is the cloud computing systems interface software, which can be as simple as Web browser, and the Cloud network takes care of the rest. We all have experienced cloud computing at some instant of time, some of the popular cloud services we have used or we are still using are mail services like gmail, hotmail or yahoo etc.

While accessing e-mail service our data is stored on cloud server and not on our computer. The technology and infrastructure behind the cloud is invisible. It is less important whether cloud services are based on HTTP, XML, Ruby, PHP or other specific technologies as far as it is user friendly and functional. An individual user can connect to cloud system from his/her own devices like desktop, laptop or mobile.

Cloud computing harnesses small business effectively having limited resources, it gives small businesses access to the technologies that previously were out of their reach. Cloud computing helps small businesses to convert their maintenance cost into profit. Let‘s see how?

In an in-house IT server, you have to pay a lot of attention and ensure that there are no flaws into the system so that it runs smoothly. And in case of any technical glitch you are completely responsible; it will seek a lot of attention, time and money for repair. Whereas, in cloud computing, the service provider takes the complete responsibility of the complication and the technical faults.

Benefits of cloud computing

Cloud computing offers plenty of benefits. The top 10 are as follows:

1. **Instant scalability:**

Cloud computing enables immediate scalability of infrastructure capacity depending on the business need. It is like having an unlimited IT resource, which can be scaled up and/ or down to meet user demands

2. **Anywhere accessibility:**

A lot of factors have enabled globalization, and a key one is technology and high-speed internet connectivity. Cloud empowers enterprises to deploy their applications across the globe so they can service their customers at a fraction of the cost of a traditional brick-and-motor businesses. Due to lower latency, customers around the world get an identical digital experience while using the applications

3. **Increased speed and operational agility:**

Today, to cope with competition, businesses must have the ability to instantly scale their cloud capacity by accessing bandwidth demands from remote servers of a particular cloud service provider. If the business demand is more, the enterprise can turn up its computing capacity and IT resources availability with the click of a button. Such an ability improves organizational agility, productivity and efficiency making scope to experiment with new ideas and thereby,

offering competitive advantage and the ability for the organization of any size to disrupt the market

4. **Reduced expenditure:**

With the cloud, enterprises can focus on building their business rather than investing in hardware infrastructure and data centers that either remain idle, or underutilized. Cloud costs, however, depend on the consumption -- a variable expense

5. **Automatic updates/ patches:**

When enterprises deal with several different kinds of software, operating systems, and applications from various vendors for their everyday operations, they have to have software and security updates rolled out from time to time. This is a very time-consuming process and the downtime for system maintenance means loss of productivity. A cloud service provider or a managed service provider can take care of these automatically, saving time and manual effort on maintenance.

6. **Disaster recovery:**

The world is going digital, making robust backup and disaster recovery crucial for business of all sizes. However, on-premises investments for disaster recovery are things of the past today. Especially since cloud computing helps both large corporations and small enterprises save time and effort involved in this exercise

7. **High security:**

Protecting sensitive, personally identifiable and/ or financial information is a considerable challenge for CIOs. Advanced cloud security features, however, have reduced the risks of information loss and cyber stealth

8. **Reduced carbon footprint:**

Last, but not the least, cloud infrastructure significantly reduces power, IT infrastructure, and resource consumption by offering resources as per demand, thereby reducing e-waste and adverse impact on the environmental

9. **Flexibility:**

From closed cabins to bringing your internet-enabled devices to work, irrespective of the device type and/ or global location, cloud offers vast flexibility and empowerment to businesses as well as to their employees

10. **Enterprise collaboration:**

Company information no longer exists in silos (except confidential ones, of course). Centralized documentation control on cloud-based, file-sharing and social communication apps (like Slack, Yammer, etc) offers transparency and visibility into work processes, streamlining information flow and enabling better collaboration between teams, departments and employees seated in different time zones -- all of which leads to improved productivity and bottom line.

Benefits of Cloud Computing

The potential for cost saving is the major reason of cloud services adoption by many organizations. Cloud computing gives the freedom to use services as per the requirement and pay only for what you use. Due to cloud computing it has become possible to run IT operations as a outsourced unit without much in-house resources.

Following are the benefits of cloud computing:

1. Lower IT infrastructure and computer costs for users
2. Improved performance
3. Fewer Maintenance issues
4. Instant software updates
5. Improved compatibility between Operating systems
6. Backup and recovery
7. Performance and Scalability
8. Increased storage capacity
9. Increase data safety

Benefits of cloud computing

Cloud computing is a big shift from the traditional way businesses think about IT resources. Here are seven common reasons organisations are turning to cloud computing services:

Cost

Cloud computing eliminates the capital expense of buying hardware and software and setting up and running on-site datacenters—the racks of servers, the round-the-clock electricity for power and cooling, the IT experts for managing the infrastructure. It adds up fast.

Speed

Most cloud computing services are provided self service and on demand, so even vast amounts of computing resources can be provisioned in minutes, typically with just a few mouse clicks, giving businesses a lot of flexibility and taking the pressure off capacity planning.

Global scale

The benefits of cloud computing services include the ability to scale elastically. In cloud speak, that means delivering the right amount of IT resources—for example, more or less computing power, storage, bandwidth—right when it is needed and from the right geographic location.

Productivity

On-site datacenters typically require a lot of ‒racking and stacking‖—hardware setup, software patching, and other time-consuming IT management chores. Cloud computing removes the need for many of these tasks, so IT teams can spend time on achieving more important business goals.

Performance

The biggest cloud computing services run on a worldwide network of secure datacenters, which are regularly upgraded to the latest generation of fast and efficient computing hardware. This offers several benefits over a single corporate datacenter, including reduced network latency for applications and greater economies of scale.

Reliability

Cloud computing makes data backup, disaster recovery and business continuity easier and less expensive because data can be mirrored at multiple redundant sites on the cloud provider‘s network.

Security

Many cloud providers offer a broad set of policies, technologies and controls that strengthen your security posture overall, helping protect your data, apps and infrastructure from potential threats.

Interoperability in Internet of Things

Basic Definition of Interoperability

Interoperability in cloud computing refers to the ability of customers to use the same management tools, server images and other software with a variety of **cloud computing** providers and platforms.

Interoperability in the area of Cloud computing means *"the ability to write code that works with more than one Cloud provider simultaneously, regardless of the differences between the providers"*

The ability of IT systems, as well as the business processes they support, to exchange data and enable the sharing of information and knowledge. **Interoperability** means enabling the **Cloud** ecosystem so that multiple **cloud** platforms can exchange information.

Interoperability can be defined as a measure of the degree to which diverse systems or components can work together successfully. More formally, IEEE and ISO define interoperability as the ability for two or more systems or applications to exchange information and mutually use the information that has been exchanged. In the context of cloud computing, interoperability should be viewed as the capability of public cloud services, private cloud services, and other diverse systems within the enterprise to understand each other's application and service interfaces, configuration, forms of authentication and authorization, data formats, etc. in order to work with each other. In cloud computing, the most significant interacting components are those which belong to the cloud service customer which interact with components of the cloud service provider. The nature of the interaction is a network connection using a prescribed interface or API. There are typically multiple separate interfaces, each dealing with a different aspect of the cloud service. For example, there are the functional interfaces of the cloud service itself, authentication and authorization interfaces, interfaces for administration of the cloud services, and business interfaces for billing and invoicing. The ideal of interoperability is that the interfaces are standardized in some way – i.e., they are interoperable - so that the customer can switch to another cloud service provider with minimal impact on the customer's components.

The concept of interoperability is not a new one. In the fields of information technology or systems engineering, it has been defined as the ability of two or more heterogeneous elements to not only exchange, but also use the exchanged information (interoperate). However, in the field of Cloud computing, the concept of interoperability is rather new and has recently been an active field of research. In this section we will define and explain all related concepts. The interoperability cannot be uniformly defined - there are very many different definitions which vary in technological aspects, and development frameworks, which can be more general or address only some standard details. Generally, the definition of interoperability depends on the context of its application.

Need for Interoperability in IoT

This topic is on interoperability in internet of things. Internet of things uses different types of devices. These devices are made by different vendors following different specifications; there is no one standard for IOT. So, consequently what happens is for different things the different IoT devices are made by different vendors following different specifications. Again these different devices by different vendors they follow different protocols not necessarily that they all follow the same protocol. Even the kind of users their user profiles these can also be different. So, there is so much of diversity that is inherent to these systems IoT systems and that is why it is very important to address this particular issue.

In internet of things one of the core problems or issues that has been studied quite extensively is heterogeneity of devices, protocols, user groups and many other heterogeneity aspects in from different other angles. So, this has been studied quite extensively. And one of the requirements to handle this heterogeneity issue is basically to have some kind of interoperability between these different heterogeneous aspects.

Interoperability means what that let us say that one particular device is following a particular protocol; another device follows another protocol. So, how do they talk to each other they speak the different language they speak language. So, they do not speak the same language. So, how do they talk to each other, this is one aspect.

Similarly, at different physical levels, different specifications, and different devices how do they talk to each other, they all have been made in different ways because there is no one standard that has been followed in developing these systems. So, when you want to build a singular IoT system comprising of all these different heterogeneous objects, devices, protocols, standards etcetera you need to have some kind of handshaking. And that handshaking is where these protocols have been devised which can be some kind of a middleman a middleware rather which can help these two different diverse groups to be able to talk to each other.

In the context of the cloud service, they have their own way of, –how a user or cloud application interacts with their cloud?‖ There may be different Application Programming interfaces available. This kills the cloud environment by limiting cloud choice because of vendor locking, portability, ability to use the cloud services provided by multiple vendors including the ability to use an organization‘s own existing data center resources seamlessly. So there is a need for complex developed business applications on the clouds to be interoperable. If there is not a good way of integrating data and applications across clouds then its adoption will be hampered and so the basic need for interoperability arises in organization. There are many scenarios that are desired to make use of cloud interoperability

- To move entire infrastructure to new cloud provider
- To change the cloud providers
- To move across boundaries

Interoperability can be regarded as the ongoing process of ensuring that the systems, procedures and culture of an organization are managed in such a way as to maximize opportunities for exchange and reuse of information. It includes many areas with its characteristics:

- Technical interoperability - development of standards of communication, transport and representation
- Semantic interoperability - the use of various different terms to describe similar concepts may cause problems in communication, execution of programmes and data transfers.
- Political/Human interoperability - the decision to make resources widely available has implications for organizations, their employees and end-users.
- Interoperability of communities or societies - there is an increasing need to require access to information from a wide range of sources and communities.
- International interoperability - in international matters, there are variations in standard, communication problems, language barriers, differences in communication styles, and a lack of common basis.

The main dimensions of the interoperability include: ·

Technical: This is about the physical and software compatibility of the devices including the protocols used to communicate between the devices.

Syntactical: This is the format of the message through which communication takes place and is the key to communications. For example using the XML to transmit information.

Semantic: These refers to common standards and units for example when the phrase sensors is used, it is expected that they should communicate with the devices using the same units. A temperature sensor would be able to communicate semantically with a device if both the sensor and the device use the same units of temperature.

Organizational: This is the requirement of the industries to maintain the same pattern of organization so that multiple clients can be managed.

Types of Interoperability in IoT:

1. **Device Level Interoperability**
   - **Technical Interoperability:** hardware/software components, systems and platforms that enable common machine-to-machine communication. Mainly on protocols and the infrastructure needed for the protocols to operate.
   - **Syntactic Interoperability:** Where two or more systems are able to communicate and exchange data. It allows different software components to cooperate, even if the interface and the programming language are different.
   - **Semantic Interoperability:** Where the data exchanged between two or more systems is understandable to each system. The information exchanged should be meaningful, since semantic interoperability requires useful results defined by the users of the systems involved in the exchange.
   -

2. Network Level Interoperability:

Network level interoperability deals with mechanisms to exchange messages between systems through different networks (networks of networks) to provide end-to-end communication. To make systems interoperable, each system should be able to exchange messages with other systems through various types of networks. In this level, protocol interoperability is the main focus

3. Platform level interoperability

Platform interoperability issues in IoT arises due to the availability of diverse operating systems (OSs), programming languages, data structures, architectures and access mechanisms for things and data. Developers need to obtain extensive knowledge of the platform specific APIs and information models of each different platform to be able to adapt their applications from one platform to another.

3.1 Cross-Platform Interoperability

A cross-platform IoT application can access different IoT platforms and integrate data from various platforms. For example, consider the following application scenario: a user who has health problems uses an IoT cross platform application every day to help him with his everyday tasks. The IoT application connects to the user‘s smart health platform of wearable sensors to continuously monitor his health conditions (heart rate, fall situation, and glucose level) and in an emergency, locates him and sends an ambulance. The application can also access a smart-city platform to buy a ticket to the user desired destination and shows the fastest route to the bus/train station. The cross-platform interoperability between things and data in this scenario enables interoperability across separate IoT platforms specific to one vertical domain such as smart home, smart healthcare, smart garden, etc.

3.2 Cross-Domain Interoperability

After cross-platform interoperability is enabled, cross-domain interoperability can be achieved in which different platforms within heterogeneous domains are federated to build horizontal IoT applications.

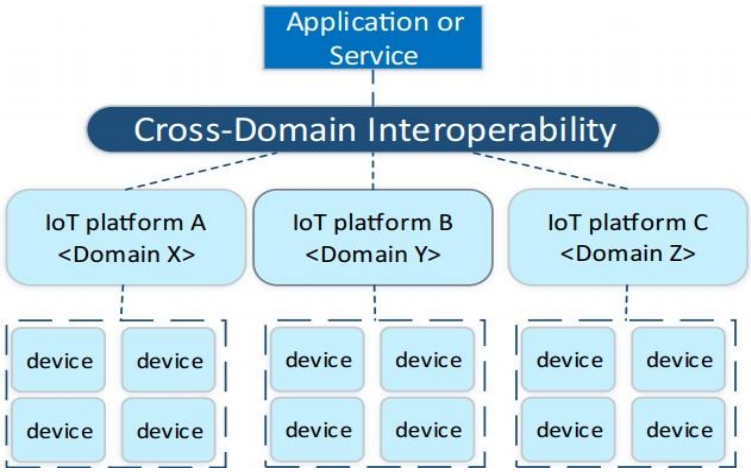


Fig. Cross-domain interoperability

Above fig. shows the concept behind cross-domain interoperability where different IoT platforms from different IoT domains (e.g. health, home, transport, etc.) can be integrated to build new innovative applications. For example, a smart home platform can provide domain-specific enablers such as air temperature and the lighting conditions. These enablers can then be exploited by other IoT platforms, such as smart healthcare, to provide more innovative applications and scenarios.

Main issues or Challenges for interoperability in IoT

The main issues in interoperability are discussed below.

a) Lack of Resources:

Resources are key to the success of an IoT project. They allow the providers to divert their resources into design and maintenance of an open API. These IoT providers mostly working in MRI or sensors. As more and more IoT enabled devices are coming to the fore, their interoperability issues have also changed in dynamics and operation. To maintain open APIs, resources are required to test the API for common mechanisms and also monitor the devices using a common monitoring layer. Other resource intense operation of the IoT system is the push and pull of the information from IoT based devices using common interfaces. Another resource consuming issue is the security of the devices. As most of the third party security software would be used, so its really hard to use those for all devices as they have interoperability issues.

b) Proprietary Technology

Another issue is that propriety software giants would not want interoperability of their devices so that they can have the market edge over consumers and so they will not support open APIs of their products. For example, Apple is a case in point.

c) Complexity

APIs are mostly incompatible with each other as far as the IoT devices are concerned and they would require a common API management system layer, which can abstract the complexities of the IoT devices but for that again the common patterns need to be simplified. The complexity of the devices is a major challenge in interoperability of the devices.

d) Security

Security of any IoT based device is becoming crucial as the privacy and protection of the users as well as the companies needs to be protected. With increasing connectivity of the IoT devices making secure communication is becoming complex as well. This revolves around the access given to various users at different levels. This depends on the role the user has for example an administrator would have different and more privileges as compared to user who is just a consumer. This aspect becomes even more complex when there is interoperability between devices. As the IoT devices have different platforms with differing standards. So to increase interoperability between IoT devices, the security of communication is also a hurdle.

e) Heterogeneous Devices

The other main issue in IoT interoperability is the heterogeneous devices that have varying data representations and heterogeneous APIs. These pose a major obstacle in the communication between the IoT devices as there is a lack of data semantics and common standards to decipher

the meaning of the data. The dynamic nature of the data calls for a complex processing like the data from the sensors, for example. So there is a lack of data continuity as the devices are incompatible at the data layer. Also the data needs to be discoverable, which becomes a challenge especially when working with large-scale data that is distributed networks and cloud based devices.

Summary of Issues and Challenges for interoperability in IoT

- Co-existence of multifarious systems (devices, sensors, equipment, etc.) that interchange location/ time dependent information in varied data formats, languages, data models/ constructs, data quality and complex interrelationships.
- Multi-version systems designed by manufacturers over time for varied application domains making formulation of global agreements and commonly accepted specifications very difficult.
- New ―Things‖ that get introduced and which support new unanticipated structures and protocols.
- Existence of low-powered devices which need to exchange data over ―lossy‖ networks and may have minimal likelihood/ accessibility for a power recharge in months/ years.
- Heterogeneous, multi-vendor and dispersed characteristics of IoT networks.

Interoperability is crucial in building systems and keeps the cost low and reduces the risk in building systems out of systems. This is how heterogeneous large-scale systems can spur innovation the manufacturers of devices have to keep in mind the cost benefit analysis while developing the design and performance of the device. Mostly the devices are resource constrained like mobile devices and sensor based devices. So investment in propriety standards seems cheaper than developing and testing new standards. The interoperability issues and promising solutions are given in Table I.

Table I
Interoperability Issues and Promising Solutions

Issue	Significance to IoT system interoperability	Promising solution
Inadequate resources	• Facilitate development of an open API. • Affects the security systems installed for the IoT devices.	• Design and development of a common monitoring layer for cheaper and efficient APIs. • Funding by government and non-governmental organizations.
Proprietary technology	Affects the universal compatibility of IoT devices	Device manufacturers to produce product platform using a standard that allow interoperability of all devices.
Complexity	Affects the management system layer making it difficult for the devices to communicate among the system.	Developing common and simplified API management system layer
Security	Affects the systems vulnerability to cyber attacks.	Build standard platforms that uses same data format to enhance the security of all that stakeholders.
Heterogeneous devices	Affects the data semantics and deciphering process making the system complex and limit the interoperability.	Establish a common standard to be used in the development of device platforms.

Cloud Computing Architecture

What Is Cloud Computing Architecture?

A cloud software system mainly requires hardware to power operations, and a way for end users to access the platform. The way this is structured in terms of components and subcomponents is an integral part of planning cloud systems out.

In an enterprise setting, determining which hardware and software components go into building a cloud environment makes up most of cloud architecture. While hardware can be chosen as off-the-shelf pieces, software is a complex part of the equation to solve for.

Many cloud service providers have made a name for themselves in the market by offering extensible hardware solutions and pairing them with easy-to-use and accessible software.

Choosing the right cloud software architecture is an important part of determining the cloud approach of any company. Inefficient cloud architecture planning can lead to over- or underuse of resources and lower cost-effectiveness.

Good planning of cloud software architecture allows for efficient and cost-effective scaling. Cloud architecture is also an integral part of fleshing out any company‘s architecture and ensuring that all cloud compute needs are taken care of.

What Are the Components of Cloud Architecture?

The components of cloud architecture are generally classified into 3 categories: a front-end platform, a backend platform, and cloud-based delivery. The architecture of the system needs the Internet for communication between the front end and the back end.

The delivery system, as the name suggests, is what allows information to be delivered between the front end and the backend. These include Infrastructure-as-a-Service (IaaS), Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) and more.

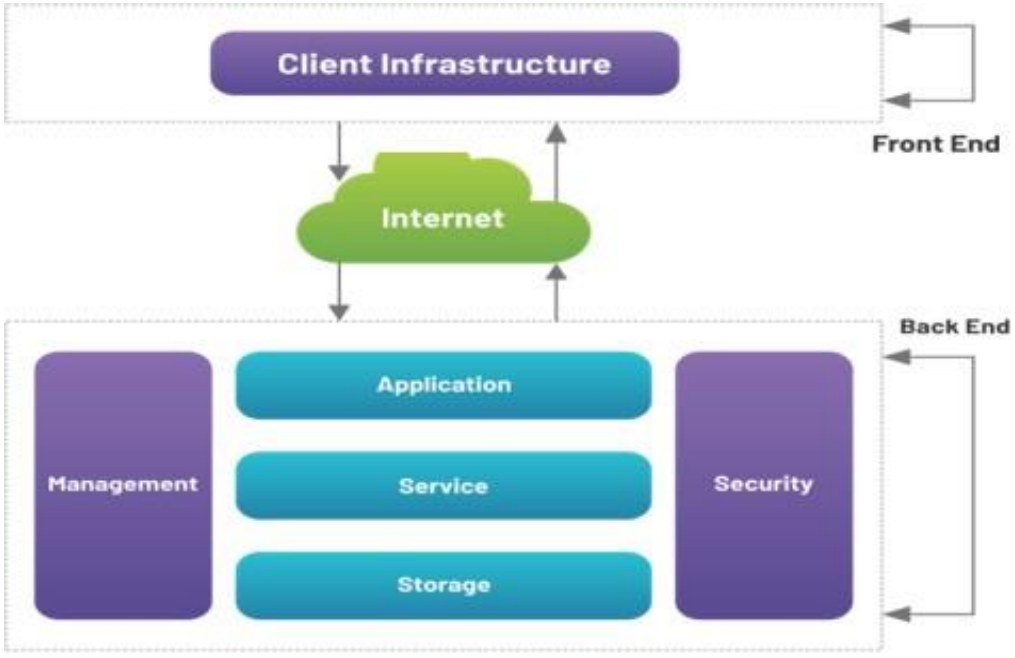


Fig.: The Structure of Cloud Computing Architecture

Cloud computing architecture also includes many sub disciplines such as cloud-based security, cloud security architecture and multi-cloud architecture.

What Is Front End Architecture?

Front end architecture is a term used to denote any user-facing part of cloud computing architecture. This is the part that the end user interacts with, and it is comprised of subcomponents that make up the user experience. Front end architecture generally takes the form of a user interface and is an integral part of how the user interacts with cloud computing software.

Most working IT professionals today will mainly interact with front-end cloud software architecture. Examples of front-end architecture include web browsers, local networks and common web apps. Gmail, a popular cloud service used by millions daily, has its front-end architecture in the form of the web application. The interface allows users to access the services offered by the Gmail architecture as a whole.

Front end architecture is made up of three parts**:**

Software

Front end software architecture includes the software that allows cloud computing software to be run from the user‘s side. In today‘s web-powered world, the front-end software architecture generally takes the form of a web browser or client-side application.

User Interface

The user interface is what the end user directly interacts with in order to perform tasks on the cloud. This includes the text editor in services such as Google Docs, or the interface to send and receive emails on Gmail. However, in some cases, the UI and software architecture are rolled into one.

Client Device/Network

The client-side device or network is an integral part of front-end architecture. This includes client-side hardware such as the user‘s PC and input devices. Generally, in cloud computing, the client device does not require a lot of computing power. This is because most of the ‒heavy‖ tasks are processed in the cloud.

What Is Back End Architecture?

Back end architecture is the part of the cloud computing architecture that powers the front-end architecture. This includes the core components of the system such as hardware and storage, and is generally located in a server farm in a geographically distant location.

Back end architecture is taken care of by the cloud service provider offering cloud software. Prominent cloud providers such as AWS generally have robust backend architecture to ensure continuous uptime with low latency. In addition to this, powerful front-end cloud architecture plays a role in creating a dependable and easy-to use product.

Back end architecture is made up of many distinct parts, and also includes services such as management and security. The system generally includes components such as:

Application

The application component of the back-end cloud software architecture generally refers to the interface being offered to the end user, except on the side of the server. This layer coordinates the various needs of the consumer with resources in the back end.

Service

This is where all the magic happens. The service is directly responsible for providing the utility in the architecture. This means that any task that is being run on the cloud computing system is taken care of mainly by the service.

Services can perform a wide variety of tasks, and function in a cloud runtime. Some services that are popular among cloud users include storage, application development environments, and web services. It is similar to the heart of the architecture, and is one of the main components in the entire system.

Cloud Runtime

The cloud runtime is where the service runs. It is similar to an operating system in the cloud, with technologies such as virtualization allowing multiple runtimes to exist on the same server.

Runtimes are created with the help of virtualization software, and are commonly referred to as Hypervisors. Software such as VMWare Fusion, Oracle Virtual Box and Oracle VM for x86 are common examples of Hypervisors. It can be compared to the foundation on which the service is built, as it manages the resources and tasks for the service.

Storage

As the name suggests, storage is where all the data required to operate the cloud software resides. Storage varies with cloud service providers today, with all of them having a product dedicated solely to cloud storage. The different kinds of storage offered includes hard drives, solid state drives, and more recently, Intel Optane DC Persistent storage.

In back end architecture, it takes the form of many hard drives in server bays. This is then managed by the management software which partitions the drives into what is needed by the operating system in the cloud to run various services.

Infrastructure

The infrastructure is the engine powering all cloud software services. This includes computing components such as the Central Processing Unit (CPU), Graphics Processing Unit (GPU), motherboard and all the other components required for the system to function smoothly, like network cards and additional specific accelerator cards for special use-cases.

The infrastructure also differs from workload to workload. While lower-powered CPUs and GPUs are available as a cheaper option, enterprise level workloads usually depend on cutting-edge hardware to run. Many cloud service providers also provide accelerators, such as Google‘s Tensor Processing Unit, available to Google Cloud Platform customers to run AI tasks.

In addition to hardware and software, there are services that are required for the architecture to function smoothly. These include:

Management

In a traditional server setting, many virtual cloud systems, known as runtimes, are on the same physical server. This means that the resources need to be managed according to the needs of the end user. This also has to occur in real-time to ensure seamless use and flexibility for the user.

Management software is in charge of allocating specific resources for certain tasks. While physical resources are abundant in a public cloud solution, ineffective management can cause bottlenecks. This makes management software essential to the ‗smooth' functioning of a cloud environment.

Management usually takes the form of what is known as ‗middleware', as it interfaces between the back end and front end. Middleware is used to divide system resources and infrastructure in a seamless and dynamic manner.

Security

Cloud security architecture is an important part of cloud software architecture. It is generally built keeping in mind visibility in order to allow for easier debugging in case of an issue with the system.

The way the system is structured should also ensure that mission-critical tasks do not get interrupted. Usually, this is done by duplicating the system virtually, so as to ensure redundancy in the tasks. Storage backups must also be done regularly, and such tasks fall under security.

Cloud security architecture also focuses on securing the server with virtual firewalls, preventing data loss and redundancy mechanisms. These are ways to keep the system running even when it is under potential attack or experiencing system failure owing to malfunctioning hardware.

Such services are integral to ensuring a complete end user experience, as they ensure the smooth functioning of the system. Tasks such as resource management are also critical to ensuring consistency uptime and redundancy for cloud tasks. Security is also an important feature to keep the data contained in the system safe from attackers.

Features/ characteristics of Cloud Computing

1. Resources Pooling

It means that the Cloud provider pulled the computing resources to provide services to multiple customers with the help of a multi-tenant model. There are different physical and virtual resources assigned and reassigned which depends on the demand of the customer. The customer generally has no control or information over the location of the provided resources but is able to specify location at a higher level of abstraction

2. On-Demand Self-Service

It is one of the important and valuable features of Cloud Computing as the user can continuously monitor the server uptime, capabilities, and allotted network storage. With this feature, the user can also monitor the computing capabilities.

3. Easy Maintenance

The servers are easily maintained and the downtime is very low and even in some cases, there is no downtime. Cloud Computing comes up with an update every time by gradually making it better. The updates are more compatible with the devices and perform faster than older ones along with the bugs which are fixed.

4. Large Network Access

The user can access the data of the cloud or upload the data to the cloud from anywhere just with the help of a device and an internet connection. These capabilities are available all over the network and accessed with the help of internet.

5. Availability

The capabilities of the Cloud can be modified as per the use and can be extended a lot. It analyzes the storage usage and allows the user to buy extra Cloud storage if needed for a very small amount.

6. Automatic System

Cloud computing automatically analyzes the data needed and supports a metering capability at some level of services. We can monitor, control, and report the usage. It will provide transparency for the host as well as the customer.

7. Economical

It is the one-time investment as the company (host) has to buy the storage and a small part of it can be provided to the many companies which save the host from monthly or yearly costs. Only the amount which is spent is on the basic maintenance and a few more expenses which are very less.

8. Security

Cloud Security, is one of the best features of cloud computing. It creates a snapshot of the data stored so that the data may not get lost even if one of the servers gets damaged. The data is stored within the storage devices, which cannot be hacked and utilized by any other person. The storage service is quick and reliable.

9. Pay as you go

In cloud computing, the user has to pay only for the service or the space they have utilized. There is no hidden or extra charge which is to be paid. The service is economical and most of the time some space is allotted for free.

10. Measured Service

Cloud computing resources used to monitor and the company uses it for recording. This resource utilization is analyzed by supporting charge-per-use capabilities. This means that the resource usages which can be either virtual server instances that are running in the cloud are getting monitored measured and reported by the service provider. The model pay as you go is variable based on actual consumption of the manufacturing organization.

11. Scalability

Resource pooling enables scalability for cloud providers and users because compute, storage and networking assets can be added or removed as needed. This helps enterprise IT teams optimize

their cloud-hosted workloads and avoid end-user bottlenecks. Clouds can scale vertically or horizontally, and providers offer automation software to handle dynamic scaling for users.

12. Resiliency

Cloud providers use a number of techniques to guard against downtime, such as minimizing regional dependencies to avoid single points of failure. Users can also extend their workloads across availability zones, which have redundant networks connecting multiple data centers in relatively close proximity. Some higher-level services automatically distribute workloads across availability zones.

13. Multi-Tenant architecture :

Set of resources provided over the cloud and been accessed by number of users across the organization with set of permissions.

14. Elastic Scalability

A key characteristic and benefit of cloud computing is elastic scalability. For example, if a web application gets an unusual amount of traffic, more servers may be created to provide that service. Thus, the application can gracefully and automatically scale with demand.

Scalability also allows cost-effectively running workloads that need a very high number of servers but only for short periods of time or occasionally. Many customers have such workloads, and especially if they can utilize the spare capacity, they can be run very cost-effectively.

Types of Cloud in Cloud Computing:

Types of Cloud

Cloud computing is an Internet-based computing in which shared the pool of resources are available over a broad network access, these resources can e provisioned or released with minimum management efforts and service provider interaction.

There are four types of cloud:

1. Public cloud
2. Private cloud
3. Hybrid cloud
4. Community cloud

1. Public cloud:

Whole computing infrastructure is located on the premises of a cloud computing company that offers the **cloud service**.

Public cloud are managed by third parties which provide cloud services over the internet to public, these services are available as pay-as-you-go billing mode.

They offer solutions for minimizing IT infrastructure costs and act as a good option for handling peak loads on the local infrastructure. They are a go to option for small enterprises, which are able to start their businesses without large upfront investments by completely relying on public infrastructure for their IT needs.

A fundamental characteristic of public clouds is multitenancy. A public cloud is meant to serve multiple users, not a single customer. A user requires a virtual computing environment that is separated, and most likely isolated, from other users.

Public clouds are the most common way of deploying cloud computing. The cloud resources (like servers and storage) are owned and operated by a third-party cloud service provider and delivered over the Internet. Microsoft Azure is an example of a public cloud. With a public cloud, all hardware, software and other supporting infrastructure is owned and managed by the cloud provider. In a public cloud, you share the same hardware, storage and network devices with other organizations or cloud ‒tenants.‖ You access services and manage your account using a web browser. Public cloud deployments are frequently used to provide web-based email, online office applications, storage and testing and development environments.

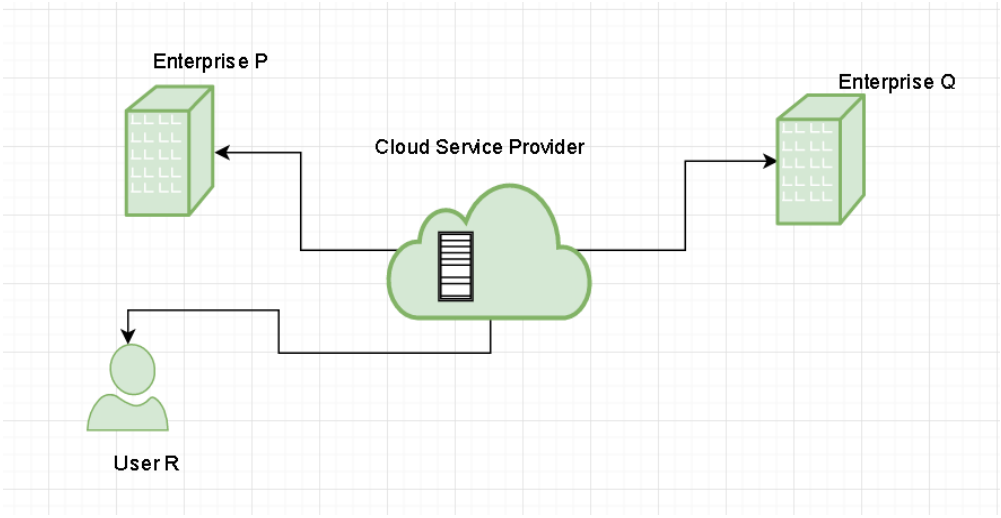


Figure: Public cloud Service

Advantages of public clouds:

- Lower costs—no need to purchase hardware or software and you pay only for the service you use.
- No maintenance—your service provider provides the maintenance.
- Near-unlimited scalability—on-demand resources are available to meet your business needs.
- High reliability—a vast network of servers ensures against failure.

2. Private cloud

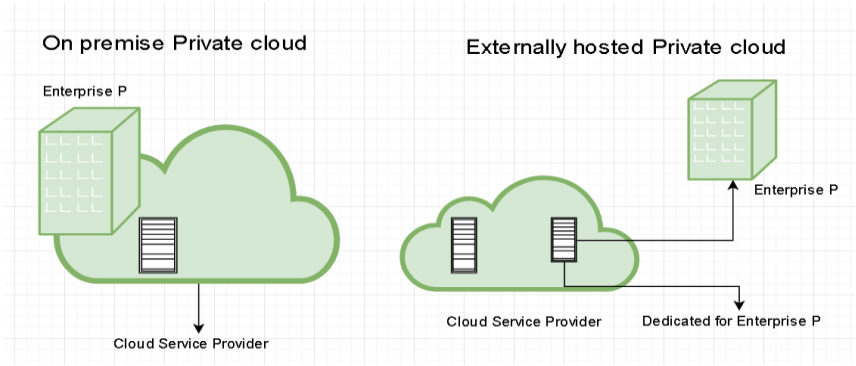


Figure: Private Cloud Service

Hosting all your computing infrastructure yourself and is not shared. The security and control level is highest while using a private network.

A private cloud consists of computing resources used exclusively by one business or organization. The private cloud can be physically located at your organization's on-site datacenter or it can be hosted by a third-party service provider. But in a private cloud, the services and infrastructure are always maintained on a private network and the hardware and software are dedicated solely to your organization. In this way, a private cloud can make it easier for an organization to customize its resources to meet specific IT requirements. Private clouds are often used by government agencies, financial institutions, any other mid- to large-size organizations with business-critical operations seeking enhanced control over their environment. Private clouds are distributed systems that work on a private infrastructure and providing the users with dynamic provisioning of computing resources. Instead of a pay-as-you-go model as in public clouds, there could be other schemes in that take into account the usage of the cloud and proportionally billing the different departments or sections of an enterprise.

Advantages of a private clouds:

- More flexibility—your organization can customize its cloud environment to meet specific business needs.
- Improved security—resources are not shared with others, so higher levels of control and security are possible.
- High scalability—private clouds still afford the scalability and efficiency of a public cloud.

3. Hybrid cloud

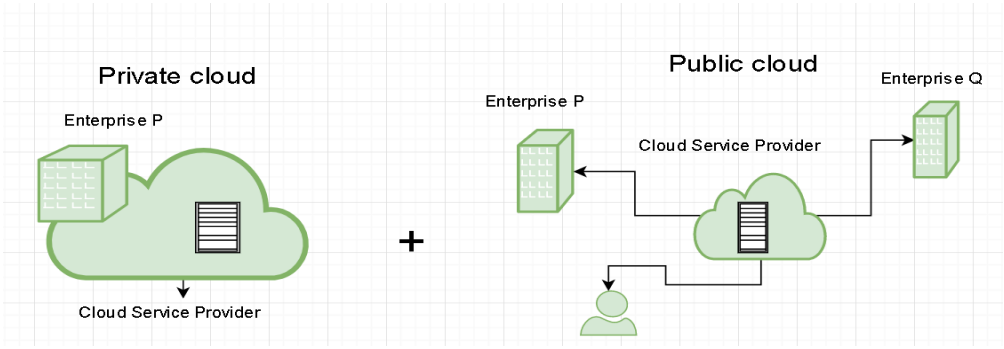


Figure: Hybrid Cloud Service

Using both private and public clouds, depending on their purpose. You host your most important applications on your own servers to keep them more secure and secondary applications elsewhere.

Hybrid cloud is a heterogeneous distributed system resulted by combining facilities of public cloud and private cloud. For this reason they are also called heterogeneous clouds.
A major drawback of private deployments is the inability to scale on demand and to efficiently address peak loads. Here public clouds are needed. Hence, a hybrid cloud takes advantages of both public and private cloud.

Often called ―the best of both worlds,‖ hybrid clouds combine on-premises infrastructure, or private clouds, with public clouds so organizations can reap the advantages of both. In a hybrid cloud, data and applications can move between private and public clouds for greater flexibility and more deployment options. For instance, you can use the public cloud for high-volume, lower-security needs such as web-based email and the private cloud (or other on-premises infrastructure) for sensitive, business-critical operations like financial reporting. In a hybrid cloud, ―cloud bursting‖ is also an option. This is when an application or resource runs in the private cloud until there is a spike in demand (such as seasonal event like online shopping or tax filing), at which point the organization can ―burst through‖ to the public cloud to tap into additional computing resources.

Advantages of hybrid clouds:

- Control—your organization can maintain a private infrastructure for sensitive assets.
- Flexibility—you can take advantage of additional resources in the public cloud when you need them.
- Cost-effectiveness—with the ability to scale to the public cloud, you pay for extra computing power only when needed.
- Ease—transitioning to the cloud does not have to be overwhelming because you can migrate gradually—phasing in workloads over time.

4. Community Cloud

A community cloud is shared between organizations with a common goal or that fit into a specific community (professional community, geographic community, etc.).

Community clouds are distributed systems created by integrating the services of different clouds to address the specific needs of an industry, a community, or a business sector.
In community cloud, the infrastructure is shared between organization which have shared concerns or tasks. The cloud may be managed by an organization or a third party.

Sectors that use community clouds are:

1. **Media industry:** Media companies are looking for quick, simple, low-cost way for increasing efficiency of content generation. Most media productions involve an extended ecosystem of partners. In particular, the creation of digital content is the outcome of a

collaborative process that includes movement of large data, massive compute-intensive rendering tasks, and complex workflow executions.

2. **Healthcare industry:** In healthcare industry community clouds are used to share information and knowledge on the global level with sensitive data in the private infrastructure.
3. **Energy and core industry:** In these sectors, the community cloud is used to cluster set of solution which collectively addresses management, deployment, and orchestration of services and operations.
4. **Scientific research:** In this organization with common interests of science share large distributed infrastructure for scientific computing.

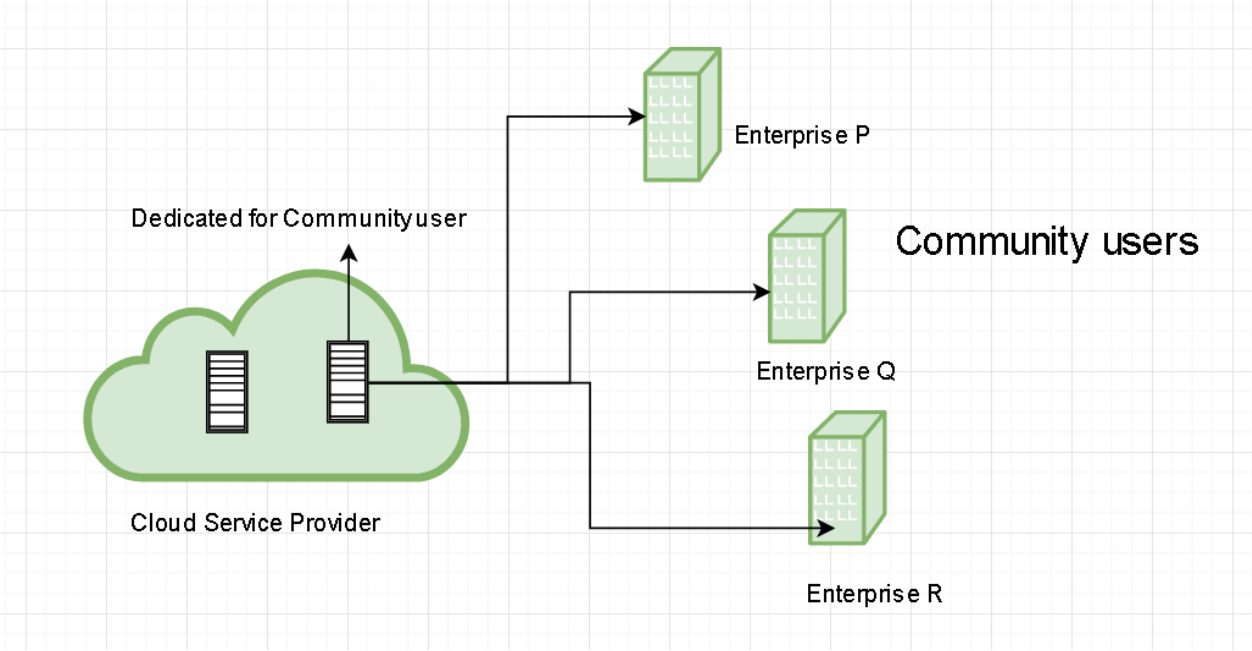


Figure: Community Cloud Service

Comparison of Public, Private and Community cloud:

Cloud Storage Type	Host	Owner	Access	Users
Public cloud	service provider	service provider	Internet	public as individuals, organizations
Private cloud	Enterprise (Third Party)	Enterprise	Intranet, VPN	Business organizations
Hybrid cloud	Enterprise (Third Party)	Enterprise	Intranet, VPN	Business organizations
Community cloud	Community (Third party)	Community	Intranet, VPN	Community members

Difference	Private	Public	Hybrid
Tenancy	Single tenancy: there's only the data of a single organization stored in the cloud.	Multi-tenancy: the data of multiple organizations in stored in a shared environment.	The data stored in the public cloud is usually multi-tenant, which means the data from multiple organizations is stored in a shared environment. The data stored in private cloud is kept private by the organization.
Exposed to the Public	No: only the organization itself can use the private cloud services.	Yes: anyone can use the public cloud services.	The services running on a private cloud can be accessed only the organization's users, while the services running on public cloud can be accessed by anyone.
Data Center Location	Inside the organization's network.	Anywhere on the Internet where the cloud service provider's services are located.	Inside the organization's network for private cloud services as well as anywhere on the Internet for public cloud services.
Cloud Service Management	The organization must have their own administrators managing their private cloud services.	The cloud service provider manages the services, where the organization merely uses them.	The organization itself must manage the private cloud, while the public cloud is managed by the CSP.
Hardware Components	Must be provided by the organization itself, which has to buy physical servers to build the private cloud on.	The CSP provides all the hardware and ensures it's working at all times.	The organization must provide hardware for the private cloud, while the hardware of CSP is used for public cloud services.
Expenses	Can be quite expensive, since the hardware, applications and network have to be provided and managed by the organization itself.	The CSP has to provide the hardware, set-up the application and provide the network accessibility according to the SLA.	The private cloud services must be provided by the organization, including the hardware, applications and network, while the CSP manages the public cloud services.

Types of cloud services:

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

1. Infrastructure as a Service (IaaS)

The most basic category of cloud computing services. With IaaS, you rent IT infrastructure—servers and virtual machines (VMs), storage, networks, operating systems—from a cloud provider on a pay-as-you-go basis

Infrastructure as a Service often provides the infrastructure such as servers, virtual machines, networks, operating system, storage, and much more on a pay-as-you-use basis. IaaS providers offer VM from small to extra-large machines.

The IaaS gives you complete freedom while choosing the instance type as per your requirements:

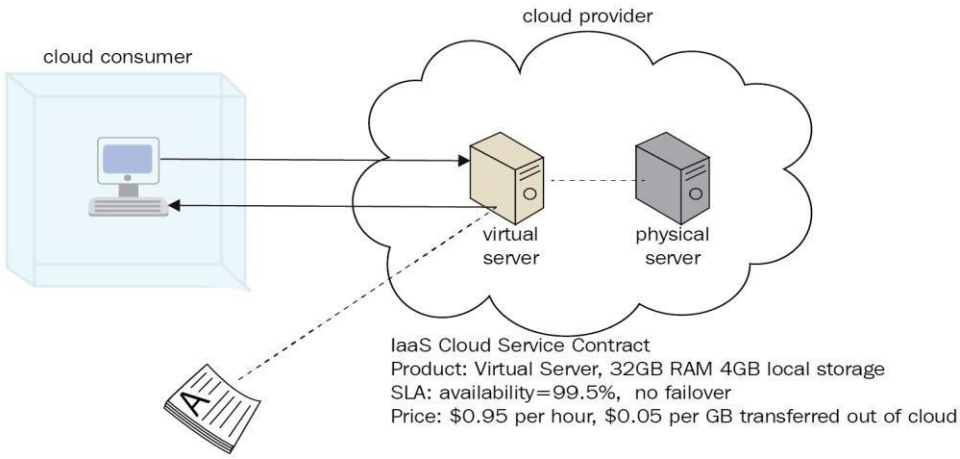


Figure: Infrastructure as a Service (IaaS)

IaaS Key Features

- Highly scalable resources
- Enterprise-grade infrastructure
- Cost depends on consumption
- Multitenant architecture, i.e. a single piece of hardware serves many users
- The client gets complete control over the infrastructure

IaaS Advantages

- The most flexible and dynamic model
- Cost-effective due to pay-as-you-go pricing
- Easy to use due to the automate d deployment of hardware
- Management tasks are virtualized, so employees have more free time for other tasks
- **Eliminates capital expense and reduces ongoing cost.** IaaS sidesteps the upfront expense of setting up and managing an on-site datacenter, making it an economical option for start-ups and businesses testing new ideas.

- **Improves business continuity and disaster recovery.** Achieving high availability, business continuity and disaster recovery is expensive, since it requires a significant amount of technology and staff. But with the right service level agreement (SLA) in place, IaaS can reduce this cost and access applications and data as usual during a disaster or outage.
- **Innovate rapidly.** As soon as you have decided to launch a new product or initiative, the necessary computing infrastructure can be ready in minutes or hours, rather than the days or weeks—and sometimes months—it could take to set up internally.
- **Respond quicker to shifting business conditions.** IaaS enables you to quickly scale up resources to accommodate spikes in demand for your application— during the holidays, for example—then scale resources back down again when activity decreases to save money.
- **Focus on your core business.** IaaS frees up your team to focus on your organisation‘s core business rather than on IT infrastructure.
- **Increase stability, reliability and supportability.** With IaaS there is no need to maintain and upgrade software and hardware or troubleshoot equipment problems. With the appropriate agreement in place, the service provider assures that your infrastructure is reliable and meets SLAs.
- **Better security.** With the appropriate service agreement, a cloud service provider can provide security for your applications and data that may be better than what you can attain in-house.
- **Gets new apps to users faster.** Because you don‘t need to first set up the infrastructure before you can develop and deliver apps, you can get them to users faster with IaaS.

IaaS Disadvantages

- Data security issues due to multitenant architecture
- Vendor outages make customers unable to access their data for a while
- The need for team training to learn how to manage new infrastructure

When to Use IaaS

IaaS can be especially advantageous in some situations:

- If you are a small company or a startup that has no budget for creating your own infrastructure
- If you are a rapidly growing company and your demands are unstable and changeable
- If you are a large company that wants to have effective control over infrastructure but pay only for the resources you actually use

Examples of IaaS

The best-known IaaS solution s vendors are Microsoft Azure, Google Compute Engine (GCE), Amazon Web Services (AWS), Cisco Metapod, DigitalOcean, Linode and Rackspace.

Common cloud vendors providing the IaaS services are:

- Google Cloud Platform
- Amazon Web Services
- IBM
- HP Public Cloud

2. Platform as a Service (PaaS)

Platform-as-a-service (PaaS) refers to the supply an on-demand environment for developing, testing, delivering and managing software applications. It is designed to quickly create web or mobile apps, without worrying about setting up or managing the underlying infrastructure of servers, storage, network and databases needed for development.

Platform as a service refers to cloud computing services that supply an on-demand environment for developing, testing, delivering and managing software applications. PaaS is designed to make it easier for developers to quickly create web or mobile apps, without worrying about setting up or managing the underlying infrastructure of servers, storage, network and databases needed for development.

The PaaS model is similar to IaaS, but it also provides the additional tools such as database management system, business intelligence services, and so on. The following figure illustrates the architecture of the PaaS model:

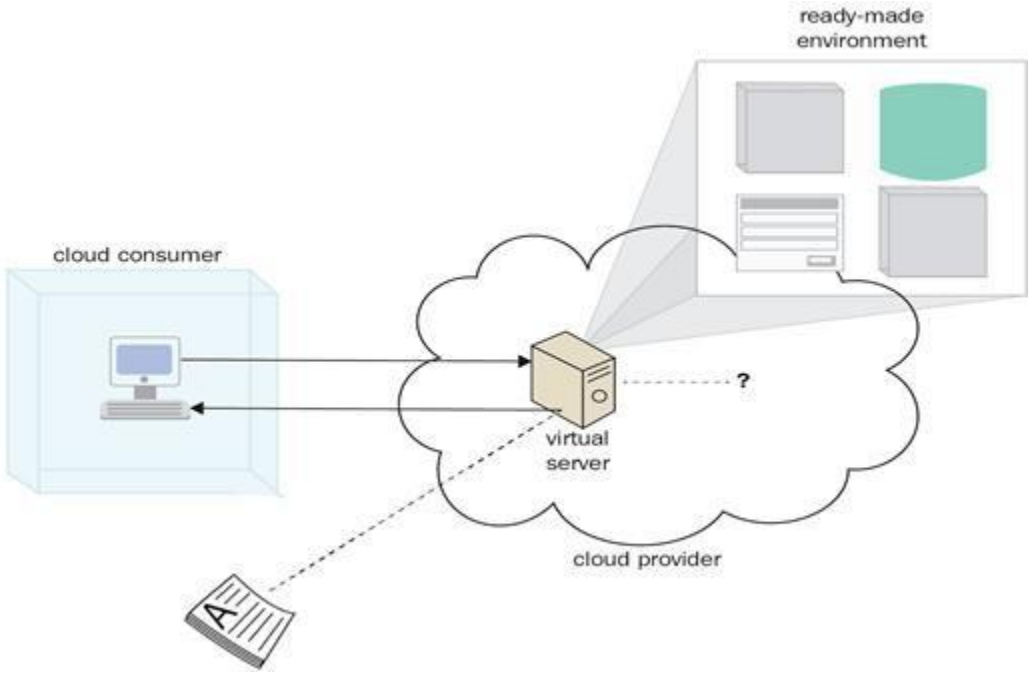


Figure: Platform as a Service (PaaS)

PaaS Key Features

- Allows for developing, testing and hosting apps in the same environment
- Resources can be scaled up and down depending on business needs
- Multiple users can access the same app in development
- The user doesn‘t have complete control over the infrastructure
- Web services and databases are integrated
- Remote teams can collaborate easily

PaaS Advantages

- PaaS-built software is highly scalable, available and multi-tenant, as it is cloud-based
- The development process is quickened and simplified
- Reduced expenses for creating, testing and launching apps
- Automated company policy
- Reduced amount of coding required
- Allows for easy migrating to the hybrid cloud

- **Automated Updates:**

  The software updates are automatically done and you get access to all the latest versions, updates etc. There is no risk of missing out on an update. This can save you plenty of resources and time. Security patches take care of any bugs and other security issues.

- **Low Investments:**

  You can use the software without having to make any investment in infrastructure that otherwise they procure in an in-house environment.

  Reduced Costs:

- Since the payment model is pay-as-you-use, there is considerable savings for you without having to install the necessary software like databases, server etc to run the applications.

- **Collaboration:**

  Since it is an internet-based platform, it enables the employees to login and work on the applications from anywhere, make their updates without physically being present in the business premises.

- **Focus:**

  Your IT team can focus better on core applications rather than being diverted on maintenance of the system.

PaaS Disadvantages

- Data security issues
- Compatibility of existing infrastructure (not every element can be cloud-enabled)
- Dependency on vendor‘s speed, reliability and support
-

When to Use PaaS

Such solutions are especially profitable to developers who want to spend more time coding, testing and deploying their applications. Utilizing PaaS is beneficial when:

- Multiple developers work on one project
- Other vendors must be included
- You want to create your own customized apps

Examples of PaaS

The best-known PaaS solutions vendors are Google App Engine, Amazon AWS, Windows Azure Cloud Services, Heroku, AWS Elastic Beanstalk, Apache Stratos and OpenShift.

Cloud platforms providing PaaS services are as follows:

- Windows Azure
- Google App Engine
- Cloud Foundry
- Amazon Web Services

3. Software as a Service (SaaS)

Software-as-a-service (SaaS) is a method for delivering software applications over the Internet as per the demand and on a subscription basis. SaaS helps you host and manage the software application and underlying infrastructure and handle any maintenance (software upgrades and security patching).

Software as a service is a method for delivering software applications over the Internet, on demand and typically on a subscription basis. With SaaS, cloud providers host and manage the software application and underlying infrastructure and handle any maintenance, like software upgrades and security patching. Users connect to the application over the Internet, usually with a web browser on their phone, tablet or PC.

Software as a Service (**SaaS**) makes the users connect to the products through the internet (or sometimes also help them build in-house as a private cloud solution) on a subscription basis model.

Below image shows the basic architecture of SaaS model.

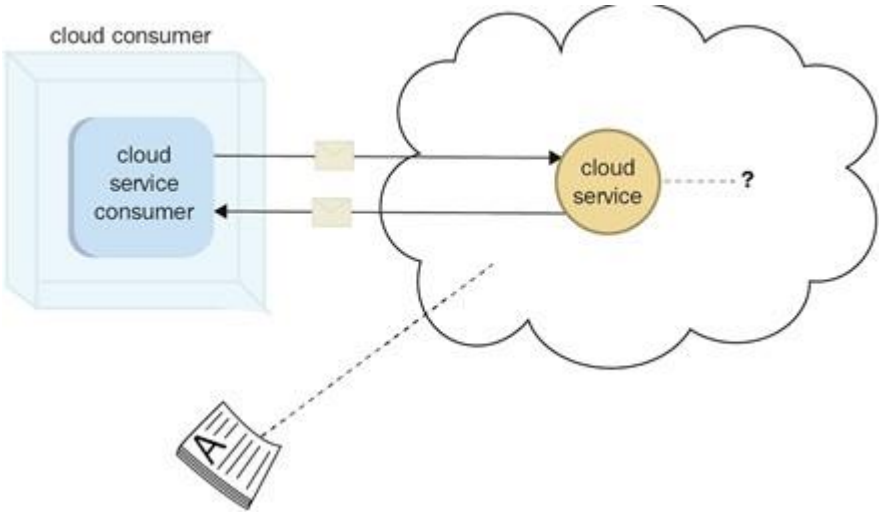


Figure: Software as a Service (SaaS)

SaaS Key Features

- The subscription model of utilizing
- No need to download, install or upgrade software
- Resources can be scaled depending on requirements
- Apps are accessible from any connected device
- The provider is responsible for everything

Advantages of SaaS

- No hardware costs
- No initial setup costs
- Automated upgrades
- Cross-device compatibility
- Accessible from any location
- Pay-as-you-go model
- Scalability
- Easy customization
- **Gain access to sophisticated applications.** To provide SaaS apps to users, you don‘t need to purchase, install, update or maintain any hardware, middleware or software. SaaS makes even sophisticated enterprise applications, such as ERP and CRM, affordable for

organizations that lack the resources to buy, deploy and manage the required infrastructure and software themselves.

- **Pay only for what you use.** You also save money because the SaaS service automatically scales up and down according to the level of usage.
- **Use free client software.** Users can run most SaaS apps directly from their web browser without needing to download and install any software, although some apps require plugins. This means that you don‘t need to purchase and install special software for your users.
- **Mobilise your workforce easily.** SaaS makes it easy to ―mobilise‖ your workforce because users can access SaaS apps and data from any Internet-connected computer or mobile device. You don‘t need to worry about developing apps to run on different types of computers and devices because the service provider has already done so. In addition, you don‘t need to bring special expertise onboard to manage the security issues inherent in mobile computing. A carefully chosen service provider will ensure the security of your data, regardless of the type of device consuming it.
- **Access app data from anywhere.** With data stored in the cloud, users can access their information from any Internet-connected computer or mobile device. And when app data is stored in the cloud, no data is lost if a user‘s computer or device fails.

SaaS Disadvantages

- Loss of control
- Limited range of solutions
- Connectivity is a must

When to Use SaaS

- If your company needs to launch a ready-made software quickly
- For short-term projects that require collaboration
- If you use applications on a temporary basis
- For applications that need both web and mobile access

Examples of SaaS

The best-known SaaS solutions vendors are Google Apps, Dropbox, Gmail, Salesforce, Cisco WebEx, Concur, GoToMeeting, Office365.

Some cloud vendors providing SaaS are:

- Google Application
- Salesforce
- Zoho
- Microsoft Office 365

Differences between SaaS, PaaS, and IaaS

Course area	IaaS	PaaS	SaaS
Cloud computing	– Virtualization – Software defined networks – Software defined data centres – Cloud storage	– Cloud based simulation – Development of cloud software (Google App Engine, Windows Azure, Amazon)	– Communication apps – Cloud storage apps – Social computing apps – Apps management – Web hosting service
Mobile technologies	– Software defined radio networks	– SMS API – Mobile application development – Mobile agents – Mobile cloud applications	– Mobile commerce apps – M-payment apps – Mobile learning apps – Mobile social apps
Internet of Things	– Software defined wireless sensor networks – Smart environments	– API for accessing the sensor data – API for context-aware applications – API for wearable computing applications	– Software for smart device management
Big data	– Environment for Hadoop projects – Environment for MongoDB projects	– Map-reduce API	– Data analysis – Visualization
IT management	– Cloud management	– Salesforce PaaS – Heroku PaaS	– Project management software – CRM software
Computer simulation and virtual reality	– Resources for simulation execution – Environment for rendering	– API for developing 3D models	– Web simulation software – 3D modelling software tools

Challenges of cloud computing

Following diagram shows the major challenges in cloud computing.

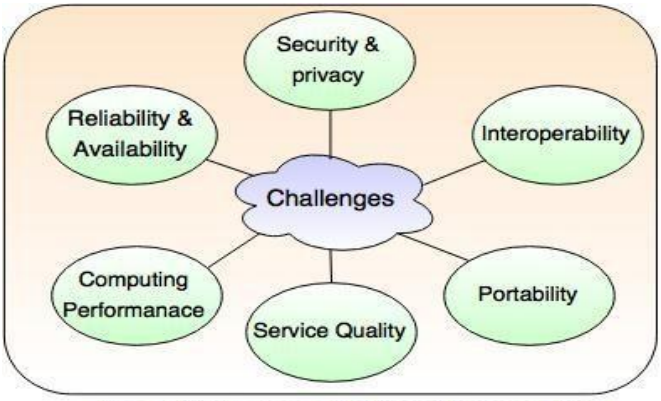


Fig. - Challenges in Cloud Computing

Security and Privacy

- Security and privacy are the main challenge in cloud computing.
- These challenges can reduced by using security applications, encrypted file systems, data loss software.

Interoperability

- The application on one platform should be able to incorporate services from the other platform. This is known as Interoperability
- It is becoming possible through web services, but to develop such web services is complex.

Portability

- The applications running on one cloud platform can be moved to new cloud platform and it should operate correctly without making any changes in design, coding.
- The portability is not possible, because each of the cloud providers uses different standard languages for their platform.

Service Quality

- The Service-Level Agreements (SLAs) of the providers are not enough to guarantee the availability and scalability. The businesses disinclined to switch to cloud without a strong service quality guarantee.

Computing Performance

- High network bandwidth is needed for data intensive applications on cloud, this results in high cost.

- In cloud computing, low bandwidth does not meet the desired computing performance.

Reliability and Availability

- Most of the businesses are dependent on services provided by third-party, hence it is mandatory for the cloud systems to be reliable and robust.

Password Security

- Industrious password supervision plays a vital role in cloud security. However, the more people you have accessing your cloud account, the less secure it is. Anybody aware of your passwords will be able to access the information you store there.
- Businesses should employ multi-factor authentication and make sure that passwords are protected and altered regularly, particularly when staff members leave. Access rights related to passwords and usernames should only be allocated to those who require them.

Data privacy

- Sensitive and personal information that is kept in the cloud should be defined as being for internal use only, not to be shared with third parties. Businesses must have a plan to securely and efficiently manage the data they gather.

Cost

- Cloud computing itself is affordable, but tuning the platform according to the company‘s needs can be expensive. Furthermore, the expense of transferring the data to public clouds can prove to be a problem for short-lived and small-scale projects.
- Companies can save some money on system maintenance, management, and acquisitions. But they also have to invest in additional bandwidth, and the absence of routine control in an infinitely scalable computing platform can increase costs.

Service Provider Reliability

- The capacity and capability of a technical service provider are as important as price. The service provider must be available when you need them. The main concern should be the service provider‘s sustainability and reputation. Make sure you comprehend the techniques via which a provider observes its services and defends dependability claims.

Downtime

- Downtime is a significant shortcoming of cloud technology. No seller can promise a platform that is free of possible downtime. Cloud technology makes small companies reliant on their connectivity, so companies with an untrustworthy internet connection probably want to think twice before adopting cloud computing.

Lack of resources/expertise

- One of the cloud challenges companies and enterprises are facing today is lack of resources and/or expertise. Organizations are increasingly placing more workloads in the cloud while cloud technologies continue to rapidly advance. Due to these factors, organizations are having a tough time keeping up with the tools. Also, the need for expertise continues to grow. These challenges can be minimized through additional training of IT and development staff.

Hacking of Brand

- Cloud computing carries some major risk factors like hacking. Some professional hackers are able to hack the application by breaking the efficient firewalls and steal the sensitive information of the organisations. A cloud provider hosts numerous clients; each can be affected by actions taken against any one of them. When any threat came into the main server it affects all the other clients also. As in distributed denial of service attacks server requests that inundate a provider from widely distributed computers.

Recovery of lost data

- Cloud services faces issue of data loss. A proper backup policy for the recovery of data must be placed to deal with the loss. Vendors must set proper infrastructures to efficiently handle with server breakdown and outages. All the cloud computing service providers must set up their servers at economically stable locations where they should have proper arrangements for the backup of all the data in at least two different locations. Ideally they should manage a hot backup and a cold backup site.

Migrating Existing Applications onto the Cloud

- You compare developing a totally new cloud application as against migrating an existing application onto the cloud – the answer is straight and simple. Migrating an existing one has its own set of hurdles, drawbacks, and challenges to face. Time has proven that cloud migration has faced troubles like security configuration, time consumption, budget overflow, unmatched requirements, downtime, etc.

Optimizing Cloud Expenses

- This has been one of the most challenging tasks for cloud users. A tad more than security issues, managing cloud spending is a tough task. With multiple reasons involved, organizations tend to waste quite a lot of their budget in unnecessary activities involved

through the cloud. Be it carelessness, lack of knowledge, hurried operations, unskilled resources – what happens is that the costs associated with cloud computing go beyond limits.

Internet connectivity

- You need good Internet connectivity and a powered-up device to access the cloud. This can be a challenge in a developing economy like Kenya, particularly outside the urban centers. Accessing cloud services through public Wi-Fi could pose a risk, unless the necessary security measures are taken.

IoT Platform

IoT platforms: An introduction

IoT platforms are regarded as the most critical component of the IoT ecosystem. Any IoT device has to connect to other IoT devices and applications (mostly cloud-based) to transfer information using standard Internet protocols. The gap between the device sensors and data networks is filled by IoT platforms. IoT platforms connect the data to the sensor arrangement, and provide insights using back-end applications to make sense of the plethora of data generated by the countless sensors.

A complete IoT system requires the following:

- Hardware such as sensors or devices. The task of sensors is to acquire data from the environment and perform actions accordingly.
- Connectivity in order to transmit the data to the cloud and a way to receive commands from the cloud. For some IoT systems, this bridge between the hardware and the cloud could be a router or a gateway in the network.
- Software that is installed on the cloud, the main task of which is to analyse the data collected from sensors and perform the task of decision making.
- A user interface, to facilitate user interaction. This can be a Web interface, where collected data is represented as numbers or graphs, or it could be a mobile app installed on a smartphone to interpret data and issue alerts to the end users, depending on sensor environments.

Types of IoT platforms

There are, essentially, four types of IoT platforms.

- ***End-to-end IoT platforms*** facilitate the handling of millions of simultaneous device connections by providing hardware, software, connectivity, security and device management tools. In addition, these platforms provide OTA (over the air) firmware updates, cloud connectivity and device management to facilitate device monitoring.
- ***Connectivity platforms*** provide cheap and low power connectivity solutions via Wi-Fi and cellular technology.
- ***Cloud platforms*** reduce the complexity of building complex network stacks and provide back-end facilities for device monitoring.
- ***Data platforms*** based on IoT platforms provide numerous tools for data routing, and facilitate management and visualisation of data using data analytics tools.

Evaluation of IoT platforms

In order to select the best IoT based platform, the following factors need to be considered.

- ***Scalability:*** The IoT platform should be scalable enough to accommodate growing needs without any hiccup. The platform should support dynamic growth, 24×7 uptime and efficient backup to prevent all sorts of errors.
- ***Reliability:*** The IoT platform should handle all sorts of failover and have disaster recovery options to keep the overall system up and running without any issues. It is very important for every end user to pay sufficient attention to the reliability of the platform, which involves a combination of parameters linked to the architecture and operations.
- ***Customisation:*** IoT platforms should support lots of customisations and be well integrated with cloud services. They should consist of APIs, and have extensive libraries as well as strong integration with other platforms to enhance the core functionality of the system with the programmer‘s customised code.
- ***Operations:*** The IoT platform‘s operations should not be hidden; rather the end user should be greeted with a centralised interface which gives all the details with regard to system statistics, hardware information and the modules or services.
- ***Protocols:*** The most important and foremost protocol to facilitate IoT is MQTT. The IoT platform should support the MQTT protocol to provide all sorts of communication from

sensors to the cloud. Almost all cloud service providers use MQTT. These include Microsoft Azure, Amazon AWS, Google Cloud, etc. The IoT platform should support the API over WebSockets, REST and CoAP.

- ***Hardware support:*** To choose an effective IoT platform, it is very important that the hardware facilitates low-end devices by using MQTT or CoAP protocols and, overall, the systems should be flexible and powerful enough to run programs written in any language.
- ***Cloud technology:*** The IoT platform should be flexible enough to support all operations without any hiccups, whether in the cloud, on premise or hybrid.
- ***Technical support:*** IoT platforms should have strong back-end technical support, and service providers should ensure regular updation of the platform with all the basic upgrades, security patches and bug fixes. Strong support develops confidence among end users.
- ***Systems architecture:*** An IoT platform should deploy production-grade, well-supported frameworks, tools and languages, making the platform easy to operate and flexible enough to support all sorts of custom implementations.
- ***Security:*** This is another significant parameter for an IoT platform. The service provider should offer SSL/TLS based communication with devices and applications, along with authentication for devices and users. It is very important to facilitate encrypted communication and the platform should have authorisation capabilities. It should be secure enough to defend the end user data from all sorts of backdoors, DDoS attacks, website vulnerabilities, malware and even rootkits.
- ***Operational expenses:*** When evaluating an IoT platform, the end user must consider parameters like pay per node, pay per active device, pay per message, the price of premium features, the price of support, etc.

Amazon Web Services IoT Platform (AWS)

AWS IoT provides secure, bi-directional communication between Internet-connected devices such as sensors, actuators, embedded micro-controllers, or smart appliances and the AWS Cloud. This enables you to collect telemetry data from multiple devices, and store and analyze the data. You can also create applications that enable your users to control these devices from their phones or tablets.

AWS IoT Core is the IoT product suite from Amazon. With AWS IoT Device Management and AWS IoT Defender, you can safely manage and monitor your devices online. AWS shines

with its Edge software that is **AWS FreeRTOS** and AWS **Greengrass**. FreeRTOS is an open-source microcontroller operating system that makes it easy to program, deploy, back up, connect, and manage small, low power edge devices. With **AWS IoT Greengrass**, connected devices can perform AWS lambda functions, make predictions based on machine learning models, keep device data in sync, and communicate securely with other devices even when disconnected from the Internet.

Features of AWS:

- **AWS IoT Device SDK** lets you quickly and easily connect your hardware device or mobile application to the AWS IoT Core.
- **Device Gateway** manages all active device connections and applies semantics to multiple protocols to ensure devices can communicate securely and efficiently with the AWS IoT core.
- **Message Broker** is a high-throughput pub/sub message broker that securely transmits low-latency messages from all your IoT devices and applications.
- **Authentication and Authorization** so there will be no exchange of data between the data and AWS IoT Core without a proven identity.
- **Rule Engine** can be used to create IoT applications that can collect, process, analyze, and process data from globally connected devices without having to manage infrastructure.
- Device management
- Secure gateway for devices
- Authentication and encryption
- Device shadow

Benefits or advantages of AWS

Following are the benefits or **advantages of AWS**:

- It offers easy to sign up process and easy to use UI (User Interface) or Management Console.
- Moreover all the AWS services are well documented for beginners.
- It offers simple billing with dynamic options as per need including per hour billing, region specific pricing, term specific pricing etc.
- Amazon is trusted vendor and ensures stability of services.

- AWS services are available at global scale which covers about 15 regions which include US, Europe and Asia pacific.
- There are multiple availability zones in each of these regions. Moreover it offers massive data centers.
- The Amazon is working towards adding new services in all the domains every day. Hence users will get access to new services as and when they are available.
- Amazon continuously focuses on machine learning, SAAS (Software as a Service) products and reduction in the costs of their services.
- Like other cloud computing platforms, AWS offers no limitations on capacity, offers speed and agility, secure and reliable environment and so on.

Drawbacks or disadvantages of AWS

Following are the drawbacks or **disadvantages of AWS**:

- There are limits on resources available on Amazon EC2 and Amazon VPC console. However one can request to increase the same.
- There are limitations on security features. For example, EC-2 classic supports max. 500 per instance and each group supports max. of 100 permissions. Moreover EC2 VPC supports max. 100 groups per VPC.
- There is technical support fees which vary as per different packages which include developers, businesses and enterprises.
- There are generic cloud computing drawbacks such as internet dependency, security concerns and so on.

AWS IoT integrates directly with the following AWS services:

- **Amazon Simple Storage Service**—Provides scalable storage in the AWS Cloud. For more information, see Amazon S3.
- **Amazon DynamoDB**—Provides managed NoSQL databases. For more information, see Amazon DynamoDB.
- **Amazon Kinesis**—Enables real-time processing of streaming data at a massive scale. For more information, see Amazon Kinesis.
- **AWS Lambda**—Runs your code on virtual servers from Amazon EC2 in response to events. For more information, see AWS Lambda.
- **Amazon Simple Notification Service**—Sends or receives notifications. For more information, see Amazon SNS.

Microsoft Azure IoT Cloud Platform

Microsoft Azure IoT solution was developed for various industry requirements. **Azure IoT** is built on decades of experience with Microsoft companies. It offers solutions for remote monitoring, predictive maintenance, smart areas, and connected products. It offers secure and scalable third party services.

Whatever your industry or the size of your business or whether you're working with a partner or alone, Azure IoT has all the tools, devices, data analytics, and security you need to meet your IoT goals.

Features Microsoft Azure:

- It provides Extensive integration with SAP, Salesforce, Oracle, WebSphere, etc.
- It offers **Condition Monitoring** that Monitor key device parameters to detect anomalies.
- **Facility Management** that Optimizes energy use, space utilization and employee productivity in your factory.
- **Asset Tracking** identifies your assets, tools, and devices used in multiple locations with the Azure IoT Remote Monitoring Solution Accelerator.
- Optimize device performance for your Azure IoT Solution Accelerator operation to increase the efficiency of factory-connected devices.
- Easy Device Registry.
- Rich Integration with SAP, Salesforce, Oracle, Web Sphere, etc
- Dashboards and visualization
- Real-time streaming

Advantages of Microsoft Azure

Here, are advantages of using Azure:

- Azure infrastructure will cost-effectively enhance your business continuity strategy
- It allows you to access the application without buying a license for the individual machine
- Windows Azure offers the best solution for your data needs, from SQL database to blobs to tables
- Offers scalability, flexibility, and cost-effectiveness
- Helps you to maintain consistency across clouds with familiar tools and resources
- Allows you to extend data center with a consistent management toolset and familiar development and identity solutions.
- You can deploy premium virtual machines in minutes which also include Linux and Windows servers
- Helps you to scale your IT resources up and down based on your needs
- You are not required to run the high-powered and high-priced computer to run cloud computing's web-based applications.
- You will not require processing power or hard disk space if you are using Azure
- Cloud computing offers virtually limitless storage

- If your personal computer or laptop crashes, all your data is still out there in the cloud, and it is still accessible
- Sharing documents leads directly to better collaboration
- If you change your device your computers, applications and documents follow you through the cloud

Disadvantages of Azure

- Cloud computing is not possible if you can't connect to the Internet
- Azure is a web-based application which requires a lot of bandwidth to download, as do large documents
- Web-based applications can sometimes be slower compared to accessing a similar software program on your desktop PC

ThingSpeak IoT Platform

ThingSpeak is regarded as the topmost open source platform for IoT. It‘s an IoT application and uses an API to store and retrieve data from things using the HTTP protocol. It facilitates the creation of sensor logging applications, location tracking applications and a social network of things with status updates.

It allows end users to aggregate, visualize and analyse live data streams in the cloud. ThingSpeak has integrated support from numerical computing software like MATLAB, allowing users to analyse, visualized and upload data using MATLAB without having to purchase the software. The platform collects and stores sensor data in the cloud and develops IoT applications. It fully supports a range of development boards like Arduino, ESP8266, BeagleBone and Pi.

The core element of ThingSpeak is a _ThingSpeak Channel‘. This stores the data that we send to ThingSpeak and comprises the following:

- Eight fields for storing data of any type — these can be used to store the data from a sensor or from an embedded device.
- Three location fields to store the latitude, longitude and the elevation. These are very useful for tracking a moving device.
- One status field, which is a short message to describe the data stored in the channel.

To use ThingSpeak, one needs to sign up and create a channel. Once that is done, send the data, and allow ThingSpeak to process and retrieve it.

Features of ThingSpeak

- Easy device configuration to transmit data to the ThingSpeak platform using standard IoT platforms.
- Real-time visualisation of sensor data and data aggregation using third party sources.
- IoT analytics runs based on schedules or events.
- MATLAB analytics, RESTful and MQTT APIs.
- Enables the prototyping and building of IoT systems without any server setup and Web software installation.
- Automatically acts on data and communication via third party services like Twilio or Twitter.
- Collect data in private channels
- App integration
- Event scheduling

Pros

- Free hosting for channels
- Easy visualization
- Provides additional features for Ruby, Node.js, and Python

Cons

- Limited data uploading for API
- ThingSpeak API can be a hurdle for beginners

Google Cloud IoT Platform

The Google Cloud IoT platform is a complete set of tools for connecting, processing, storing and analyzing data on the cloud. The platform includes scalability and fully managed cloud services.

The platform supports a variety of embedded operating systems that work with Debian Linux OS, providing immediate turnkey support for leading device manufacturers such as Intel and

Microchip. Cloud features also trigger automated changes based on real-time events through workflows.

Features Google Cloud:

- **Predictive maintenance** to make the prediction automatically when the equipment needs servicing.
- **Real-time tracking** of assets tracks valuable assets in real-time and performs complex analysis and machine learning of captured data.
- Logistics and Supply Chain Management Fleet management, inventory tracking, cargo integrity monitoring.
- Provides integration with other Google services.
- Provides huge storage
- Cuts cost for server maintenance
- Business through a fully protected, intelligent, and responsive IoT data
- Efficient and scalable
- Analyze big data

Advantages of Google Cloud –

- Designed for cloud-native businesses.
- Commitment to open source and portability.
- Deep discounts and flexible contracts.
- Fastest input/output
- Lesser access time
- Provides integration with other Google services
- There is more flexibility for a promising startup or business to get free credits for 1–2 years, hence allowing you to laverage GCP and of course is a juicy deal!
- Cheaper than AWS and Azure. Around 10–15%
- Very good in Big Data.
- Pricing model seems easier to manage.
- Easy to customize instances concerning CPU/RAM, etc.
- Better Pricing Plans Availability. Economically, Google Cloud Hosting Plans are cheaper than other Platforms Hosting Plans. ...
- Enhanced Execution. ...

- Benefits of Live Migration. ...
- Private Network. ...
- Commitment to Constant Development. ...
- Control and Security. ...
- Redundant Backups.

Disadvantages Google Cloud -

- Late entrant to IaaS market.
- Fewer features and services.
- Most of the components are Google technologies
- Limited programming language choices
- Fewer worldwide data centers.
- There are some production services that feel still as beta - but you might not need them! :)
- Google Cloud and Azure are behind AWS. Every year there is a new feature on AWS that GCP/Azure are following.
- There are some DevOps services or technologies that take more time than on AWS.

Unit IV: Implementation of IoT

Contents:

Implementation of IoT with Arduino: Introduction to Arduino, Arduino board overview, Programming environment, Arduino UNO board, Interfacing LED, LDR, LM 35, DC motor, Ultrasonic sensor and DHT 11 with Arduino, Sending data to Cloud, analysis using any IoT platform. Introduction to Raspberry Pi, Raspberry Pi board overview, Programming environment, introduction to python programming, Simple assignments/programs using Raspberry Pi, interfacing of LED, ultrasonic sensor with Raspberry Pi, Sending data to cloud, analysis of data using any IoT platform.

https://www.tutorialspoint.com/arduino/arduino_board_description.html

https://www.electronicshub.org/dht11-humidity-sensor-arduino/

Introduction to arduino:

What is Arduino?

Arduino Definition

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programed (referred to as a microcontroller) and a ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

Arduino platform offers open source hardware and software that is easy to use and is used widely for hobby projects and prototyping.

Arduino refers to an open-source electronics platform or board and the software used to program it. Arduino is designed to make electronics more accessible to artists, designers, hobbyists and ayone interested in creating interactive objects or environments.

Arduino is an open-source prototyping platform in electronics based on easy-to-use hardware and software. Subtly speaking, Arduino is a microcontroller based prototyping board which can be used in developing digital devices that can read inputs like finger on a button, touch on a screen, light on a sensor etc. and turning it in to output like switching on an LED, rotating a motor, playing songs through a speaker etc.

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++,

making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

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. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

The name Arduino comes from a bar in Ivrea, Italy, where some of the founders of the project used to meet. The bar was named after Arduin of Ivrea, who was the margrave of the March of Ivrea and King of Italy from 1002 to 1014

The key features of Arduino are −

- Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
- You can control your board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software).
- Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable.
- Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.
- Finally, Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

What do you mean by Open Source hardware and software?

When it is said that Arduino is an open source platform, it means that all the information regarding the hardware and software that Arduino provides is freely available and can be used by anyone. They provide the designs for their hardware and software which can be used by anyone. Anyone can use these designs to manufacture the hardware and can distribute the software, no licenses are required.

Why Arduino?

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a

key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step by step instructions of a kit, or sharing ideas online with other members of the Arduino community.

Because of following features Arduino is widely used of preferred

- **Inexpensive** - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than $50
- **Cross-platform** - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.

Simple, clear programming environment - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.

Open source and extensible software - The Arduino software is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.

Open source and extensible hardware - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.

Arduino provides many types of boards which are useful for various applications.

Arduino also provides its own IDE (Integrated Development Environment) which is free to download from Arduino’s website and use.

Many libraries are provided with the software and a large number of libraries developed by people from around the world are available for free. The functions implemented in these libraries can be used for quick development purposes.

This allows the user to use certain interfaces or modules without having in depth knowledge about the internal structure or working.

The point is that lack of technical or coding knowledge should not become an obstacle for development or creative thinking.

They also offer a number of expansion boards (or shields as they call them) which can be plugged into their boards and used.

All of these enable fast and easy development.

Arduino Board Types

Arduino boards based on ATMEGA328 microcontroller

Board Name	Operating Volt	Clock Speed	Digital i/o	Analog Inputs	PWM	UART	Programming Interface
Arduino Uno R3	5V	16MHz	14	6	6	1	USB via ATMega16U2
Arduino Uno R3 SMD	5V	16MHz	14	6	6	1	USB via ATMega16U2
Red Board	5V	16MHz	14	6	6	1	USB via FTDI
Arduino Pro 3.3v/8 MHz	3.3V	8MHz	14	6	6	1	FTDI-Compatible Header
Arduino Pro 5V/16MHz	5V	16MHz	14	6	6	1	FTDI-Compatible Header
Arduino mini 05	5V	16MHz	14	8	6	1	FTDI-Compatible Header
Arduino Pro mini 3.3v/8mhz	3.3V	8MHz	14	8	6	1	FTDI-Compatible Header
Arduino Pro mini 5v/16mhz	5V	16MHz	14	8	6	1	FTDI-Compatible Header
Arduino Ethernet	5V	16MHz	14	6	6	1	FTDI-Compatible Header
Arduino Fio	3.3V	8MHz	14	8	6	1	FTDI-Compatible Header
LilyPad Arduino 328	3.3V	8MHz	14	6	6	1	FTDI-

main board							Compatible Header
LilyPad Arduino simple board	3.3V	8MHz	9	4	5	0	FTDI-Compatible Header

Arduino boards based on ATMEGA32u4 microcontroller

Board Name	Operating Volt	Clock Speed	Digital i/o	Analog Inputs	PWM	UART	Programming Interface
Arduino Leonardo	5V	16MHz	20	12	7	1	Native USB
Pro micro 5V/16MHz	5V	16MHz	14	6	6	1	Native USB
Pro micro 3.3V/8MHz	5V	16MHz	14	6	6	1	Native USB
LilyPad Arduino USB	3.3V	8MHz	14	6	6	1	Native USB

Arduino boards based on ATMEGA2560 microcontroller

Board Name	Operating Volt	Clock Speed	Digital i/o	Analog Inputs	PWM	UART	Programming Interface
Arduino Mega 2560 R3	5V	16MHz	54	16	14	4	USB via ATMega16U2B
Mega Pro 3.3V	3.3V	8MHz	54	16	14	4	FTDI-Compatible Header
Mega Pro 5V	5V	16MHz	54	16	14	4	FTDI-Compatible Header
Mega Pro Mini 3.3V	3.3V	8MHz	54	16	14	4	FTDI-Compatible Header

Arduino boards based on AT91SAM3X8E microcontroller

Board Name	Operating Volt	Clock Speed	Digital i/o	Analog Inputs	PWM	UART	Programming Interface
Arduino Mega 2560 R3	3.3V	84MHz	54	12	12	4	USB native

Features of Arduino Boards

Arduino Board	Processor	Memory	Digital I/O	Analogue I/O
Arduino Uno	16Mhz ATmega328	2KB SRAM, 32KB flash	14	6 input, 0 output
Arduino Due	84MHz AT91SAM3X8E	96KB SRAM, 512KB flash	54	12 input, 2 output
Arduino Mega	16MHz ATmega2560	8KB SRAM, 256KB flash	54	16 input, 0 output
Arduino Leonardo	16MHz ATmega32u4	2.5KB SRAM, 32KB flash	20	12 input, 0 output

Arduino Pro Mini **Processor**: ATmega328 **Frequency**: 8 (3.3 V)/16, (5 V) MHz **Format**: Mini **Size**: 17.8 mm × 33.0 mm **Host interface**: 6-pin serial header **Voltage**: 3.3 V / 5 V **Flash**: 32 KB **EEPROM**: 1 KB **SRAM**: 2 KB **Digital I/O Pins**: 14 (6 are PWM based) **Analogue Input**: 6 **Analogue Output**: N/A **Miscellaneous**: This Arduino model was designed and manufactured by SparkFun Electronics.	**Arduino Uno** **Processor**: ATmega328P **Frequency**: 16 MHz **Format**: Arduino **Size**: 68.6 mm × 53.3 mm **Host interface**: USB/8U2(Rev1&2)/16U2(Rev3) **Voltage**: 5 V **Flash**: 32 KB **EEPROM**: 1 KB **SRAM**: 2 KB **Digital I/O Pins**: 2 (14 are PWM based) **Analogue Input**: 6 **Analogue Output**: 6 **Miscellaneous**: This works with the very same ATmega328 as late-model Duemilanove, but while the Duemilanove incorporated an FTDI IC for USB, the Uno operates with an ATmega16U2 (ATmega8U2 before rev3) programmed as a serial converter.

Arduino Micro **Processor**: Atmega32U4 **Frequency**: 16 MHz **Format**: Mini **Size**: 17.8 mm × 48.3 mm **Host interface**: N/A **Voltage**: 5 V **Flash**: 32 KB **EEPROM**: 1 KB **SRAM**: 2.5 KB **Digital I/O Pins**: 20 (7 are PWM based) **Analogue Input**: 12 **Analogue Output**: N/A **Miscellaneous**: This Arduino model was designed in collaboration with by Adafruit.	**Arduino Nano** **Processor**: ATmega328 (ATmega168 before v3.0) **Frequency**: 16 MHz **Format**: minimal **Size**: 43.18 mm × 18.54 mm **Host interface**: USB/ FTDIFT232R **Voltage**: 5 V **Flash**: 16/32 KB **EEPROM**: 0.5/1 KB **SRAM**: 1/2 KB **Digital I/O Pins**: 14 (6 are PWM based) **Analogue Input**: 8 **Analogue Output**: N/A Miscellaneous**: This is a tiny version of the Arduino that works with USB power and is built with a surface-mounted processor.**
Arduino 101 / Genuino 101 **Processor**: Intel® Curie™ module 2 tiny cores, one x86 (Quark SE) along with ARC **Frequency**: 32 MHz **Format**: Arduino / Genuino **Size**: 68.6 mm × 53.4 mm **Host interface**: USB **Voltage**: 3.3 V **Flash**: 196 KB **EEPROM**: N/A **SRAM**: 24 KB **Digital I/O Pins**: 14 (4 are PWM based) **Analogue Input**: 6 **Analogue Output**: N/A **Miscellaneous:** Includes 6-axis accelerometer, gyroscope and Bluetooth	**Arduino Zero** **Processor**: ATSAMD21G18A **Frequency**: 48 MHz **Format**: Arduino **Size**: 68.6 mm × 53.3 mm **Host interface**: USB **Voltage**: 3.3 V **Flash**: 256 KB **EEPROM**: 0-16 Kb emulation **SRAM**: 32 KB **Digital I/O Pins**: 14 (12 are PWM based) **Analogue Input**: 6 **Analogue Output**: 1 **Miscellaneous**: 32-bit architecture
Arduino Leonardo **Processor**: Atmega32U4 **Frequency**: 16 MHz **Format**: Arduino **Size**: 68.6 mm × 53.3 mm **Host interface**: USB/32U4 **Voltage**: 5 V **Flash**: 32 KB **EEPROM**: 1 KB **SRAM**: 2.5 KB **Digital I/O Pins**: 20 (7 are PWM based)	**Arduino Mega2560** **Processor**: ATmega2560 **Frequency**: 16 MHz **Format**: Mega **Size**: 101.6 mm × 53.3 mm **Host interface**: USB/8U2(Rev1&2)/16U2(Rev3) **Voltage**: 5 V **Flash**: 256 KB **EEPROM**: 4 KB **SRAM**: 8 KB **Digital I/O Pins**: 54 (15 are PWM based)

Analogue Input: 12 **Analogue Output**: N/A **Miscellaneous**: The Leonardo is designed with the Atmega32U4 processor, which features an integrated USB controller, which reduces a single chip as compared to other Arduino versions.	**Analogue Input**: 16 **Analogue Output**: N/A **Miscellaneous**: The majority of the shielding that had been created for the Duemilanove, Diecimila, or Uno will have no problems here, however some shields may not accommodate due to of incompatibility with the supplemental pins.

Components of an Arduino Board

ATmega Microcontroller
USB or barrel Jack
Reset button
Power indicator LED
TX RX LED's
Main IC
Voltage regulator
Pins like analog, digital, PWM (pulse width modulation), AREF, 5V, 3.3V

Advantages of Arduino Board

The Arduino boards are very easy to get started.

The Arduino boards are used in the automatic room light control.

On the Arduino, both software, hardware, and IDE are open source.

The Arduino boards are used in the real-time application.

Arduino UNO board

Features of Arduino Uno Board

The **features of Arduino Uno ATmega328** includes the following.
The operating voltage is 5V
The recommended input voltage will range from 7v to 12V
The input voltage ranges from 6v to 20V
Digital input/output pins are 14
Analog i/p pins are 6
DC Current for each input/output pin is 40 mA
DC Current for 3.3V Pin is 50 mA
Flash Memory is 32 KB
SRAM is 2 KB
EEPROM is 1 KB
CLK Speed is 16 MHz

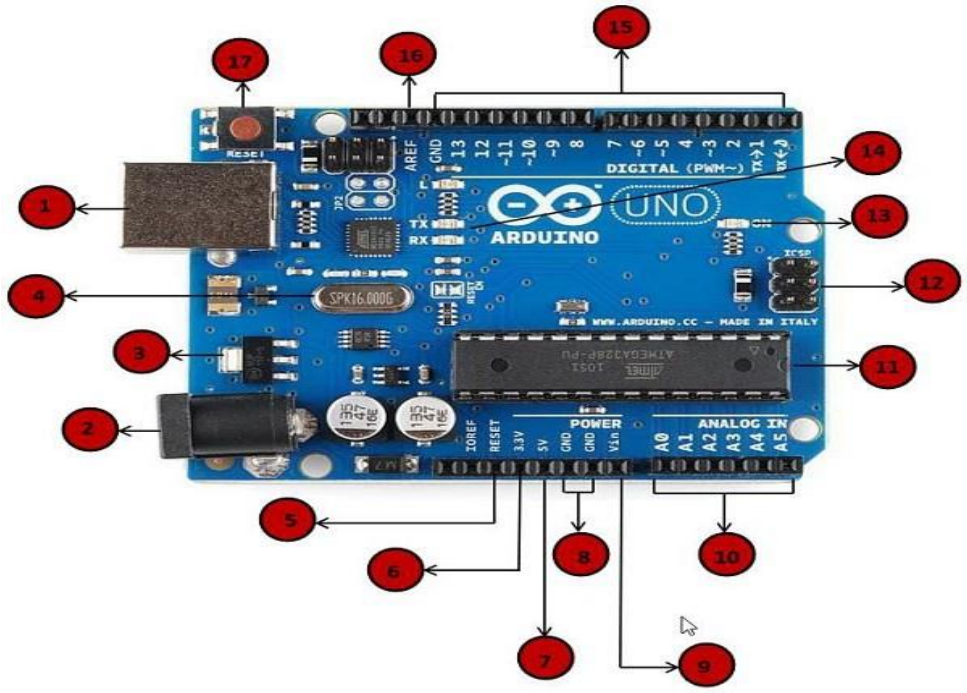


Fig. : Arduino UNO board Pin diagram

1	**Power USB** Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection (1).
2	**Power (Barrel Jack)** Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack (2).
3	**Voltage Regulator** The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.
4	**Crystal Oscillator** The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz.
5,17	**Arduino Reset** You can reset your Arduino board, i.e., start your program from the beginning. You can reset the UNO board in two ways. First, by using the reset button (17) on the board. Second, you can connect an external reset button to the Arduino pin labelled RESET (5).

6,7 8,9	**Pins (3.3, 5, GND, Vin)** 3.3V (6) − Supply 3.3 output volt 5V (7) − Supply 5 output volt Most of the components used with Arduino board works fine with 3.3 volt and 5 volt. GND (8)(Ground) − There are several GND pins on the Arduino, any of which can be used to ground your circuit. Vin (9) − This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.
10	**Analog pins** The Arduino UNO board has six analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.
11	**Main microcontroller** Each Arduino board has its own microcontroller (11). You can assume it as the brain of your board. The main IC (integrated circuit) on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company. You must know what IC your board has before loading up a new program from the Arduino IDE. This information is available on the top of the IC. For more details about the IC construction and functions, you can refer to the data sheet.
12	**ICSP pin** Mostly, ICSP (12) is an AVR, a tiny programming header for the Arduino consisting of MOSI, MISO, SCK, RESET, VCC, and GND. It is often referred to as an SPI (Serial Peripheral Interface), which could be considered as an "expansion" of the output. Actually, you are slaving the output device to the master of the SPI bus.
13	**Power LED indicator** This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.
14	**TX and RX LEDs** On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led (13). The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.
15	**Digital I/O** The Arduino UNO board has 14 digital I/O pins (15) (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled “~” can be used to generate PWM.

16	**AREF** AREF stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Programming environment:

The Arduino software is open-source. The source code for the Java environment is released under the GPL and the C/C++ microcontroller libraries are under the LGPL.

Sketch − The first new terminology is the Arduino program called "**sketch**".

Structure

Arduino programs can be divided in three main parts: **Structure, Values** (variables and constants), and **Functions**. In this tutorial, we will learn about the Arduino software program, step by step, and how we can write the program without any syntax or compilation error.

Let us start with the **Structure**. Software structure consist of two main functions −

Setup( ) function

Loop( ) function

```
Void setup ( )
{

}
```

PURPOSE − The **setup ()** function is called when a sketch starts. Use it to initialize the variables, pin modes, start using libraries, etc. The setup function will only run once, after each power up or reset of the Arduino board.

INPUT − -
OUTPUT − -
RETURN − -

```
Void Loop ( )
 {

 }
```

PURPOSE − After creating a **setup ()** function, which initializes and sets the initial values, the **loop ()** function does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the Arduino board.

INPUT −

OUTPUT −

RETURN –

Data types used in Arduino programs

Data types in C refers to an extensive system used for declaring variables or functions of different types. The type of a variable determines how much space it occupies in the storage and how the bit pattern stored is interpreted.

The following table provides all the data types that you will use during Arduino programming.

Void	**Boolean**	**char**	**Unsigned char**	**byte**	**int**	**Unsigned int**	**word**
Long	Unsigned long	short	Float	double	array	String-char array	String-object

void

The void keyword is used only in function declarations. It indicates that the function is expected to return no information to the function from which it was called.

Example

```
Void Loop ( )
        {
// rest of the code
          }
```

Boolean

A Boolean holds one of two values, true or false. Each Boolean variable occupies one byte of memory.

Example

```
boolean val = false; // declaration of variable with type boolean and initialize it with false
boolean state = true; // declaration of variable with type boolean and initialize it with true
```

Char

A data type that takes up one byte of memory that stores a character value. Character literals are written in single quotes like this: 'A' and for multiple characters, strings use double quotes: "ABC".

However, characters are stored as numbers. You can see the specific encoding in the ASCII chart. This means that it is possible to do arithmetic operations on characters, in which the ASCII value of the character is used. For example, 'A' + 1 has the value 66, since the ASCII value of the capital letter A is 65.

Unsigned char

Unsigned char is an unsigned data type that occupies one byte of memory. The unsigned char data type encodes numbers from 0 to 255.

Example

Unsigned Char chr_y = 121 ; // declaration of variable with type Unsigned char and initialize it with character y

byte

A byte stores an 8-bit unsigned number, from 0 to 255.

Example

byte m = 25 ;//declaration of variable with type byte and initialize it with 25

int

Integers are the primary data-type for number storage. int stores a 16-bit (2-byte) value. This yields a range of -32,768 to 32,767 (minimum value of -2^15 and a maximum value of (2^15) - 1).

The **int** size varies from board to board. On the Arduino Due, for example, an **int** stores a 32-bit (4-byte) value. This yields a range of -2,147,483,648 to 2,147,483,647 (minimum value of -2^31 and a maximum value of (2^31) - 1).

Example

int counter = 32 ;// declaration of variable with type int and initialize it with 32

Unsigned int

Unsigned ints (unsigned integers) are the same as int in the way that they store a 2 byte value. Instead of storing negative numbers, however, they only store positive values, yielding a useful range of 0 to 65,535 (2^16) - 1). The Due stores a 4 byte (32-bit) value, ranging from 0 to 4,294,967,295 (2^32 - 1).

Example

Unsigned int counter = 60 ; // declaration of variable with type unsigned int and initialize it with 60

Word

On the Uno and other ATMEGA based boards, a word stores a 16-bit unsigned number. On the Due and Zero, it stores a 32-bit unsigned number.

Example

word w = 1000 ;//declaration of variable with type word and initialize it with 1000

Long

Long variables are extended size variables for number storage, and store 32 bits (4 bytes), from -2,147,483,648 to 2,147,483,647.

Example

```
Long velocity = 102346;      //declaration of variable with type Long and initialize it with 102346
```

unsigned long

Unsigned long variables are extended size variables for number storage and store 32 bits (4 bytes). Unlike standard longs, unsigned longs will not store negative numbers, making their range from 0 to 4,294,967,295 (2^32 - 1).

Example

```
Unsigned Long velocity = 101006 ;// declaration of variable with type Unsigned Long and initialize it with 101006
```

short

A short is a 16-bit data-type. On all Arduinos (ATMega and ARM based), a short stores a 16-bit (2-byte) value. This yields a range of -32,768 to 32,767 (minimum value of -2^15 and a maximum value of (2^15) - 1).

Example

```
short val = 13 ;//declaration of variable with type short and initialize it with 13
```

float

Data type for floating-point number is a number that has a decimal point. Floating-point numbers are often used to approximate the analog and continuous values because they have greater resolution than integers.

Floating-point numbers can be as large as 3.4028235E+38 and as low as -3.4028235E+38. They are stored as 32 bits (4 bytes) of information.

Example

```
float num = 1.352;//declaration of variable with type float and initialize it with 1.352
```

double

On the Uno and other ATMEGA based boards, Double precision floating-point number occupies four bytes. That is, the double implementation is exactly the same as the float, with no gain in precision. On the Arduino Due, doubles have 8-byte (64 bit) precision.

Example

```
double num = 45.352 ;// declaration of variable with type double and initialize it with 45.352
```

Before we start explaining the variable types, a very important subject we need to make sure, you fully understand is called the **variable scope**.

What is Variable Scope?

Variables in C programming language, which Arduino uses, have a property called scope. A scope is a region of the program and there are three places where variables can be declared. They are − Inside a function or a block, which is called **local variables**.

In the definition of function parameters, which is called **formal parameters**.

Outside of all functions, which is called **global variables**.

Local Variables

Variables that are declared inside a function or block are local variables. They can be used only by the statements that are inside that function or block of code. Local variables are not known to function outside their own. Following is the example using local variables −

```
Void setup ()
{
}
Void loop () {
  int x , y ;
  int z ; Local variable declaration
  x = 0;
  y = 0; actual initialization
  z = 10;
}
```

Global Variables

Global variables are defined outside of all the functions, usually at the top of the program. The global variables will hold their value throughout the life-time of your program.

A global variable can be accessed by any function. That is, a global variable is available for use throughout your entire program after its declaration.

The following example uses global and local variables –

```
Int T , S ;
float c = 0 ; Global variable declaration
Void setup ()
 {
 }
Void loop ()
{
  int x , y ;
  int z ; Local variable declaration
  x = 0;
  y = 0; actual initialization
  z = 10;
```

```
}
```

Operator used in Arduino programs

An operator is a symbol that tells the compiler to perform specific mathematical or logical functions. C language is rich in built-in operators and provides the following types of operators − Arithmetic Operators

Comparison Operators

Boolean Operators

Bitwise Operators

Compound Operators

Arithmetic Operators

Assume variable A holds 10 and variable B holds 20 then –

Operator name	Operator simple	Description	Example
assignment operator	=	Stores the value to the right of the equal sign in the variable to the left of the equal sign.	A = B
addition	+	Adds two operands	A + B will give 30
subtraction	-	Subtracts second operand from the first	A - B will give -10
multiplication	*	Multiply both operands	A * B will give 200
division	/	Divide numerator by denominator	B / A will give 2
modulo	%	Modulus Operator and remainder of after an integer division	B % A will give 0

Comparison Operators

Assume variable A holds 10 and variable B holds 20 then −

Operator name	Operator simple	Description	Example
equal to	==	Checks if the value of two operands is equal or not, if yes then condition becomes true.	(A == B) is not true
not equal to	!=	Checks if the value of two operands is equal or not, if values are not equal then condition becomes true.	(A != B) is true
less than	<	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(A < B) is true
greater than	>	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	(A > B) is not true
less than or equal to	<=	Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	(A <= B) is true
greater than or equal to	>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(A >= B) is not true

Boolean Operators

Assume variable A holds 10 and variable B holds 20 then −

Operator name	Operator simple	Description	Example
and	&&	Called Logical AND operator. If both the operands are non-zero then then condition becomes true.	(A && B) is true
or	||	Called Logical OR Operator. If any of the two operands is non-zero then then condition becomes true.	(A || B) is true
not	!	Called Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true then	!(A && B) is false

		Logical NOT operator will make false.	

Bitwise Operators

Assume variable A holds 60 and variable B holds 13 then −

Operator name	Operator simple	Description	Example
and	**&**	Binary AND Operator copies a bit to the result if it exists in both operands.	(A & B) will give 12 which is 0000 1100
or	|	Binary OR Operator copies a bit if it exists in either operand	(A | B) will give 61 which is 0011 1101
xor	^	Binary XOR Operator copies the bit if it is set in one operand but not both.	(A ^ B) will give 49 which is 0011 0001
not	~	Binary Ones Complement Operator is unary and has the effect of 'flipping' bits.	(~A) will give -60 which is 1100 0011
shift left	<<	Binary Left Shift Operator. The left operands value is moved left by the number of bits specified by the right operand.	A << 2 will give 240 which is 1111 0000
shift right	>>	Binary Right Shift Operator. The left operands value is moved right by the number of bits specified by the right operand.	A >> 2 will give 15 which is 0000 1111

Compound Operators

Assume variable A holds 10 and variable B holds 20 then −

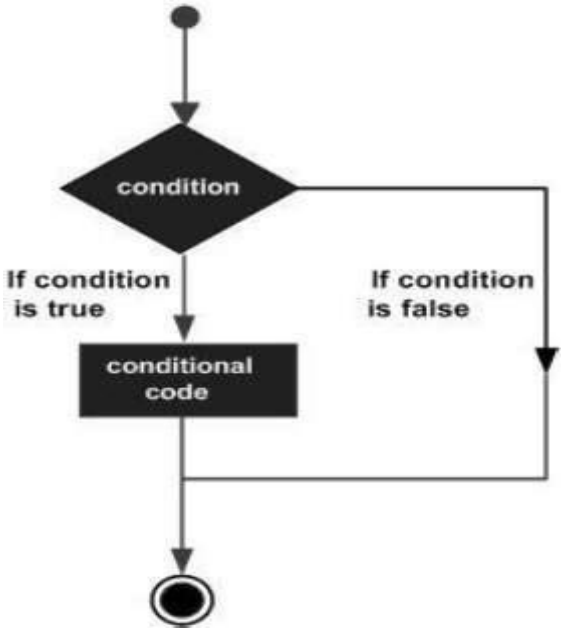
Operator name	Operator simple	Description	Example
increment	++	Increment operator, increases integer value by one	A++ will give 11
decrement	--	Decrement operator, decreases integer value by one	A-- will give 9

compound addition	**+=**	Add AND assignment operator. It adds right operand to the left operand and assign the result to left operand	B += A is equivalent to B = B+ A
compound subtraction	**-=**	Subtract AND assignment operator. It subtracts right operand from the left operand and assign the result to left operand	B -= A is equivalent to B = B - A
compound multiplication	***=**	Multiply AND assignment operator. It multiplies right operand with the left operand and assign the result to left operand	B*= A is equivalent to B = B* A
compound division	**/=**	Divide AND assignment operator. It divides left operand with the right operand and assign the result to left operand	B /= A is equivalent to B = B / A
compound modulo	**%=**	Modulus AND assignment operator. It takes modulus using two operands and assign the result to left operand	B %= A is equivalent to B = B % A
compound bitwise or	**|=**	bitwise inclusive OR and assignment operator	A |= 2 is same as A = A | 2
compound bitwise and	**&=**	Bitwise AND assignment operator	A &= 2 is same as A = A & 2

Decision making structures

Decision making structures require that the programmer specify one or more conditions to be evaluated or tested by the program. It should be along with a statement or statements to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false.

Following is the general form of a typical decision making structure found in most of the programming languages −

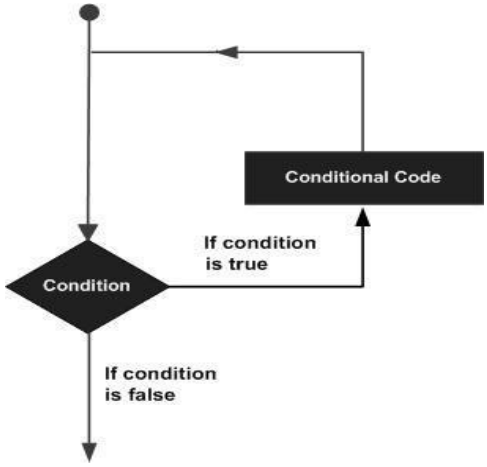


Control Statements are elements in Source Code that control the flow of program execution. They are −

S.N.	Control Statement & Description
1	**If statement** It takes an expression in parenthesis and a statement or block of statements. If the expression is true then the statement or block of statements gets executed otherwise these statements are skipped.
2	**If …else statement** An **if** statement can be followed by an optional else statement, which executes when the expression is false.
3	**If…else if …else statement** The **if** statement can be followed by an optional **else if...else**statement, which is very useful to test various conditions using single if...else if statement.
4	**switch case statement** Similar to the if statements, **switch...case** controls the flow of programs by allowing the programmers to specify different codes that should be executed in various conditions.
5	**Conditional Operator? :** The conditional operator? : is the only ternary operator in C.

Programming languages provide various control structures that allow for more complicated execution paths.

A loop statement allows us to execute a statement or group of statements multiple times and following is the general form of a loop statement in most of the programming languages −



C programming language provides the following types of loops to handle looping requirements.

S.N.	Loop & Description
1	**while loop** while loops will loop continuously, and infinitely, until the expression inside the parenthesis, () becomes false. Something must change the tested variable, or the while loop will never exit.
2	**do…while loop** The **do…while** loop is similar to the while loop. In the while loop, the loop-continuation condition is tested at the beginning of the loop before performed the body of the loop.
3	**for loop** A **for loop** executes statements a predetermined number of times. The control expression for the loop is initialized, tested and manipulated entirely within the for loop parentheses.
4	**Nested Loop** C language allows you to use one loop inside another loop. The following example illustrates the concept.
5	**Infinite loop** It is the loop having no terminating condition, so the loop becomes infinite.

Functions allow structuring the programs in segments of code to perform individual tasks. The typical case for creating a function is when one needs to perform the same action multiple times in a program.

Arduino provides four different time manipulation functions. They are −

S. No.	Function & Description
1	**delay () function** The way the **delay()** function works is pretty simple. It accepts a single integer (or number) argument. This number represents the time (measured in milliseconds).
2	**delayMicroseconds () function** The **delayMicroseconds()** function accepts a single integer (or number) argument. There are a thousand microseconds in a millisecond, and a million microseconds in a second.
3	**millis () function** This function is used to return the number of milliseconds at the time, the Arduino board begins running the current program.
4	**micros () function** The micros() function returns the number of microseconds from the time, the Arduino board begins running the current program. This number overflows i.e. goes back to zero after approximately 70 minutes.

Declaring Arrays

Arrays occupy space in memory. To specify the type of the elements and the number of elements required by an array, use a declaration of the form −

type array Name [array Size] ;

The compiler reserves the appropriate amount of memory. (Recall that a declaration, which reserves memory is more properly known as a definition). The arraySize must be an integer constant greater than zero. For example, to tell the compiler to reserve 11 elements for integer array C, use the declaration –

int C[12]; // C is an array of 12 integers

Arrays can be declared to contain values of any non-reference data type. For example, an array of type string can be used to store character strings.

The pins on the Arduino board can be configured as either inputs or outputs. We will explain the functioning of the pins in those modes. It is important to note that a majority of Arduino analog pins, may be configured, and used, in exactly the same manner as digital pins.

Pins Configured as INPUT

Arduino pins are by default configured as inputs, so they do not need to be explicitly declared as inputs with **pinMode()** when you are using them as inputs. Pins configured this way are said to be in a high-impedance state. Input pins make extremely small demands on the circuit that they are sampling, equivalent to a series resistor of 100 megaohm in front of the pin.

This means that it takes very little current to switch the input pin from one state to another. This makes the pins useful for such tasks as implementing a capacitive touch sensor or reading an LED as a photodiode.

Pins configured as pinMode(pin, INPUT) with nothing connected to them, or with wires connected to them that are not connected to other circuits, report seemingly random changes in pin state, picking up electrical noise from the environment, or capacitively coupling the state of a nearby pin.

Pull-up Resistors

Pull-up resistors are often useful to steer an input pin to a known state if no input is present. This can be done by adding a pull-up resistor (to +5V), or a pull-down resistor (resistor to ground) on the input. A 10K resistor is a good value for a pull-up or pull-down resistor.

Using Built-in Pull-up Resistor with Pins Configured as Input

There are 20,000 pull-up resistors built into the Atmega chip that can be accessed from software. These built-in pull-up resistors are accessed by setting the **pinMode()** as INPUT_PULLUP. This effectively inverts the behavior of the INPUT mode, where HIGH means the sensor is OFF and LOW means the sensor is ON. The value of this pull-up depends on the microcontroller used. On most AVR-based boards, the value is guaranteed to be between 20kΩ and 50kΩ. On the Arduino Due, it is between 50kΩ and 150kΩ. For the exact value, consult the datasheet of the microcontroller on your board.

When onnecting a sensor to a pin configured with INPUT_PULLUP, the other end should be connected to the ground. In case of a simple switch, this causes the pin to read HIGH when the switch is open and LOW when the switch is pressed. The pull-up resistors provide enough current to light an LED dimly connected to a pin configured as an input. If LEDs in a project seem to be working, but very dimly, this is likely what is going on.

Same registers (internal chip memory locations) that control whether a pin is HIGH or LOW control the pull-up resistors. Consequently, a pin that is configured to have pull-up resistors turned on when the pin is in INPUTmode, will have the pin configured as HIGH if the pin is then switched to an OUTPUT mode with pinMode(). This works in the other direction as well, and an output pin that is left in a HIGH state will have the pull-up resistor set if switched to an input with pinMode().

Example

```
pinMode(3,INPUT) ; // set pin to input without using built in pull up resistor
pinMode(5,INPUT_PULLUP) ; // set pin to input using built in pull up resistor
```

Different in built functions in Arduino IDE

pinMode() Function

The pinMode() function is used to configure a specific pin to behave either as an input or an output. It is possible to enable the internal pull-up resistors with the mode INPUT_PULLUP. Additionally, the INPUT mode explicitly disables the internal pull-ups.

pinMode() Function Syntax

```
Void setup () {
  pinMode (pin , mode);
}
```

pin − the number of the pin whose mode you wish to set

mode − INPUT, OUTPUT, or INPUT_PULLUP.

digitalWrite() Function

The **digitalWrite()** function is used to write a HIGH or a LOW value to a digital pin. If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW. If you do not set the pinMode() to OUTPUT, and connect an LED to a pin, when calling digitalWrite(HIGH), the LED may appear dim. Without explicitly setting pinMode(), digitalWrite() will have enabled the internal pull-up resistor, which acts like a large current-limiting resistor.

digitalWrite() Function Syntax

```
Void loop()
{
  digitalWrite (pin ,value);
}
```

pin − the number of the pin whose mode you wish to set

value − HIGH, or LOW.

digitalRead()

Description

Reads the value from a specified digital pin, either HIGH or LOW.

Syntax

digitalRead(pin)

Parameters

pin: the number of the digital pin you want to read (*int*)

Returns

HIGH or LOW

analogRead( ) function

Reads the value from the specified analog pin. The Arduino board contains a 6 channel, 10-bit analog to digital converter. This means that it will map input voltages between 0 and 5 volts into integer values between 0 and 1023. This yields a resolution between readings of: 5 volts / 1024 units or, .0049 volts (4.9 mV) per unit. The input range and resolution can be changed using analogReference().

It takes about 100 microseconds (0.0001 s) to read an analog input, so the maximum reading rate is about 10,000 times a second.

In the lower-right part of the Arduino board, you will see six pins marked "Analog In". These special pins not only tell whether there is a voltage applied to them, but also its value. By using the **analogRead()** function, we can read the voltage applied to one of the pins.

This function returns a number between 0 and 1023, which represents voltages between 0 and 5 volts. For example, if there is a voltage of 2.5 V applied to pin number 0, analogRead(0) returns 512.

analogRead() function Syntax

```
analogRead(pin);
```

pin − the number of the analog input pin to read from (0 to 5 on most boards, 0 to 7 on the Mini and Nano, 0 to 15 on the Mega)

Returns-int *(0 to 1023)*

analogWrite()

Writes an analog value (PWM wave) to a pin. Can be used to light a LED at varying brightnesses or drive a motor at various speeds. After a call to analogWrite(), the pin will generate a steady square wave of the specified duty cycle until the next call to analogWrite() (or a call to digitalRead() or digitalWrite() on the same pin). The frequency of the PWM signal on most pins is approximately 490 Hz. On the Uno and similar boards, pins 5 and 6 have a frequency of approximately 980 Hz. Pins 3 and 11 on the Leonardo also run at 980 Hz.

On most Arduino boards (those with the ATmega168 or ATmega328), this function works on pins 3, 5, 6, 9, 10, and 11. On the Arduino Mega, it works on pins 2 - 13 and 44 - 46. Older Arduino boards with an ATmega8 only support analogWrite() on pins 9, 10, and 11.

Syntax

analogWrite(pin, value)

Parameters
pin: the pin to write to.
value: the duty cycle: between 0 (always off) and 255 (always on).
Returns-Nothing

analogReference()

Configures the reference voltage used for analog input (i.e. the value used as the top of the input range). The options are:

DEFAULT: the default analog reference of 5 volts (on 5V Arduino boards) or 3.3 volts (on 3.3V Arduino boards)
INTERNAL: an built-in reference, equal to 1.1 volts on the ATmega168 or ATmega328 and 2.56 volts on the ATmega8 (not available on the Arduino Mega)
INTERNAL1V1: a built-in 1.1V reference (Arduino Mega only)
INTERNAL2V56: a built-in 2.56V reference (Arduino Mega only)
EXTERNAL: the voltage applied to the AREF pin (0 to 5V only) is used as the reference.

Syntax
analogReference(type)

Parameters
type: which type of reference to use (DEFAULT, INTERNAL, INTERNAL1V1, INTERNAL2V56, or EXTERNAL).

Serial.begin

Sets the data rate in bits per second (baud) for serial data transmission. For communicating with the computer, use one of these rates: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, or 115200. You can, however, specify other rates - for example, to communicate over pins 0 and 1 with a component that requires a particular baud rate.
An optional second argument configures the data, parity, and stop bits. The default is 8 data bits, no parity, one stop bit.

Syntax
Serial.begin(speed)
Serial.begin(speed, config)

Serial.print()

Syntax

Serial.print(val)
Serial.print(val, format)

Parameters

val: the value to print - any data type
format: specifies the number base (for integral data types) or number of decimal places (for floating point types)
Returns

size_t (long): print() returns the number of bytes written, though reading that number is optional
For example:

Serial.print(78) gives "78"
Serial.print(1.23456) gives "1.23"
Serial.print('N') gives "N"
Serial.print("Hello world.") gives "Hello world."

Serial.print(78, BIN) gives "1001110"

Serial.print(78, OCT) gives "116"

Serial.print(78, DEC) gives "78"

Serial.print(78, HEX) gives "4E"

Serial.println(1.23456, 0) gives "1"

Serial.println(1.23456, 2) gives "1.23"

Serial.println(1.23456, 4) gives "1.2346"

Serial.println

Prints data to the serial port as human-readable ASCII text followed by a carriage return character (ASCII 13, or '\r') and a newline character (ASCII 10, or '\n'). This command takes the same forms as Serial.print().

Syntax
Serial.println(val)
Serial.println(val, format)

Parameters

val: the value to print - any data type
format: specifies the number base (for integral data types) or number of decimal places (for floating point types)

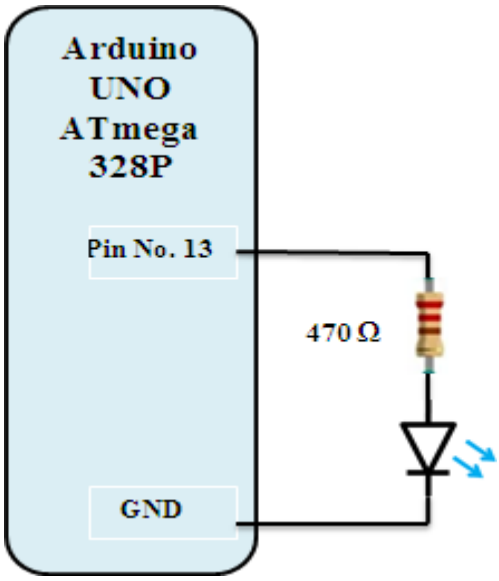
Returns

size_t (long): println() returns the number of bytes written, though reading that number is optional

Simple Assignments using Arduino:

Interfacing LED with Arduino

Blinking of LED Anode of LED connected to pin number 13 & cathode to GND



Algorithm:

1. Start
2. Define pin number 13 as LED pin
3. Set pin mode 13 as output pin
4. Send high signal on pin 13(LED)
5. delay
6. Send low signal on pin 13(LED)
7. Delay
8. Go to step 4.

Program:

```
int led = 13;
void setup()
 {
pinMode(led, OUTPUT);
}
void loop()
{
Serial.begin(9600);
digitalWrite(led, HIGH);
delay(1000);
```

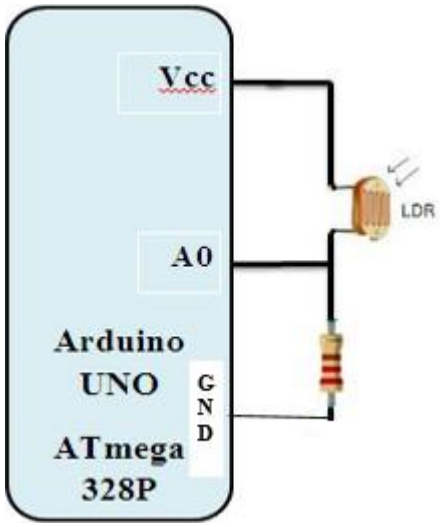
```
digitalWrite(led, LOW);
delay(1000);
}
```

Code to change the brightness of the LED using PWM.

```
//Initializing LED Pin
int led_pin = 6;
void setup()
 {
  //Declaring LED pin as output
  pinMode(led_pin,  OUTPUT);
 }
  void loop()
 {
    for(int i=0; i<255; i++)
     {
    analogWrite(led_pin, i);
    delay(5);
     }
  for(int i=255; i>0; i--)
     {
    analogWrite(led_pin, i);
    delay(5);
      }
 }
```

Interfacing LDR Sensor to Arduino:

Interfacing diagram



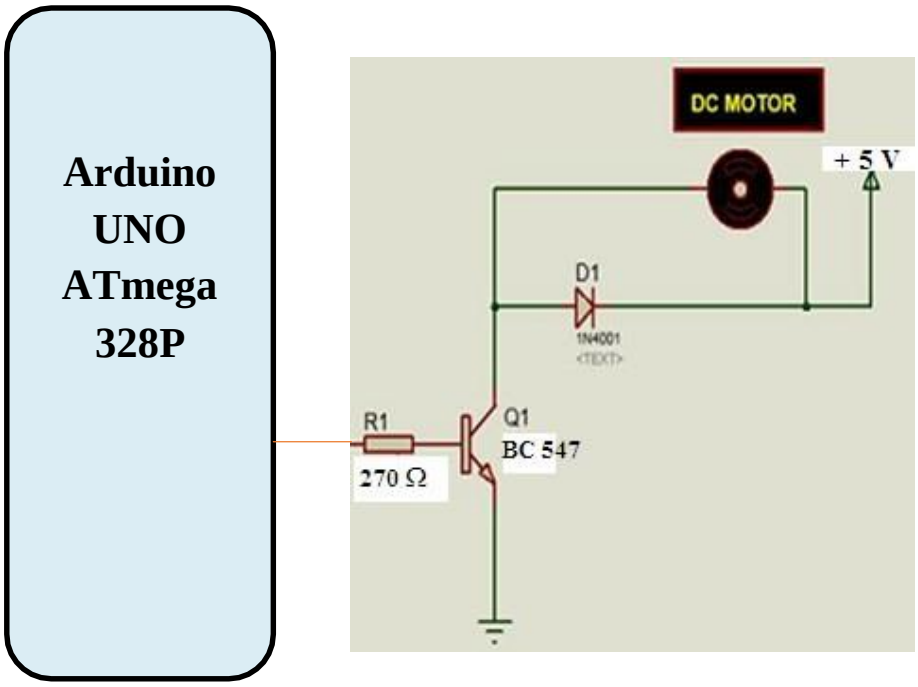
Algorithm

Start
Initialize Ao pin for taking the data from sensor
Initial sensor value =0
Initialize baud rate = 9600
Read the sensor value
Send the sensor value to serial port
Go to step 5

Program

```
int sensorPin = A0;
int sensorValue = 0;
void setup()
{
Serial.begin(9600); //sets serial port for communication
}
void loop()
{
sensorValue = analogRead(sensorPin);
Serial.println(sensorValue);
delay(100);
}
```

Interfacing DC MOTOR with Arduino:



The pin no. 9 is connected through transistor driver. The transistor is used to drive motor

Spin Control Arduino Code

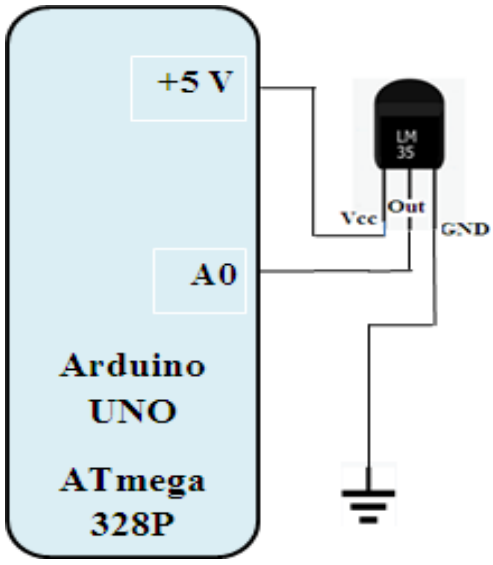
```
int motorPin = 9;
void setup() {
}
void loop() {
  digitalWrite(motorPin, HIGH);
}
```

Code to change the speed of DC motor using PWM.

```
int motor_pin = 9;
void setup()
```

```
{
  //Declaring motor pin as output
  pinMode(motor_pin, OUTPUT);
}
void loop()
 {
  for(int i=0; i<255; i++)
{
    analogWrite(motor_pin, i);
    delay(5);
  }
  for(int i=255; i>0; i--)
{
    analogWrite(motor_pin, i);
    delay(5);
  }
```

Interfacing of Temperature Sensor LM35 with Arduino



The output of LM 35 sensor is analog and it is connected to analog pin A0 One pin is connected to + 5V and one pin connected to ground.

```
float temp;
```

```
int tempPin = A0;
void setup()
 {
Serial.begin(9600);
}
void loop() {
temp = analogRead(tempPin);
temp = temp *0.48828125;
Serial.print("Temprature =");
Serial.print  (temp);
Serial.print(" Degree Celcisus");
Serial.println();
delay(1000);
}
```

Temperature & Humidity Sensor DHT11 using Arduino

All the DHT11 Sensors are accurately calibrated in the laboratory and the results are stored in the memory. A single wire communication can be established between any microcontroller like Arduino and the DHT11 Sensor.

Also, the length of the cable can be as long as 20 meters. The data from the sensor consists of integral and decimal parts for both Relative Humidity (RH) and temperature.

The data from the DHT11 sensor consists of 40 – bits and the format is as follows:

8 – Bit data for integral RH value, 8 – Bit data for decimal RH value, 8 – Bit data for integral Temperature value, 8 – Bit data for integral Temperature value and 8 – Bit data for checksum.

Example

Consider the data received from the DHT11 Sensor is

00100101 00000000 00011001 00000000 00111110.

This data can be separated based on the above mentioned structure as follows

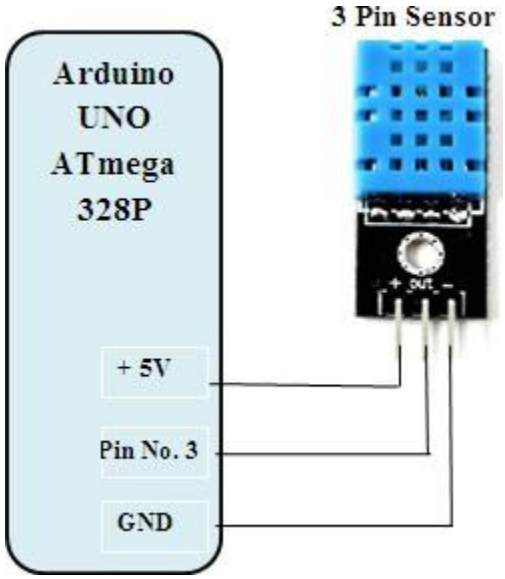
In order to check whether the received data is correct or not, we need to perform a small calculation. Add all the integral and decimals values of RH and Temperature and check whether the sum is equal to the checksum value i.e. the last 8 – bit data.

00100101 + 00000000 + 00011001 + 00000000 = 00111110

This value is same as checksum and hence the received data is valid. Now to get the RH and Temperature values, just convert the binary data to decimal data.

RH = Decimal of 00100101 = 37%

Temperature = Decimal of 00011001 = 25^0C



Program: for DHT11 Temperature and Humidity

```
#include <dht.h>        // Include library
#define dhtoutPin 3     // Defines pin number to which the sensor is connected
dht DHT;                        // Creates a DHT object
void setup()
{
        Serial.begin(9600);
}
void loop() {
        int readData = DHT.read11(dhtoutPin);
        float t = DHT.temperature;     // Read temperature
        float h = DHT.humidity;                 // Read humidity
        Serial.print("Temperature = ");
        Serial.print(t);
        Serial.print("°C ");
        Serial.print("Humidity = ");
        Serial.print(h);
```

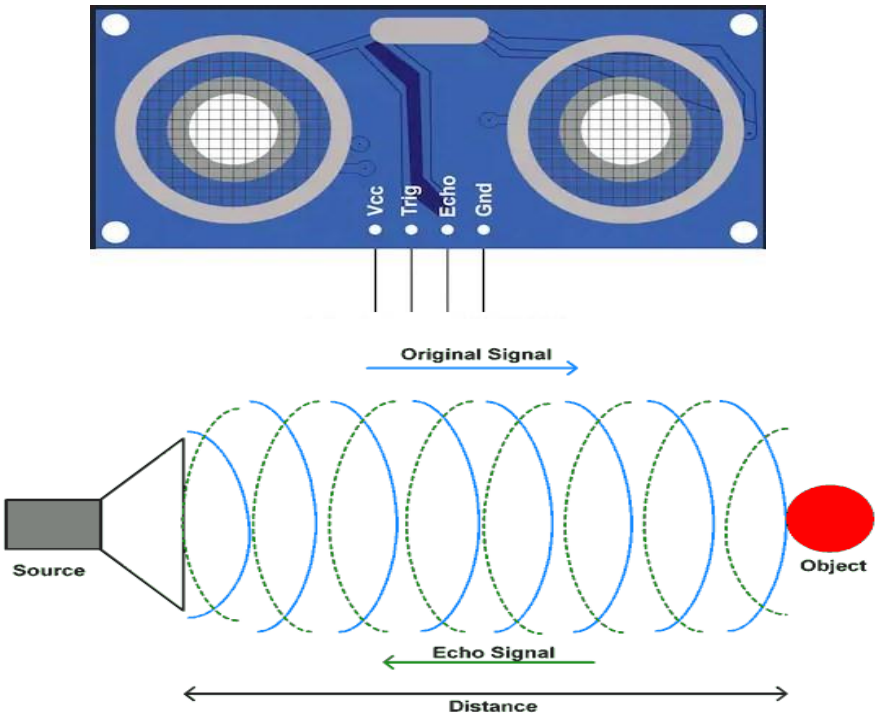
```
        Serial.println("% ");
        Serial.println("");
        delay(2000); // wait two seconds
}
```

Interfacing of Ultrasonic Sensor Arduino

The HC-SR04 ultrasonic sensor uses SONAR to determine the distance of an object just like the bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package from 2 cm to 400 cm or 1" to 13 feet.

The ultrasonic sensor works on the principle of SONAR and RADAR system which is used to determine the distance to an object.

An ultrasonic sensor generates the high-frequency sound (ultrasound) waves. When this ultrasound hits the object, it reflects as echo which is sensed by the receiver as shown in below figure.



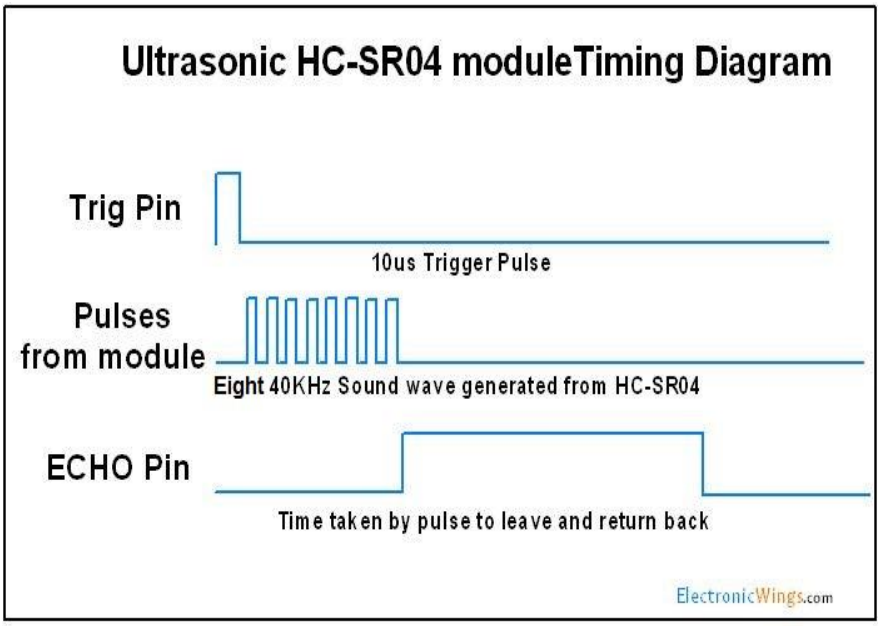
VCC - +5 V supply

TRIG – Trigger input of sensor. Microcontroller applies 10 us trigger pulse to the HC-SR04 ultrasonic module.

ECHO–Echo output of sensor. Microcontroller reads/monitors this pin to detect the obstacle or to find the distance.

GND – Ground

By measuring the time required for the echo to reach to the receiver, we can calculate the distance. This is the basic working principle of Ultrasonic module to measure distance.



HC-SR04 Ultrasonic Module Timing Diagram

We need to transmit trigger pulse of at least 10 us to the HC-SR04 Trig Pin.

Then the HC-SR04 automatically sends Eight 40 kHz sound wave and wait for rising edge output at Echo pin.

When the rising edge capture occurs at Echo pin, start the Timer and wait for falling edge on Echo pin.

As soon as the falling edge is captured at the Echo pin, read the count of the Timer. This time count is the time required by the sensor to detect an object and return back from an object.

Now how to calculate distance?

We know that,

$$\text{Distance} = \text{Speed x Time}$$

The speed of sound waves is 343 m/s. or 34300 cm/sec

So,

$$\textbf{Distance}(in\ cm) = \frac{\textbf{34300}*\textbf{Time}}{\textbf{2}}$$

As Time is in micro seconds

So,

Distance(in cm)=((34300*Time)/2)/1000000

Distance(in cm)=(0.0343*Time)/2

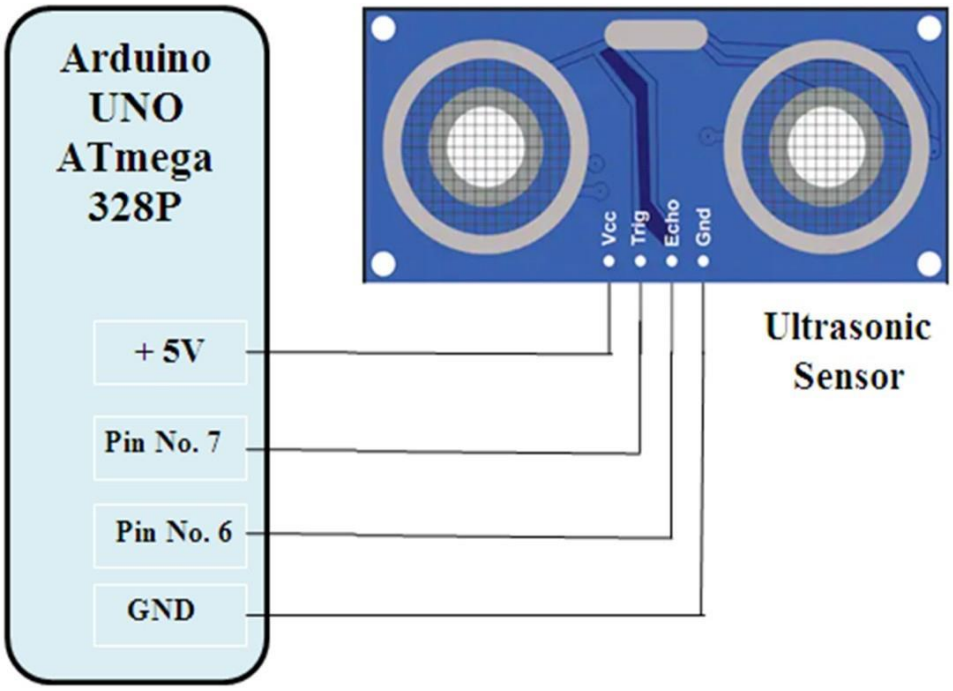
Technical Specifications HC-SR04

Power Supply − +5 DC
Quiescent Current − <2mA
Working Current − 15mA
Effectual Angle − <15°
Ranging Distance − 2cm – 400 cm/1″ – 13ft
Resolution − 0.3 cm
Measuring Angle − 30 degree

Components Required

You will need the following components −

1 × Breadboard
1 × Arduino Uno R3
1 × ULTRASONIC Sensor (HC-SR04)



In diagram pin 7 of Arduino is connected to TRIGGER pin of Ultrasonic sensor and Pin 6 of Arduino is connected to ECHO of Ultrasonic sensor

Arduino Code

```
const int trigPin = 7;
const int echoPin = 6;
float duration, distance;
void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
```

```
  Serial.begin(9600);
}
void loop()
{
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration*.0343)/2;
  Serial.print("Distance = ");
  Serial.print(distance);
  Serial.println(" cm");
  delay(100);
}
```

Program to detect obstacle using Ultrasonic sensor interfaced with Arduino
Assume buzzer is connected to pin 2 in above ckt.

```
  int const trigPin = 7;
int const echoPin = 6;
int const buzzPin = 2;
void setup()
{
pinMode(trigPin, OUTPUT); // trig pin will have pulses output
pinMode(echoPin, INPUT); // echo pin should be input to get pulse width
pinMode(buzzPin, OUTPUT); // buzz pin is output to control buzzering
}
void loop()
{
int duration, distance;
// Output pulse with 1ms width on trigPin
digitalWrite(trigPin, HIGH);
delay(1);
digitalWrite(trigPin, LOW);
// Measure the pulse input in echo pin
duration = pulseIn(echoPin, HIGH);
// Distance is half the duration devided by 29.1 (from datasheet)
distance= duration*0.034/2;
// if distance less than 0.5 meter and more than 0 (0 or less means over range)
if (distance <= 50 && distance >= 0)
```

```
{
// Buzz
digitalWrite(buzzPin, HIGH);
} else {
// Don't buzz
digitalWrite(buzzPin, LOW);
}
// Waiting 60 ms won't hurt any one
delay(60);
}
```

➢ Sending Data to Cloud using Arduino:

How to Send Data from Arduino to Webpage using WiFi:

Wireless communication between Electronic devices and modules is very important, to make them 'Fit' in the World of Internet of Things. HTTP protocol and HTML language have made it possible to transfer the Data anywhere in the world, over the web.

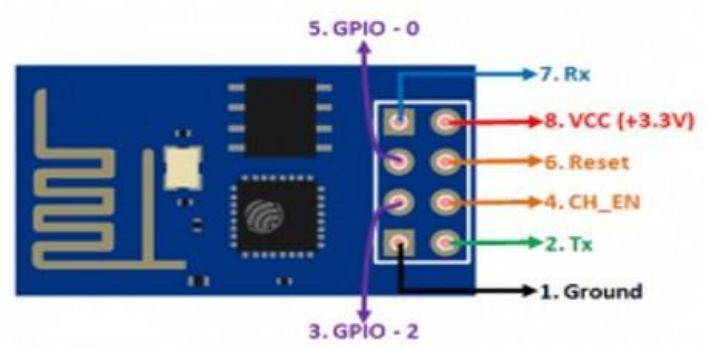
we are building a program to **Send Data to Web using Arduino and Wi-Fi module**. For this we first need an IP address of either Global or Local server, here for the ease and demonstration purpose, we are using Local Server.

Components Required:

- Arduino UNO
- ESP8266 Wi-Fi Module
- USB Cable
- Connecting wires
- Laptop
- Power supply

Pins of ESP8266

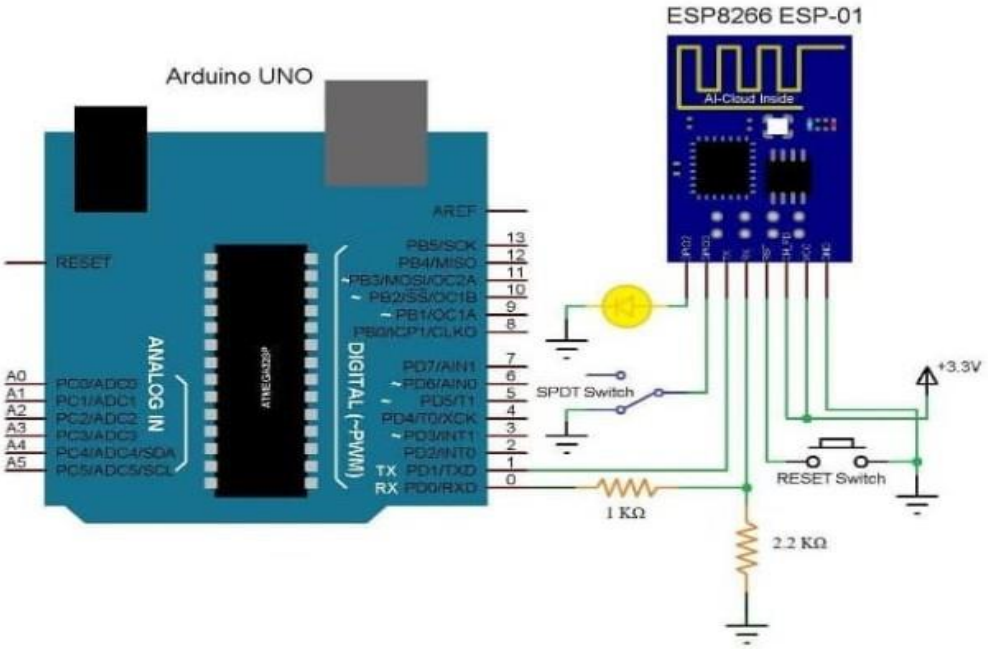
The pinout of the ESP-01 and the corresponding functions of the pins are shown below.



Pin no	*Name*	*Description*
1	VCC	Voltage (+ 3.3 V it can handle up to 3.6 V)
2	GND	Ground (0V)

3	RX	Receive data bit X
4	TX	Transmit data bit X
5	CH_PD	Chip power-down
6	RST	Reset. This is an active-low pin.
7	GPIO 0	General-purpose input/output No. 0. The ESP8266 has two modes: Programming mode and Normal mode. In the programming mode, we upload code/firmware to the device. In Normal mode, we use it for its intended purpose. To enable programming mode, this pin needs to be connected to GND first.
8	GPIO 2	General-purpose input/output No. 2

Circuit Connections:



Circuit Diagram for **"Post Data from Arduino to Web"** is given in above diagram. We mainly need an **Arduino and ESP8266 Wi-Fi module**. ESP8266's Vcc and GND pins are directly connected to 3.3V and GND of Arduino and CH_PD is also connected with 3.3V. Tx and Rx pins of ESP8266 are directly connected to pin 2 and 3 of Arduino. Software Serial Library is used to allow serial communication on pin 2 and 3 of Arduino.

By using Software Serial Library here, we have allowed serial communication on pin 2 and 3, and made them Rx and Tx respectively. By default Pin 0 and 1 of Arduino are used for serial communication but by using Software Serial library, we can allow serial communication on other digital pins of the Arduino.

Working Explanation:

First of all we need to connect our Wi-Fi module to Wi-Fi router for network connectivity. Then we will configure the local server, Send the data to Web and finally Close the connection. This process and commands have been explained in below steps:

1. First we need to test the Wi-Fi module by sending *AT* command, it will revert back a response containing *OK*.

2. After this, we need to select mode using command *AT+CWMODE=mode_id*, we have used Mode id =3. Mode ids:

1 = Station mode (client)
2= AP mode (host)
3 = AP + Station mode (Yes, ESP8266 has a dual mode!)

3. Now we need to disconnect our Wi-Fi module from the previously connected Wi-Fi network, by using the command *AT+CWQAP*, as ESP8266 is default auto connected with any previously available Wi-Fi network

4. After that, user can Reset the module with *AT+RST* command. This step is optional.

5. Now we need to connect ESP8266 to Wi-Fi router using given command

AT+CWJAP="wifi_username","wifi_password"

6. Now get IP Address by using given command:

AT+CIFSR

It will return an IP Address.

7. Now enable the multiplex mode by using *AT+CIPMUX=1* (1 for multiple connection and 0 for single connection)

8. Now configure ESP8266 as server by using *AT+CIPSERVER=1, port_no* (port may be 80). Now your Wi-Fi is ready. Here '1' is used to create the server and '0' to delete the server.

9. Now by using given command user can send data to local created server:

AT+CIPSEND =id, length of data

Id = ID no. of transmit connection

Length = Max length of data is 2 kb

10. After sending ID and Length to the server, we need to send data like : *Serial.println("circuitdigest@gmail.com");*

11. After sending data we need close the connection by given command:

AT+CIPCLOSE=0

Now data has been transmitted to local server.

12. Now type IP Address in Address Bar in web browser and hit enter. Now user can see transmitted data on webpage.

Check the Video below for complete process.

Steps for Programming:

1. Include Software Serial Library for allow serial communication on PIN 2 & 3 and declare some variables and strings.

```
#include<SoftwareSerial.h>
```

```
SoftwareSerial client(2,3); //RX, TX
String webpage="";
int i=0,k=0;
String readString;
int x=0;
boolean No_IP=false;
String IP="";
char temp1='0';
```

2. After this, we have to define some functions for performing our desired tasks.
In *Setup()* function, we initialise inbuilt serial UART communication for ESP8266 as *client.begin(9600);* at the baud rate of 9600.

```
void setup()
{
  Serial.begin(9600);
  client.begin(9600);
  wifi_init();
  Serial.println("System Ready..");
}
```

3. In *wifi_init()* function, we initialize the wifi module by sending some commands like reset, set mode, connect to router, configure connection etc. These commands have also been explained above in description part.

```
void wifi_init()
{
    connect_wifi("AT",100);
    connect_wifi("AT+CWMODE=3",100);
    connect_wifi("AT+CWQAP",100);
    connect_wifi("AT+RST",5000);
    ..... .....
 }
```

4. In *connect_wifi()* function, we send commands data to ESP8266 and then read response from ESP8266 Wi-Fi module.

```
void connect_wifi(String cmd, int t)
{
 int temp=0,i=0;
 while(1)
 {
   Serial.println(cmd);
```

```
..... .....
..... .....
```

5. *sendwebdata( )* function is used for sending data to Local Server or Webpage.

```
void sendwebdata(String webPage)
{
   int ii=0;
    while(1)
    {
     unsigned int l=webPage.length();
     Serial.print("AT+CIPSEND=0,");
     client.print("AT+CIPSEND=0,");
     ..... .....
     ..... .....
```

6. *void send()* function is used for sending data strings to *sendwebdata()* function. That will be further sent to webpage.

```
void Send()
{
     webpage = "<h1>Welcome to Circuit Digest</h1><body bgcolor=f0f0f0>";
     sendwebdata(webpage);
     webpage=name;
     webpage+=dat;
```

7. *get_ip()* function is used for getting IP address of Local created server.

8. In *void loop()* function, we send instruction to user for refreshing the page and check whether the server is connected of not. When user refresh or request the webpage, data automatically transmitted to the same IP address.

```
void loop()
{
  k=0;
  Serial.println("Please Refresh your Page");
  while(k<1000)
  .... .....
  .... .....
```

We can display any data from Arduino to Webpage using this process, like Room Temperature & Humidity, Clock time, GPS coordinates, Heart beat Rate etc.

Raspberry Pi:

Introduction to Raspberry Pi:

Eben is the creator of Raspberry Pi and a co-founder of the Raspberry Pi Foundation. And has invented Raspberry Pi called as a single-board computer The Raspberry Pi Foundation is a charitable organization registered with the Charity Commission for England and Wales

What is a Raspberry Pi?

The Raspberry Pi is a low cost, **credit-card sized computer** that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games. The **Raspberry Pi** is a tiny computer about the size of a deck of cards. It uses what's called a **system on a chip**, which integrates the CPU and GPU in a single integrated circuit, with the RAM, USB ports, and other components soldered onto the board for an all-in-one package.

The Raspberry Pi is a credit-card sized computer designed and manufactured by the Raspberry Pi Foundation, a non-profit organization dedicated to making computers and programming instruction as accessible as possible to the widest number of people. This is great for two reasons, the first is that it provides extremely cheap access to a computer, and second it is a great tool for learning more about computers

What's more, the Raspberry Pi has the ability to interact with the outside world, and has been used in a wide array of digital maker projects, from music machines and parent detectors to weather stations and tweeting birdhouses with infra-red cameras. We want to see the Raspberry Pi being used by kids all over the world to learn to program and understand how computers work.

Why raspberry pi is called raspberry pi?

Raspberry is a reference to a fruit naming tradition in the old days of microcomputers. A lot of computer companies were named after fruit. There's Tangerine Computer Systems, Apricot Computers, and the old British company Acorn, which is a family of fruit.
Pi is because originally we were going to produce a computer that could only really run Python. So the Pi in there is for Python. Now you can run Python on the Raspberry Pi but the design we ended up going with is much more capable than the original we thought of, so it's kind of outlived its name a little bit.

Why Raspberry Pi is more successful than other

Despite plentiful Raspberry Pi alternatives, the Pi remains a top choice among single-board computers. Its main advantages include a low price, tons of community support, lots of Raspberry Pi project ideas, and a bevy of accessories.

Here's why the Raspberry Pi is more successful than the other competitors.

- **Community Support**

From websites such as the **Raspberry Pi Foundation** to the **Raspberry Pi subreddit**, there's a treasure trove of community support available. This includes tutorials, reviews, and more, greatly simplifying troubleshooting. Chances are, if you have a question, someone else has already asked and answered it. We also feature plenty of robust articles and tutorials, such as our guide to **getting started with Raspberry Pi**.

- **Affordability**

Sure, there are plenty of Raspberry Pi alternatives on the market. However, low pricing makes the Pi a top choice. Whereas the Odroid XU4 retails for $62 for the board alone, a kit costs about $90. That's definitely reasonable for the hardware, but the Raspberry Pi is ultra-affordable at $35 for a Raspberry Pi 3 board. The Raspberry Pi Zero and Zero W retail for $5 and $10 respectively. Though less powerful than the Pi 3, the Zero and Zero W are ideal for less processing intensive tasks.

- **Raspberry Pi Models and Choices**

For even further flexibility, consider the many Raspberry Pi board models you can buy. There are different model of Raspberry Pi and there is a lot of choice for selection of Raspberry Pi

- **Ease of Use**

Getting started with the Raspberry Pi is only as difficult as you make it. With an image appropriately named NOOBs that includes a smattering of **Raspberry Pi operating systems**, a Raspberry Pi project can be as basic as a low-powered HTPC. Alternatively, you can create a full-on **wall-mounted Raspberry Pi dashboard**, server, or voice assistant.

- **The Raspberry Pi is a particularly useful device because:**

1. It can connect to the internet in various different languages. Python, Java, JavaScript… most of the popular languages run on the Rpi. Not only that it can also work as a local server for home automation purposes.
2. The Raspberry Pi has low level GPIO pins through which we can directly interface sensors which makes it extremely flexible in deploying systems.
3. The Raspberry Pi is one of the cheapest microcomputer boards with a full blown linux OS on it.

There are many other uses of the Raspberry Pi. However with respect to the IoT standpoint these 3 points standout.

Raspberry Pi board overview

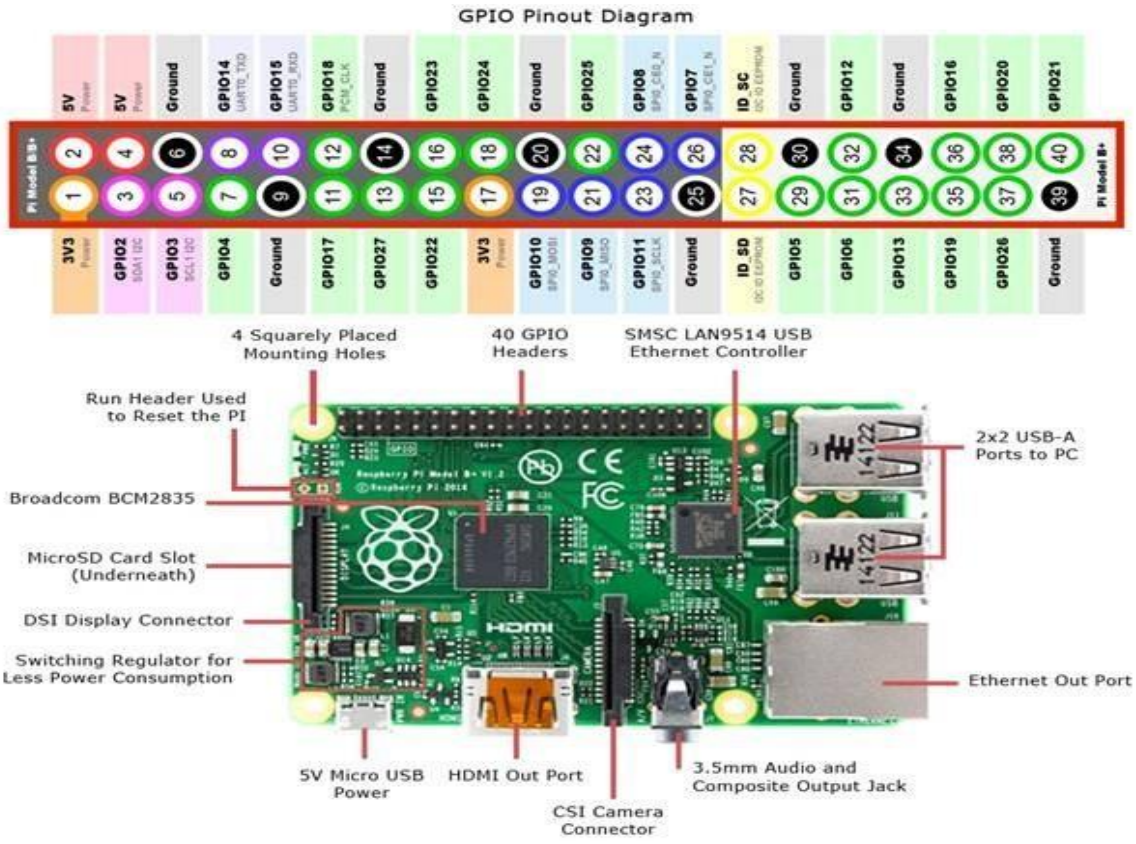
The Raspberry Pi Components

The Raspberry Pi device looks like a motherboard, with the mounted chips and ports exposed (something you'd expect to see only if you opened up your computer and looked at its internal boards), but it has all the components you need to connect input, output, and storage devices and start computing.

You'll encounter two models of the device: **Model A** and **Model B**. The only real differences are the addition of Ethernet and an extra USB port on the more expensive Model B.

Features of Raspberry Pi

- **ARM CPU/GPU** -- This is a Broadcom BCM2835 System on a Chip (SoC) that's made up of an ARM central processing unit (CPU) and a Videocore 4 graphics processing unit (GPU). The CPU handles all the computations that make a computer work (taking input, doing calculations and producing output), and the GPU handles graphics output.
- **GPIO** -- These are exposed general-purpose input/output connection points that will allow the real hardware hobbyists the opportunity to tinker.
- **RCA** -- An RCA jack allows connection of analog TVs and other similar output devices.
- **Audio out** -- This is a standard 3.55-millimeter jack for connection of audio output devices such as headphones or speakers. There is no audio in.
- **LEDs** -- Light-emitting diodes, for all of your indicator light needs.
- **USB** -- This is a common connection port for peripheral devices of all types (including your mouse and keyboard). Model A has one, and Model B has two. You can use a USB hub to expand the number of ports or plug your mouse into your keyboard if it has its own USB port.
- **HDMI** -- This connector allows you to hook up a high-definition television or other compatible device using an HDMI cable.
- **Power** -- This is a 5v Micro USB power connector into which you can plug your compatible power supply.
- **SD card slot** -- This is a full-sized SD card slot. An SD card with an operating system (OS) installed is required for booting the device. They are available for purchase from the manufacturers, but you can also download an OS and save it to the card yourself if you have a Linux machine and the wherewithal.
- **Ethernet** -- This connector allows for wired network access and is only available on the Model B.
- **UART**--The Universal Asynchronous Receiver/ Transmitter is a serial input & output port. That can be used to transfer the serial data in the form of text and it is useful for converting the debugging code.
- **Display**--The connection options of the raspberry pi board are two types such as HDMI and Composite. Many LCD and HD TV monitors can be attached using an HDMI male cable and with a low-cost adaptor. The versions of HDMI are 1.3 and 1.4 are supported and 1.4 version cable is recommended. The O/Ps of the Raspberry Pi audio and video through HMDI, but does not support HDMI I/p. Older TVs can be connected using composite video. When using a composite video connection, audio is available from the 3.5mm jack socket and can be sent to your TV. To send audio to your TV, you need a cable which adjusts from 3.5mm to double RCA connectors.



Basically there are two types of model Raspberry PI Model A and Raspberry PI Model B. The model A only has 1 USB port and no Ethernet port, whereas the model B has 2 USB ports and has an Ethernet port. Model A has 256Mb RAM, one USB port and no Ethernet (network connection). Model B has 512Mb RAM, 2 USB port and an Ethernet port.

Features of Raspberry PI Model A

- The Model A raspberry pi features mainly includes
- 256 MB SDRAM memory
- Single 2.0 USB connector
- Dual Core Video Core IV Multimedia coprocessor
- HDMI (rev 1.3 & 1.4) Composite RCA (PAL and NTSC) Video Out
- 3.5 MM Jack, HDMI, Audio Out
- SD, MMC, SDIO Card slot on board storage
- Linux Operating system
- Broadcom BCM2835 SoC full HD multimedia processor
- 8.6cm*5.4cm*1.5cm dimensions

Features of Raspberry PI Model B

- 512 MB SDRAM memory
- Broadcom BCM2835 SoC full high definition multimedia processor
- Dual Core Video Core IV Multimedia coprocessor
- Single 2.0 USB connector
- HDMI (rev 1.3 and 1.4) Composite RCA (PAL & NTSC) Video Out
- 3.5 MM Jack, HDMI Audio Out

- MMC, SD, SDIO Card slot on board storage
- Linux Operating system
- Dimensions are 8.6cm*5.4cm*1.7cm
- On board 10/100 Ethernet RJ45 jack

All of these Raspberry Pi Models share the following features:

- **GPIO:** 40 pin
- **GPU**: Videocore IV
- **Operating Systems**: Raspbian RaspBMC, Arch Linux, Rise OS, OpenELEC Pidora
- **Ports**: USB 2.0
- **Supported Resolutions**: 640x350 to 1920x1200, including 1080p, PAL & NTSC standards
- **Storage**: Micro USB
- **Video Output**: HDMI Composite RCA

What is SoC available for Raspberry Pi?

All Raspberry Pi models feature a Broadcom BCM2835 SoC consisting if ARM compatible CPU and a Videocore 4 graphics processing unit (GPU). CPU speed ranges from 700 MHz to 1.2 GHz for the Pi 3 and on board memory range from 256 MB to 1 GB RAM. SD cards are used to store the operating systems. Most boards have between one and four USB slots, HDMI composite video output and a 3.5 mm audio jack. Lower level output is provided by several GPIO pins which support common protocols like I²C. The recent B-models have an 8P8C Ethernet port and the Pi 3 and Pi Zero W have Wi-Fi 802.11n and Bluetooth on board.

Raspberry Pi Model A and Model Zero Comparison

Feature	Pi 1 A	Pi 1 A+	Pi Zero V1.2	Pi Zero V1.3	Pi Zero W (Wireless)
Released	Feb 2013	Nov 2014	Nov 2015	May 2016	Feb 2017
Architecture	ARMv6Z 32-bit	ARMv6Z 32-bit	ARMv6Z 32-bit	ARMv6Z 32-bit	ARMv6Z 32-bit
SoC	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835
CPU	700 MHz ARM1176JZF-S	700 MHz ARM1176JZF-S	1 Ghz ARM1176JZF-S	1 Ghz ARM1176JZF-S	1 Ghz ARM1176JZF-S
Cores	1	1	1	1	1
GPU	Broadcom Video Core IV HD 1080p	Broadcom Video Core IV HD 1080p	Broadcom Video Core IV HD 1080p	Broadcom Video Core IV HD 1080p	Broadcom Video Core IV HD 1080p
Memory RAM	256 MB	512 MB	512 MB	512 MB	512 MB
Operating System	Primarily Linux based	Primarily Linux based	Primarily Linux based	Primarily Linux based	Primarily Linux based
USB 2.0	1	1	1 Micro USB	1 Micro USB	1 Micro USB

Ports					
Camera Input	15-pin CSI (Camera Serial Interface)	15-pin CSI (Camera Serial Interface)	None	15-pin CSI (Camera Serial Interface)	15-pin CSI (Camera Serial Interface)
Video Output	HDMI, Composite RCA, DSI	HDMI, Composite TTRS, DSI (Display Serial Interface)	Mini HDMI, Composite on PCB	Mini HDMI, Composite on PCB	Mini HDMI, Composite on PCB
Audio Output	Analog 3.5 mm jack, Digital via HDMI	Analog 3.5 mm jack, Digital via HDMI	Digital via mini-HDMI	Digital via mini-HDMI	Digital via mini-HDMI
Storage	SD slot	Micro SD slot	Micro SD slot	Micro SD slot	Micro SD slot
Ethernet	None	None	None	None	None
Onboard WiFi	None	None	None	None	2.4 Ghz 802.11 b/g/n
Onboard Bluetooth ®	None	None	None	None	4.1 BLE
Input/ Output Pins	26	40	40 (pins not mounted)	40 (pins not mounted)	40 (pins not mounted)
Power (less peripherals)	5v 300 ma	5v 200 ma	5v 300 ma	5v 300 ma	5v 300 ma
Size	85 mm x 56 mm	65 mm x 56 mm	65 mm x 30 mm	65 mm x 30 mm	65 mm x 30 mm

Raspberry Pi Model B and Model Zero Comparison

Feature	Pi 1 B	Pi 1 B+	Pi 2 B	Pi 2 B V1.2	Pi 3 B	Pi 3 B+
Released	Apr 2012	Jul 2014	Feb 2015	Oct 2016	Feb 2016	Mar 2018
Architecture	ARMv6Z 32-bit	ARMv6Z 32-bit	ARMv7-A 32-bit	ARMv8-A 64/32-bit	ARMv8-A 64/32-bit	ARMv8-A 64/32-bit
SoC	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2836	Broadcom BCM2837	Broadcom BCM2837	Broadcom BCM2837B0
CPU	700 MHz ARM1176JZF -S	700 MHz ARM1176JZF-S	900 MHz ARM Cortex-A7	900 MHz ARM Cortex-A53	1.2 GHz ARM Cortex-A53	1.4 GHz ARM Cortex-A53
Cores	1	1	4	4	4	4
GPU	Broadcom Video Core IV HD 1080p	Broadcom Video Core IV HD 1080p	Broadcom Video Core IV HD 1080p	Broadcom Video Core IV HD 1080p	Broadcom Video Core IV HD 1080p	Broadcom Video Core IV HD 1080p

Memory RAM	512 MB	512 MB	1 GB	1 GB	1 GB	1 GB
Operating System	Primarily Linux based	Primarily Linux based	Primarily Linux based	Primarily Linux based	Primarily Linux based	Primarily Linux based
USB 2.0 Ports	2	4	4	4	4	4
Camera Input	15-pin CSI (Camera Serial Interface)	15-pin CSI (Camera Serial Interface)	15-pin CSI (Camera Serial Interface)	15-pin CSI (Camera Serial Interface)	15-pin CSI (Camera Serial Interface)	15-pin CSI (Camera Serial Interface)
Video Output	Composite 3.5 mm RCA and HDMI	HDMI, Composite, DSI (Display Serial Interface)	HDMI, Composite, DSI (Display Serial Interface)	HDMI, Composite, DSI (Display Serial Interface)	HDMI, Composite, DSI (Display Serial Interface)	HDMI, Composite, DSI (Display Serial Interface)
Audio Output	Analog 3.5 mm jack, Digital via HDMI	Analog 3.5 mm jack, Digital via HDMI	Analog 3.5 mm jack, Digital via HDMI	Analog 3.5 mm jack, Digital via HDMI	Analog 3.5 mm jack, Digital via HDMI	Analog 3.5 mm jack, Digital via HDMI
Storage	SD slot	Micro SD slot	Micro SD slot	Micro SD slot	Micro SD slot	Micro SD slot
Ethernet	10/100 Mbps	10/100 Mbps	10/100 Mbps	10/100 Mbps	10/100 Mbps	10/100/1000 (max 300) Mbps
Onboard WiFi	None	None	None	None	2.4 Ghz 802.11 b/g/n	2.4 Ghz and 5 Ghz 802.11 b/g/n/ac
Onboard Bluetooth ®	None	None	None	None	4.1 BLE	4.2 BLE
Input/ Output Pins	26	40	40	40	40	40
Power (less peripherals)	5v 700 ma	5v 320 ma	5v 750 ma	5v 750 ma	5v 850 ma	5v 950 ma
Size	85 mm x 56 mm	85 mm x 56 mm	85 mm x 56 mm	85 mm x 56 mm	85 mm x 56 mm	85 mm x 56 mm

Introduction to Python programming language

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Characteristics of Python

Following are important characteristics of **Python Programming** −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Features of Python programming language

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Python History and Versions

- Python laid its foundation in the late 1980s.
- The implementation of Python was started in the December 1989 by **Guido Van Rossum** at CWI in Netherland.
- In February 1991, van Rossum published the code (labeled version 0.9.0) to alt.sources.
- In 1994, Python 1.0 was released with new features like: lambda, map, filter, and reduce.
- Python 2.0 added new features like: list comprehensions, garbage collection system.
- On December 3, 2008, Python 3.0 (also called "Py3K") was released. It was designed to rectify fundamental flaw of the language.

- *ABC programming language* is said to be the predecessor of Python language which was capable of Exception Handling and interfacing with Amoeba Operating System.
- Python is influenced by following programming languages:
    - ABC language.
    - Modula-3

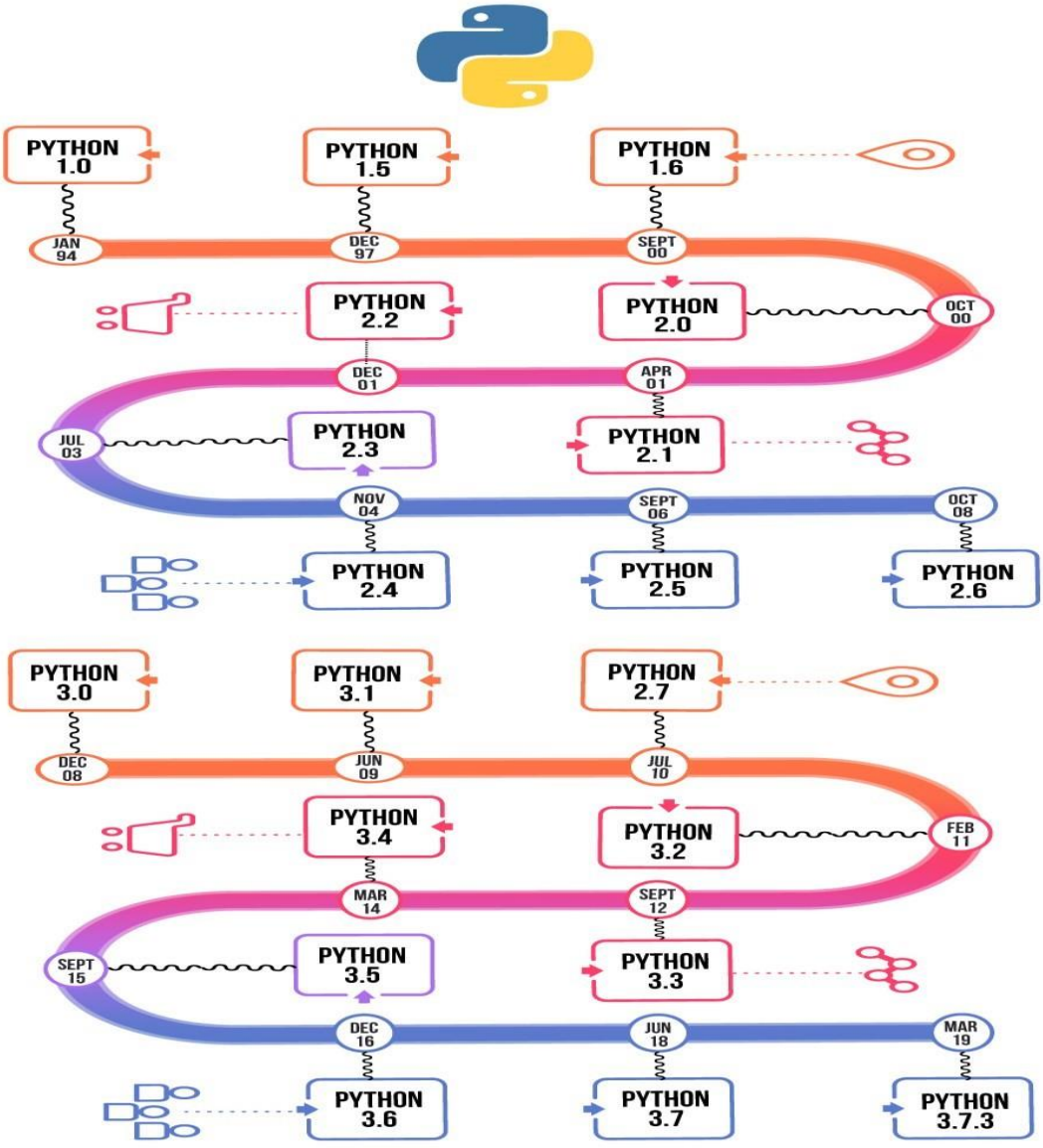
Python Version List

Python programming language is being updated regularly with new features and supports. There are lots of updations in python versions, started from 1994 to current release.

A list of python versions with its released date is given below.

Python Version	Released Date
Python 1.0	January 1994
Python 1.5	December 31, 1997
Python 1.6	September 5, 2000
Python 2.0	October 16, 2000
Python 2.1	April 17, 2001
Python 2.2	December 21, 2001
Python 2.3	July 29, 2003
Python 2.4	November 30, 2004
Python 2.5	September 19, 2006
Python 2.6	October 1, 2008
Python 2.7	July 3, 2010
Python 3.0	December 3, 2008
Python 3.1	June 27, 2009
Python 3.2	February 20, 2011
Python 3.3	September 29, 2012
Python 3.4	March 16, 2014

Python 3.5	September 13, 2015
Python 3.6	December 23, 2016
Python 3.7	June 27, 2018



Modes of Working in Python/Python Programming Mode

In Python, there are two options/methods for running code:

- **Interactive mode**
- **Script mode**

Interactive Mode

Interactive mode, also known as the **REPL** provides us with a quick way of running blocks or a single line of Python code. The code executes via the Python shell, which comes with Python installation. Interactive mode is handy when you just want to execute basic Python commands or you are new to Python programming and just want to get your hands dirty with this beautiful language.
Typically the interactive mode is used to test the features of the python, or to run a smaller script that may not be reusable.

Interactive mode is used when an user wants to run one single line or one block of code. It runs very quickly and gives the output instantly.

Example for Interactive Mode :

Open the command prompt, and go to the location which your python has been installed and hit the python command.

```
chandrashekhar@goka:~$ python
Python 2.7.12 (default, Nov 19 2016, 06:48:10)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> 100+200
300
>>> 10/2
5
>>> i=100
>>> j=500
>>> i*j
50000
>>> exit()
chandrashekhar@goka:~$
```

The >>> indicates that the Python shell is ready to execute and send your commands to the Python interpreter. The result is immediately displayed on the Python shell as soon as the Python interpreter interprets the command.
To run your Python statements, just type them and hit the enter key. You will get the results immediately.

Pros and Cons of Interactive Mode

The following are the advantages of running your code in interactive mode:

1. Helpful when your script is extremely short and you want immediate results.
2. Faster as you only have to type a command and then press the enter key to get the results.
3. Good for beginners who need to understand Python basics.

The following are the disadvantages of running your code in the interactive mode:

1. Editing the code in interactive mode is hard as you have to move back to the previous commands or else you have to rewrite the whole command again.

2. It's very tedious to run long pieces of code.

Script Mode

If you need to write a long piece of Python code or your Python script spans multiple files, interactive mode is not recommended. Script mode is the way to go in such cases. In script mode, You write your code in a text file then save it with a .py extension which stands for "Python". Note that you can use any text editor for this, including Sublime, Atom, notepad++, etc.

If you are in the standard Python shell, you can click "File" then choose "New" or simply hit "Ctrl + N" on your keyboard to open a blank script in which you can write your code. You can then press "Ctrl + S" to save it.

After writing your code, you can run it by clicking "Run" then "Run Module" or simply press F5.

Let us create a new file from the Python shell and give it the name "hello.py". We need to run the "Hello World" program. Add the following code to the file:

```
print("Hello World")
```

Click "Run" then choose "Run Module". This will run the program:

Output

```
Hello World
```

Pros and Cons of Script Mode

The following are the advantages of running your code in script mode:

1. It is easy to run large pieces of code.
2. Editing your script is easier in script mode.
3. Good for both beginners and experts.

The following are the disadvantages of using the script mode:

1. Can be tedious when you need to run only a single or a few lines of cod.
2. You must create and save a file before executing your code.

Key Differences Between Interactive and Script Mode

Here are the key differences between programming in interactive mode and programming in script mode:

1. In script mode, a file must be created and saved before executing the code to get results. In interactive mode, the result is returned immediately after pressing the enter key.
2. In script mode, you are provided with a direct way of editing your code. This is not possible in interactive mode.

Interactive mode	Script mode
A way of using the Python interpreter by typing commands and expressions at the prompt.	A way of using the Python interpreter to read and execute statements in a script.
Cant save and edit the code	Can save and edit the code
If we want to experiment with the code, we can use interactive mode.	If we are very clear about the code, we can use script mode.
we cannot save the statements for further use and we have to retype all the statements to re-run them.	we can save the statements for further use and we no need to retype all the statements to re-run them.
We can see the results immediately.	We cant see the code immediately.

The basic differences between these two modes are as follows:

- Interactive mode is used when an user wants to run one single line or one block of code. It runs very quickly and gives the output instantly.
- Script Mode, on the other hand , is used when the user is working with more than one single code or a block of code.
- It takes more time to compile. In this mode, one can simply write all the lines of code in a text file.

Literal Constants:

A **literal constant** or simply a **literal** is a value, such as a number, character, or string that may be assigned to a variable or symbolic constant, used as an operand in an arithmetic or logical operation, or as a parameter to a function.

- Literals are "hard coded" values such as the number 5, the character 'A', or the string, "Hello, world!".
- Numeric literals may be represented in a variety of formats (decimal, hexadecimal, binary, floating point, etc.)
- A literal always represents the same value (5 always represents the quanty of five)

Literals can be defined as a data that is given in a variable or constant.

Types of Literal used in Python:

1) Numerical literal:

Numeric literals can belong to following four different numerical types.

Integer (signed integers)	Long (long integers)	Float (floating point)	Complex (complex)
Numbers(can be both positive and negative)	Integers of unlimited size followed by	Real numbers with both integer and	In the form of a+bj where a forms the real part and b

with no fractional part.eg: 100	lowercase or uppercase L eg: 87032845L	fractional part eg: -26.2	forms the imaginary part of complex number. eg: 3.14j

Examples:

```
a = 346   # it is integer type literal
pi = 3.14  # it is floating type literal
x = 6.28j  # it is complex type literal
```

2) String literals:

A string literal is a sequence of characters surrounded by quotes. We can use both single, double or triple quotes for a string. And, a character literal is a single character surrounded by single or double quotes.

Examples:

str1 = "India"

str2 = " Savitaribai Phule Pune University, Pune"

3) Boolean literals

A Boolean literal can have any of the two values: True or False.

Examples:

```
x = (1 == True)
y = (1 == False)
a = True + 4
b = False + 10

print("x is", x)
print("y is", y)
print("a:", a)
print("b:", b)
```

Output

```
x is True
y is False
a: 5
b: 10
```

4) Special literals.

Python contains one special literal i.e., None.

None is used to specify to that field that is not created. It is also used for end of lists in Python.

Example:

```
1. >>> val1=10
2. >>> val2=None
3. >>> val1
4. 10
5. >>> val2
```

6. >>> **print** val2
7. None
8. >>>

Constants used in Python:

A constant is a type of variable whose value cannot be changed. It is helpful to think of constants as containers that hold information which cannot be changed later.

By definition, a constant is a type of variable whose value cannot be changed. They are usually declared in a module where a module is a file that can contain variables, functions etc. which is imported to the main file.

Let's see an example of using constants.

```
# In a file constant.py, define following constants
PI = 3.14
GRAVITY = 9.8

# In another file main.py. import the constant and use
import constant
print(constant.PI)
print(constant.GRAVITY)
```

Python Variables

A variable is a named location used to store data in the memory. It is helpful to think of variables as a container that holds data which can be changed later throughout programming. Variable names can be a group of both letters and digits, but they have to begin with a letter or an underscore.

For example,

```
1. number = 10
2. number = 1.1
```

Initially, the value of number was 10. Later it's changed to 1.1.

Variable also known as identifier and used to hold value.

Variable/Identifier Naming

Variables are the example of identifiers. An Identifier is used to identify the literals used in the program. The rules to name an identifier are given below.

- The first character of the variable must be an alphabet or underscore (_).
- All the characters except the first character may be an alphabet of lower-case(a-z), upper-case (A-Z), underscore or digit (0-9).
- Identifier name must not contain any white-space, or special character (!, @, #, %, ^, &, *).
- Identifier name must not be similar to any keyword defined in the language.
- Identifier names are case sensitive for example my name, and MyName is not the same.
- Examples of valid identifiers : a123, _n, n_9, etc.

- Examples of invalid identifiers: 1a, n%4, n 9, etc.

Rules and Naming Convention for Variables and constants

1. Constant and variable names should have a combination of letters in lowercase (a to z) or uppercase (A to Z) or digits (0 to 9) or an underscore (_). For example:

```
snake_case
MACRO_CASE
camelCase
CapWords
```

2. Create a name that makes sense. For example, vowel makes more sense than v.
3. If you want to create a variable name having two words, use underscore to separate them. For example:

```
my_name
current_salary
```

4. Use capital letters possible to declare a constant. For example:

```
PI
G
MASS
SPEED_OF_LIGHT
TEMP
```

5. Never use special symbols like !, @, #, $, %, etc.
6. Don't start a variable name with a digit.

Data Types in Python:

Variables can hold values of different data types. Python is a dynamically typed language hence we need not define the type of the variable while declaring it. The interpreter implicitly binds the value with its type.

Python enables us to check the type of the variable used in the program. Python provides us the **type()** function which returns the type of the variable passed.

Standard data types

A variable can hold different types of values. For example, a person's name must be stored as a string whereas its id must be stored as an integer.

Python provides various standard data types that define the storage method on each of them. The data types defined in Python are given below.

1. **Numeric**
2. **String**
3. **List**
4. **Tuple**
5. **Dictionary**

1) Numbers

Number stores numeric values. Python creates Number objects when a number is assigned to a variable. **For example;**

```
1. a = 3 , b = 5  #a and b are number objects
```

Python supports 4 types of numeric data. (With examples)

1. Integer (signed integers like 10, 2, 29, etc.)
2. Long (long integers used for a higher range of values like 908090800L, -0x1929292L, etc.)
3. Float (float is used to store floating point numbers like 1.9, 9.902, 15.2, etc.)
4. Complex (complex numbers like 2.14j, 2.0 + 2.3j, etc.)

   Python allows us to use a lower-case L to be used with long integers. However, we must always use an upper-case L to avoid confusion.

A complex number contains an ordered pair, i.e., x + iy where x and y denote the real and imaginary parts respectively).

2) String

The string can be defined as the sequence of characters represented in the quotation marks. In python, we can use single, double, or triple quotes to define a string.

String handling in python is a straightforward task since there are various inbuilt functions and operators provided.

In the case of string handling, the operator + is used to concatenate two strings as the operation *"hello"+" python"* returns *"hello python"*.

The operator * is known as repetition operator as the operation "Python " *2 returns "Python Python ".

The following example illustrates the string handling in python.

```
1. str1 = 'hello javatpoint' #string str1
2. str2 = ' how are you' #string str2
3. print (str1[0:2]) #printing first two character using slice operator
```

4. **print** (str1[4]) #printing 4th character of the string
5. **print** (str1*2) #printing the string twice
6. **print** (str1 + str2) #printing the concatenation of str1 and str2

Output:

he

l

hello javatpointhello javatpoint

hello javatpoint how are you

3) List

Lists are similar to arrays in C. However; the list can contain data of different types. The items stored in the list are separated with a comma (,) and enclosed within square brackets [].

We can use slice [:] operators to access the data of the list. The concatenation operator (+) and repetition operator (*) works with the list in the same way as they were working with the strings.

Consider the following example.

1. l = [1, "hi", "python", 2]
2. **print** (l[3:]);
3. **print** (l[0:2]);
4. **print** (l);
5. **print** (l + l);
6. **print** (l * 3);

Output:

[2]

[1, 'hi']

[1, 'hi', 'python', 2]

[1, 'hi', 'python', 2, 1, 'hi', 'python', 2]

[1, 'hi', 'python', 2, 1, 'hi', 'python', 2, 1, 'hi', 'python', 2]

4) Tuple

A tuple is similar to the list in many ways. Like lists, tuples also contain the collection of the items of different data types. The items of the tuple are separated with a comma (,) and enclosed in parentheses ().

A tuple is a read-only data structure as we can't modify the size and value of the items of a tuple.

Let's see a simple example of the tuple.

```
1. t  = ("hi", "python", 2)
2. print (t[1:]);
3. print (t[0:1]);
4. print (t);
5. print (t + t);
6. print (t * 3);
7. print (type(t))
8. t[2] = "hi";
```

Output:

```
('python', 2)
('hi',)
('hi', 'python', 2)
('hi', 'python', 2, 'hi', 'python', 2)
('hi', 'python', 2, 'hi', 'python', 2, 'hi', 'python', 2)
<type 'tuple'>
Trace back (most recent call last):
  File "main.py", line 8, in <module>
    t[2] = "hi";
Type Error: 'tuple' object does not support item assignment
```

5) Dictionary

Dictionary is an ordered set of a key-value pair of items. It is like an associative array or a hash table where each key stores a specific value. Key can hold any primitive data type whereas value is an arbitrary Python object.

The items in the dictionary are separated with the comma and enclosed in the curly braces {}.

Consider the following example.

```
1. d = {1:'Jimmy', 2:'Alex', 3:'john', 4:'mike'};
2. print("1st name is "+d[1]);
3. print("2nd name is "+ d[4]);
4. print (d);
5. print (d.keys());
6. print (d.values());
```

Output:

```
1st name is Jimmy
2nd name is mike
{1: 'Jimmy', 2: 'Alex', 3: 'john', 4: 'mike'}
[1, 2, 3, 4]
['Jimmy', 'Alex', 'john', 'mike']
```

Python Operators

The operator can be defined as a symbol which is responsible for a particular operation between two operands. Operators are the pillars of a program on which the logic is built in a particular programming language. Python provides a variety of operators described as follows.

- Arithmetic operators
- Comparison operators
- Assignment Operators
- Logical Operators
- Bitwise Operators
- Membership Operators
- Identity Operators

Arithmetic operators:

Arithmetic operators are used to perform arithmetic operations between two operands. It includes +(addition), - (subtraction), *(multiplication), /(divide), %(reminder), //(floor division), and exponent (**).

Consider the following table for a detailed explanation of arithmetic operators.

Operator	Description
+ (Addition)	It is used to add two operands. For example, if a = 20, b = 10 => a+b = 30
- (Subtraction)	It is used to subtract the second operand from the first operand. If the first operand is less than the second operand, the value result negative. For example, if a = 20, b = 10 => a - b = 10
/ (divide)	It returns the quotient after dividing the first operand by the second operand. For example, if a = 20, b = 10 => a/b = 2
*** (Multiplication)**	It is used to multiply one operand with the other. For example, if a = 20, b = 10 => a * b = 200
% (reminder)	It returns the reminder after dividing the first operand by the second operand. For example, if a = 20, b = 10 => a%b = 0
**** (Exponent)**	It is an exponent operator represented as it calculates the first operand power to second operand.
// (Floor division)	It gives the floor value of the quotient produced by dividing the two operands.

Comparison operator:

Comparison operators are used to comparing the value of the two operands and returns boolean true or false accordingly. The comparison operators are described in the following table.

Operator	Description
==	If the value of two operands is equal, then the condition becomes true.
!=	If the value of two operands is not equal then the condition becomes true.
<=	If the first operand is less than or equal to the second operand, then the condition becomes true.
>=	If the first operand is greater than or equal to the second operand, then the condition becomes true.
>	If the first operand is greater than the second operand, then the condition becomes true.
<	If the first operand is less than the second operand, then the condition becomes true.

Python assignment operators:

The assignment operators are used to assign the value of the right expression to the left operand. The assignment operators are described in the following table.

Operator	Description
=	It assigns the value of the right expression to the left operand.
+=	It increases the value of the left operand by the value of the right operand and assign the modified value back to left operand. For example, if a = 10, b = 20 => a+ = b will be equal to a = a+ b and therefore, a = 30.
-=	It decreases the value of the left operand by the value of the right operand and assign the modified value back to left operand. For example, if a = 20, b = 10 => a- = b will be equal to a = a- b and therefore, a = 10.
*=	It multiplies the value of the left operand by the value of the right operand and assign the modified value back to left operand. For example, if a = 10, b = 20 => a* = b will be equal to a = a* b and therefore, a = 200.
%=	It divides the value of the left operand by the value of the right operand and assign the reminder back to left operand. For example, if a = 20, b = 10 => a % = b will be equal

	to a = a % b and therefore, a = 0.
**=	a**=b will be equal to a=a**b, for example, if a = 4, b =2, a**=b will assign 4**2 = 16 to a.
//=	A//=b will be equal to a = a// b, for example, if a = 4, b = 3, a//=b will assign 4//3 = 1 to a.

Bitwise operator

The bitwise operators perform bit by bit operation on the values of the two operands.

For example,

1. **if** a = 7;
2. b = 6;
3. then, binary (a) = 0111
4. binary (b) = 0011
5. hence, a & b = 0011
6. a | b = 0111
7. a ^ b = 0100
8. ~ a = 1000

Operator	Description
& (binary and)	If both the bits at the same place in two operands are 1, then 1 is copied to the result. Otherwise, 0 is copied.
| (binary or)	The resulting bit will be 0 if both the bits are zero otherwise the resulting bit will be 1.
^ (binary xor)	The resulting bit will be 1 if both the bits are different otherwise the resulting bit will be 0.
~ (negation)	It calculates the negation of each bit of the operand, i.e., if the bit is 0, the resulting bit will be 1 and vice versa.
<< (left shift)	The left operand value is moved left by the number of bits present in the right operand.
>> (right shift)	The left operand is moved right by the number of bits present in the right operand.

Logical Operators:

The logical operators are used primarily in the expression evaluation to make a decision. Python supports the following logical operators.

Operator	Description
and	If both the expression are true, then the condition will be true. If a and b are the two expressions, a → true, b → true => a and b → true.
or	If one of the expressions is true, then the condition will be true. If a and b are the two expressions, a → true, b → false => a or b → true.
not	If an expression **a** is true then not (a) will be false and vice versa.

Membership Operators:

Python membership operators are used to check the membership of value inside a Python data structure. If the value is present in the data structure, then the resulting value is true otherwise it returns false.

Operator	Description
in	It is evaluated to be true if the first operand is found in the second operand (list, tuple, or dictionary).
not in	It is evaluated to be true if the first operand is not found in the second operand (list, tuple, or dictionary).

Identity Operators:

Operator	Description
is	It is evaluated to be true if the reference present at both sides point to the same object.
is not	It is evaluated to be true if the reference present at both side do not point to the same object.

Operator Precedence:

The precedence of the operators is important to find out since it enables us to know which operator should be evaluated first. The precedence table of the operators in python is given below.

Operator	Description
**	The exponent operator is given priority over all the others used in the expression.
~ + -	The negation, unary plus and minus.
* / % //	The multiplication, divide, modules, reminder, and floor division.
+ -	Binary plus and minus
>> <<	Left shift and right shift
&	Binary and.
^ |	Binary xor and or
<= < > >=	Comparison operators (less then, less then equal to, greater then, greater then equal to).
<> == !=	Equality operators.
= %= /= //= -= += *= **=	Assignment operators
is is not	Identity operators
in not in	Membership operators
not or and	Logical operators

Python Comments

Comments in Python can be used to explain any program code. It can also be used to hide the code as well.

Comments are the most helpful stuff of any program. It enables us to understand the way, a program works. In python, any statement written along with # symbol is known as a comment. The interpreter does not interpret the comment.

Comment is not a part of the program, but it enhances the interactivity of the program and makes the program readable.

Python supports two types of comments:

1) Single Line Comment:

In case user wants to specify a single line comment, then comment must start with ?#?

Eg:

```
1. # This is single line comment.
2. print "Hello Python"
```

Output:

```
Hello Python
```

2) Multi Line Comment:

Multi lined comment can be given inside triple quotes.

Example :

```
1. """ This
2.     Is
3.     Multipline comment"'
```

Example:

```
1. #single line comment
2. print "Hello Python"
3. """This is
4. multiline comment"'
```

Statements and Expressions:

Statements: A **statement** is an instruction that the Python interpreter can execute**. A** statement **is a unit of code that the Python interpreter can execute.**

In computer programming, a **statement** is the smallest standalone element of an imperative programming language that expresses some action to be carried out.

Expressions: An **expression** is a combination of values, variables, operators, and calls to functions. An **expression** is an instruction that *combines values and operators* and *always evaluates down to a single value*. An expression is a combination of values, variables, and operators which always returns value, where as statements never returns any value.

An expression **in a programming language is a combination of one or more explicit values, constants, variables, operators and functions that the programming language interprets and computes to produce another value.**

Expression Vs Statement

- Expression
  - Expressions always returns a value
  - Functions are also expressions. Even a non returning function will still return *None* value, so it is an expression.
  - Can print the result value
  - Examples Of Python Expressions: “Hello” + “World”, 4 + 5 etc.
- Statement
  - A statement never returns a value
  - Cannot print any result
  - Examples Of Python Statements: Assignment statements, conditional branching, loops, classes, import, def, try, except, pass, del etc

Order of Execution:

When more than one operator appears in the expression, the order of evaluation depends on the rule of precedence.

The acronym **PEMDAS** is useful to remember the order of evaluation.

1. **P:** parentheses have highest precedence hence solved first
2. **E:** Exponentiation has next highest precedence.
3. **MDAS:** Multiplication and division have same precedence, but higher than addition and subtraction. The addition and subtraction also have same precedence.
4. Operators with same precedence evaluated form left to right

Types of Expression in Python

Infix notation: X + Y

Operators are written in-between their operands. This is the usual way we write expressions.

Postfix notation (also known as "Reverse Polish notation"): X Y +

Operators are written after their operands

Prefix notation (also known as "Polish notation"): + X Y

Operators are written before their operands.

Examples of Infix, Prefix, and Postfix		
Infix Expression	**Prefix Expression**	**Postfix Expression**
A + B	+ A B	A B +
A + B * C	+ A * B C	A B C * +

Python Decision Making Statements – Python If, If-else, Nested Statements

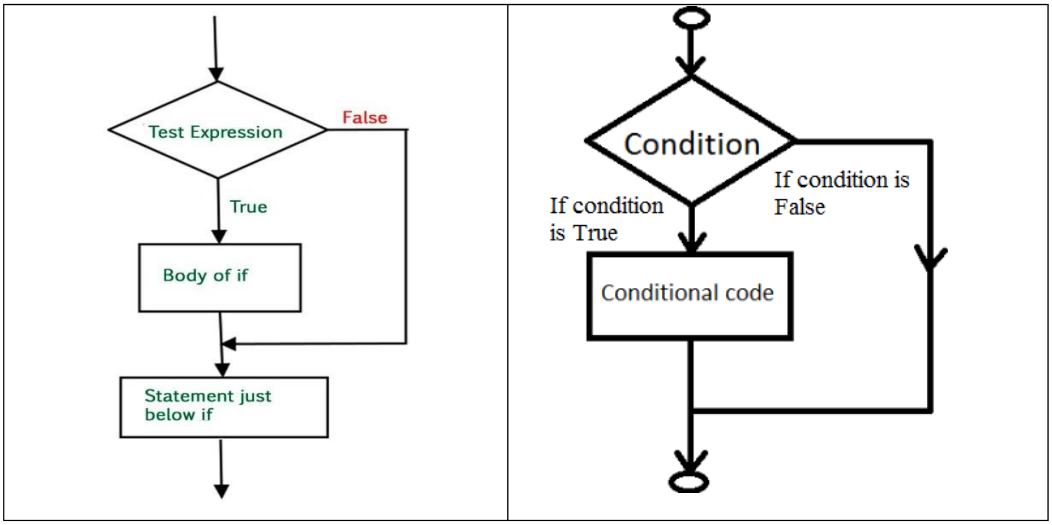
1. Python Decision Making Statements:

Syntax:

if *condition*:

  # Statements to execute if

  # condition is true



Example

```
# python program to illustrate If statement
i = 10
if (i > 15):
  print ("10 is less than 15")
print ("15 is greater than 10")
```

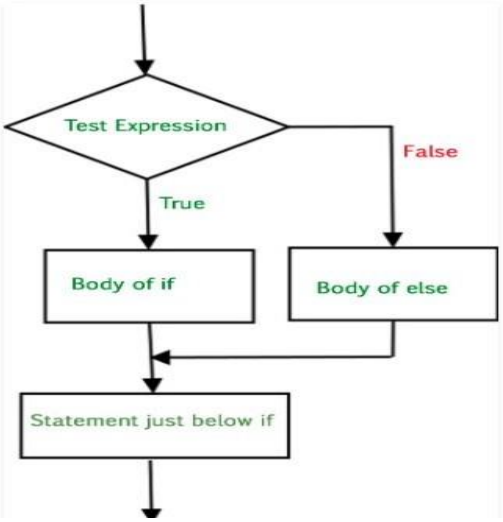
Output:

15 is greater than 10

2. Python if-else Statements

Syntax:

```
if (condition):
    # Executes this block if
    # condition is true
else:
    # Executes this block if
    # condition is false
```



Example:

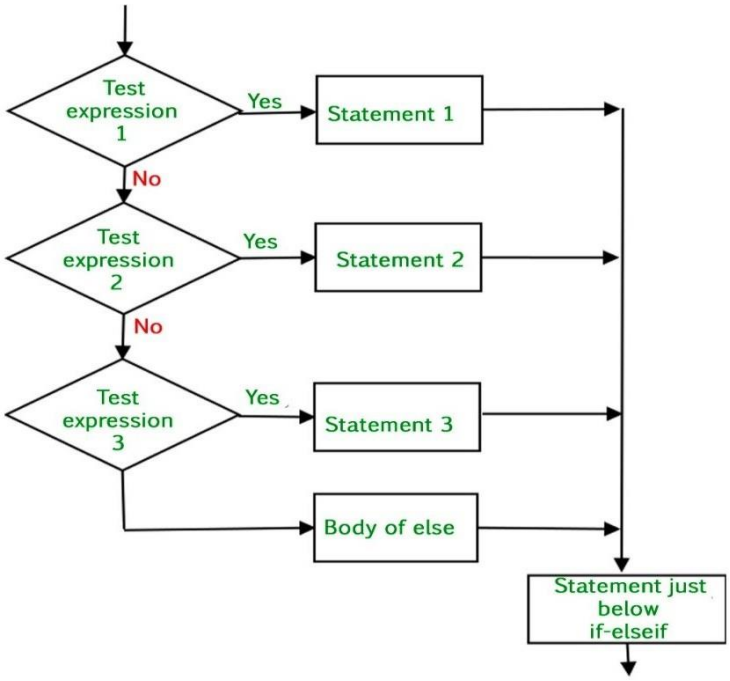
```
# python program to illustrate If else statement
#!/usr/bin/python

i = 20;
if (i < 15):
    print ("i is smaller than 15")
else:
    print ("i is greater than 15")
```

Output:

i is greater than 15

3. Python if-elif-else statement



Example:-

```
# Python program to illustrate if-elif-else ladder
#!/usr/bin/python
i = 20
if (i == 10):
    print ("i is 10")
elif (i == 15):
    print ("i is 15")
elif (i == 20):
    print ("i is 20")
else:
    print ("i is not present")
```

Output:

i is 20

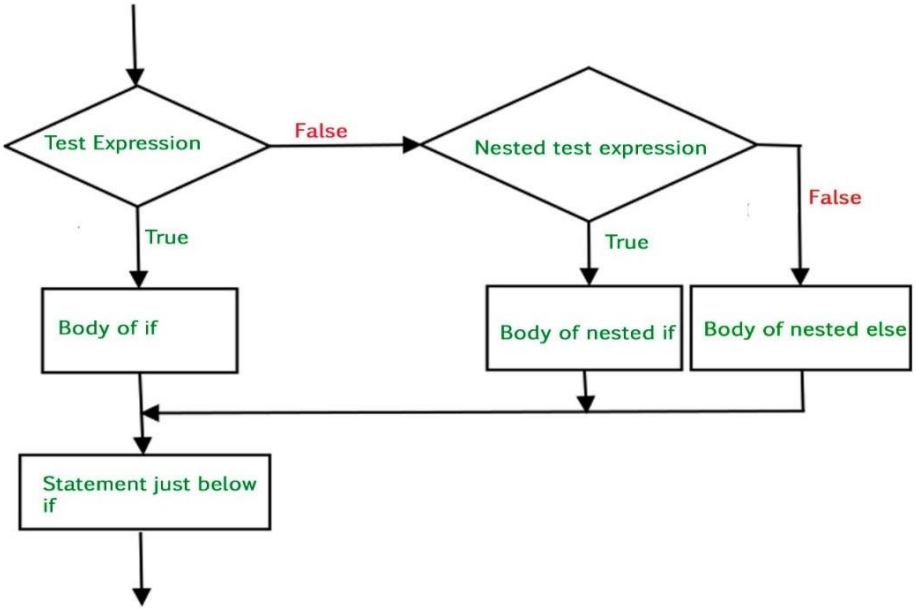
4. Python Nested if-else Statement

Syntax:

```
if (condition1):

   # Executes when condition1 is true
```

```
    if (condition2):
        # Executes when condition2 is true
    # if Block is end here
# if Block is end here
```



Example :

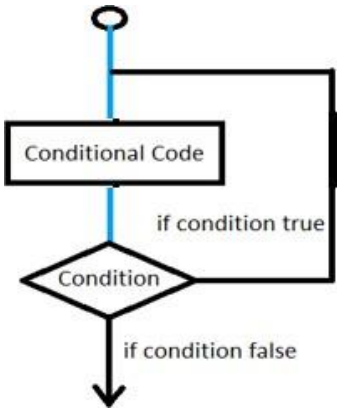
```
# python program to illustrate nested If statement
i = 10
if (i == 10):
    #  First if statement
    if (i < 15):
        print ("i is smaller than 15")
    # Nested - if statement
    # Will only be executed if statement above
    # it is true
    if (i < 12):
        print ("i is smaller than 12 too")
    else:
        print ("i is greater than 15")
```

Output:

i is smaller than 15

Python Loop Statements:

Looping statements in python are used to execute a block of statements or code repeatedly for several times as specified by the user.



Python provides us with 2 types of loops as stated below:

- While loop
- For loop

1) While loop:

While loop in python is used to execute multiple statement or codes repeatedly until the given condition is true.

We use while loop when we don't know the number of times to iterate.

Syntax:

while Condition:
Statements
exit condition

Example 1:

```
umber = 5
sum = 0
i = 0
while (i<number):
      sum = sum + i
      i = i+1
 print(sum)
```

Output:
10

For example 2:

```
print("Print hello 10 time from the help of while loop")
a=int(1)
while a<=10
   print("Hello", a)
   a=a+1
```

Output:

Hello 1

Hello 2

Hello 3

Hello 4

Hello 5

Hello 6

Hello 7

Hello 8

Hello 9

Hello 10

2. The for loop:

Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable.

Syntax:
for Condition:

Statements

For example:

```
for i in range(1,5):
    print(i)
    print("\n")
```

Output:

1
2
3
4

3. The nested loop:

You can use one or more loop inside any another while, or for loop.

For example:

```
for i in range(1,5):
    for j in range(i,5):
        print(i);
    print("\n")
```

Output:
1
1
1

1

2
2
2

3
3

4

Loop Control Statements

The Loop control statements change the execution from its normal sequence. When the execution leaves a scope, all automatic objects that were created in that scope are destroyed.
Python supports the following control statements.

1. Break statement
2. Continue statement
3. Pass statement

1. Break statement:

The **break** statement in Python terminates the current loop and resumes execution at the next statement, just like the traditional break found in C.
The most common use for break is when some external condition is triggered requiring a hasty exit from a loop. The **break** statement can be used in both *while* and *for* loops.

Example 1

```
1. list =[1,2,3,4]
2.  count = 1;
3.  for i in list:
4.      if i == 4:
5.          print("item matched")
6.          count = count + 1;
7.          break
8.  print("found at",count,"location");
```

Output:

```
item matched
found at 2 location
```

Example 2

```
for letter in 'geeksforgeeks':
    # break the loop as soon it sees 'e' or 's'
    if letter == 'e' or letter == 's':
        break
  print 'Current Letter :', letter
```

Output:

Current Letter : e

2. Continue statement:

The **continue** statement in Python returns the control to the beginning of the while loop. The **continue** statement rejects all the remaining statements in the current iteration of the loop and moves the control back to the top of the loop.

The **continue** statement can be used in both *while* and *for* loops.

Example 1:

```
# Prints all letters except 'e' and 's'
for letter in 'geeksforgeeks':
   if letter == 'e' or letter == 's':
      continue
   print 'Current Letter :', letter
   var = 10
```

Output:

Current Letter : g
Current Letter : k
Current Letter : f
Current Letter : o
Current Letter : r
Current Letter : g
Current Letter : k

Example 2:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    continue
print(x)
```

Output

Apple
Cherry

3. Pass statement:

The **pass** statement in Python is used when a statement is required syntactically but you do not want any command or code to execute.

The **pass** statement is a *null* operation; nothing happens when it executes. The **pass** is also useful in places where your code will eventually go, but has not been written yet (e.g., in stubs for example):

We use pass statement to write empty loops. Pass is also used for empty control statement, function and classes.

Example:

```
# An empty loop
for letter in 'geeksforgeeks':
    pass
print 'Last Letter :', letter
```

Output:

Last Letter : s

Example:

```
# !/usr/bin/python
for letter in 'Python':
   if letter == 'h':
      pass
      print 'This is pass block'
   print 'Current Letter :', letter

print "Good bye!"
```

This will produce following result:

Current Letter : P
Current Letter : y
Current Letter : t
This is pass block
Current Letter : h
Current Letter : o
Current Letter : n
Good bye!

Program for addition for two numbers

Source Code

```
# This program adds two numbers
num1 = 1.5
num2 = 6.3
# Add two numbers
sum = float(num1) + float(num2)
# Display the sum
print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))
```

Output

The sum of 1.5 and 6.3 is 7.8

Python Program to Find the Square Root

Example: For positive numbers

```
number = int(input("enter a number: "))
sqrt = number ** 0.5
print("square root:", sqrt)
```

Output:
enter a number: 64
square root: 8.0

Link for Python program

https://www.programiz.com/python-programming/examples
https://www.geeksforgeeks.org/python-programming-examples/
https://www.javatpoint.com/python-programs

Python Program to Check Prime Number

```
num = int(input("enter a number: "))
 for i in range(2, num):
        if num % i  == 0:
                print("not prime number")
                break
else:
        print("prime number")
```

Output
enter a number: 11
prime number

Python Program for factorial of a number

Factorial of a non-negative integer, is multiplication of all integers smaller than or equal to n.
For example factorial of 6 is 6*5*4*3*2*1 which is 720.
Note: The factorial value of 0 is 1 always. (Rule violation)
Python Program to find Factorial of a Number

```
number = int(input(" Please enter any Number to find factorial : "))
fact = 1

for i in range(1, number + 1):
    fact = fact * i
print("The factorial of %d  = %d" %(number, fact))
```

OUTPUT
Please enter any Number to find factorial : 4
The factorial of 4 = 24

Python Program to Check if a Number is Odd or Even

```
num = int(input("Enter a number: "))
if (num % 2) == 0:
   print("{0} is Even number".format(num))
else:
   print("{0} is Odd number".format(num))
```

Python Program to check if a Number is Positive, Negative or Zero

```
num = float(input("Enter a number: "))
if num > 0:
 print("{0} is a positive number".format(num))
elif num == 0:
   print("{0} is zero".format(num))
else:
   print("{0} is negative number".format(num))
```

Python program to solve quadratic equation

Quadratic equation:
Quadratic equation is made from a Latin term "quadrates" which means square. It is a special type of equation having the form of:
$ax^2+bx+c=0$

```
# import complex math module
import cmath
a = float(input('Enter a: '))
b = float(input('Enter b: '))
c = float(input('Enter c: '))
# calculate the discriminant
d = (b**2) - (4*a*c)
# find two solutions
sol1 = (-b-cmath.sqrt(d))/(2*a)
sol2 = (-b+cmath.sqrt(d))/(2*a)
print('The solution are {0} and {1}'.format(sol1,sol2))
```

Python program to find the area of a triangle

Mathematical formula:

Area of a triangle = (s*(s-a)*(s-b)*(s-c))-1/2

Here s is the semi-perimeter and a, b and c are three sides of the triangle.

```
# Three sides of the triangle is a, b and c:
```

```
a = float(input('Enter first side: '))
b = float(input('Enter second side: '))
c = float(input('Enter third side: '))
# calculate the semi-perimeter
s = (a + b + c) / 2
  # calculate the area
area = (s*(s-a)*(s-b)*(s-c)) ** 0.5
print('The area of the triangle is %0.2f' %area)
```

Python program to convert Celsius to Fahrenheit

Celsius:
Celsius is a unit of measurement for temperature. It is also known as centigrade. It is a SI derived unit used by most of the countries worldwide.

Fahrenheit:
Fahrenheit is also a temperature scale. It is named on Polish-born German physicist Daniel Gabriel Fahrenheit. It uses degree Fahrenheit as a unit for temperature.

Conversion formula:

$$T(°F) = T(°C) \times 9/5 + 32$$

Or,

$$T(°F) = T(°C) \times 1.8 + 32$$

```
# Collect input from the user
celsius = float(input('Enter temperature in Celsius: '))
  # calculate temperature in Fahrenheit
fahrenheit = (celsius * 1.8) + 32
print('%0.1f  Celsius is equal to %0.1f degree Fahrenheit'%(celsius,fahrenheit))
```

Python Program to Print the Fibonacci sequence

Fibonacci sequence:
The Fibonacci sequence specifies a series of numbers where the next number is found by adding up the two numbers just before it.

For example:
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, **and** so on....

Program for Fibonacci sequence

```
nterms = int(input("How many terms you want? "))
# first two terms
n1 = 0
n2 = 1
```

```
count = 2
# check if the number of terms is valid
if nterms <= 0:
   print("Plese enter a positive integer")
elif nterms == 1:
   print("Fibonacci sequence:")
   print(n1)
else:
   print("Fibonacci sequence:")
   print(n1,",",n2,end=', ')
   while count < nterms:
      nth = n1 + n2
      print(nth,end=' , ')
      # update values
      n1 = n2
      n2 = nth
      count += 1
```

Python Program to Find the Sum of Natural Numbers

Natural numbers:
As the name specifies, a natural number is the number that occurs commonly and obviously in the nature. It is a whole, non-negative number.
Some mathematicians think that a natural number must contain 0 and some don't believe this theory. So, a list of natural number can be defined as:

1. N= {0, 1, 2, 3, 4, ... **and** so on}
2. N= {1, 2, 3, 4, ... **and** so on}

Example:

```
num = int(input("Enter a number: "))
 if num < 0:
  print("Enter a positive number")
else:
  sum = 0
  # use while loop to iterate un till zero
  while(num > 0):
     sum += num
     num -= 1
  print("The sum is",sum)
```

Python Program to Calculate Square of a Number

```
number = float(input(" Please Enter any numeric Value : "))
square = number * number
print("The Square of a Given Number {0}  = {1}".format(number, square))
```

Python Program to Calculate Square of a Number

```
number = float(input(" Please Enter any numeric Value : "))
square = number ** 2
print("The Square of a Given Number {0}  = {1}".format(number, square))
```

Python Program to find Cube of a Number

This Python program allows users to enter any numeric value. Next, Python finds a Cube of that number using an Arithmetic Operator.

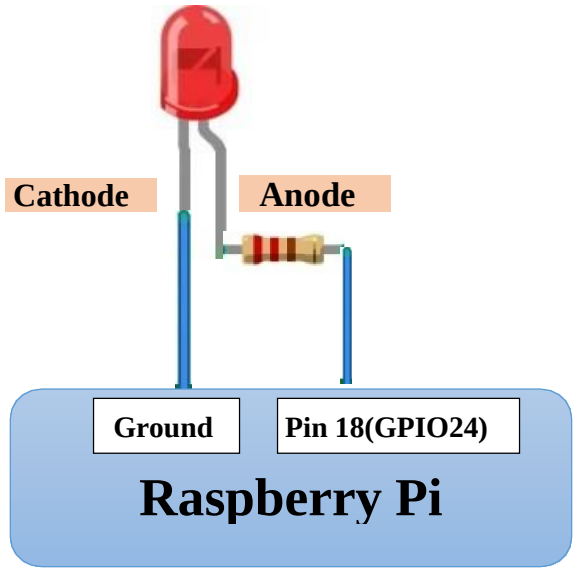
```
# Python Program to Calculate Cube of a Number
number = float(input(" Please Enter any numeric Value : "))
cube = number * number * number
print("The Cube of a Given Number {0}  = {1}".format(number, cube))
```

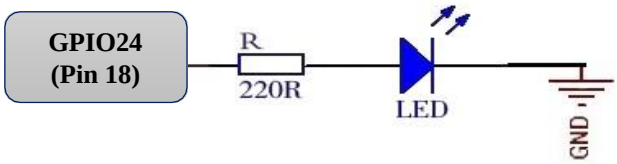
Python Program to Calculate Cube of a Number Example 2

This python cube number example is the same as above, but here, we are using the Exponent operator.

```
number = float(input(" Please Enter any numeric Value : "))
cube = number ** 3
print("The Cube of a Given Number = " (number), “is” (cube))
```

LED interfacing with Raspberry Pi





Anode of LED is connected to pin number 18(GPIO24) and cathode is connected to ground pin of Raspberry Pi.

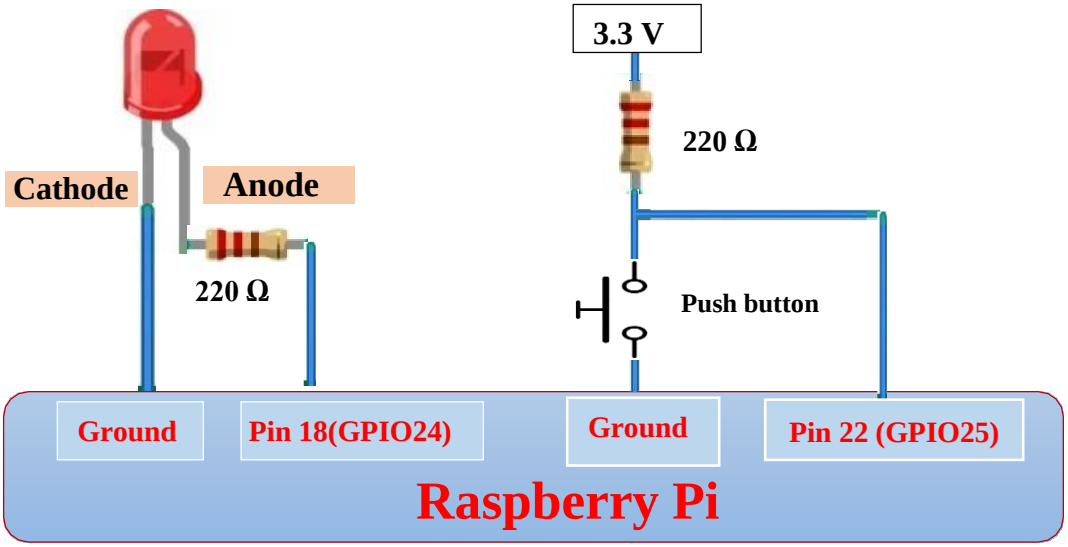
LED Blanking program using Python

```
import time

import RPi.GPIO as GPIO              ## Import GPIO library
GPIO.setmode(GPIO.BOARD)             ## Use board pin numbering
GPIO.setup(18, GPIO.OUT)             ## Setup GPIO Pin 11 to OUT
While True:

        GPIO.output(18,True)         ## Turn on Led
        time.sleep(1)                ## Wait for one second
        GPIO.output(18,False)         ## Turn off Led
        time.sleep(1)                ## Wait for one second
```

Interfacing LED and switch with Raspberry Pi



In interfacing anode of LED is connected to pin 18 (GPIO24) and cathode is connected to ground. The switch is connected to pin 22 (GPIO25) as shown.

Program to toggle LED connected at pin 18 when switch pressed. Switch is connected to pin 22

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(22, GPIO.IN, pull_up_down=GPIO.PUD_UP)     #Button to GPIO25
GPIO.setup(18, GPIO.OUT)  #LED to GPIO24
```

```
try:
   while True:
      button_state = GPIO.input(22)
      if button_state == False:
         GPIO.output(18, True)
         print('Button Pressed...')
         time.sleep(0.2)
      else:
         GPIO.output(18, False)
 except:
     GPIO.cleanup()
```

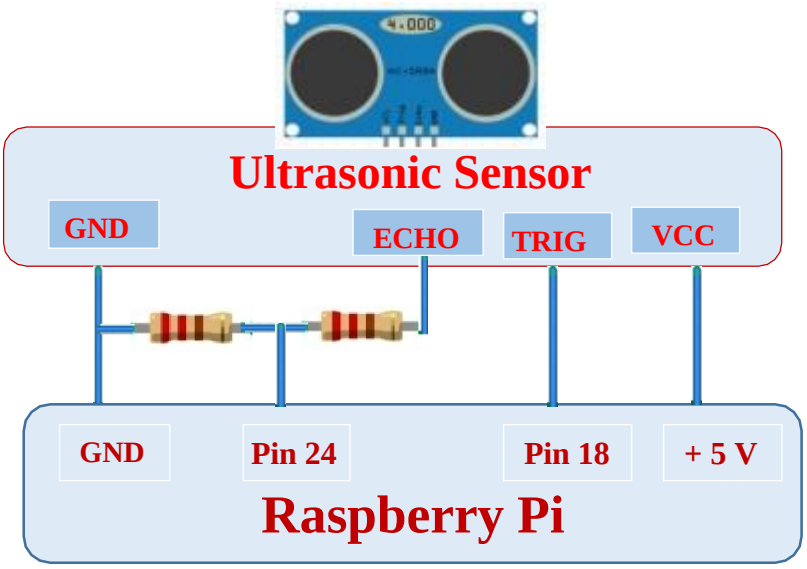
Program to turn on LED connected at pin 18 when switch pressed and turned off. Switch is connected to pin 22

```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)

#Switch Pin
GPIO.setup(22, GPIO.IN)
#LED Pin
GPIO.setup(18, GPIO.OUT)
state=false

def toggleLED(pin):
                    state = not state
                   GPIO.output(pin, state)
while True:
         try:
              if (GPIO.input(22) == True):
                        toggleLED(pin)
               sleep(.01)
               except:
      exit()
```

Interfacing of Ultrasonic sensor with Raspberry Pi



Program for distance measurement using Ultrasonic sensor and Raspberry Pi

```
import RPi.GPIO as GPIO
import time
#GPIO Mode (BOARD / BCM)
GPIO.setmode(GPIO.BCM)

GPIO_TRIGGER = 18

GPIO_ECHO = 24
GPIO.setup(GPIO_TRIGGER, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)
While True:
   GPIO.output(GPIO_TRIGGER, True)        # set Trigger to HIGH
   time.sleep(0.00001)
   GPIO.output(GPIO_TRIGGER, False)       # set Trigger after 0.01ms to LOW
   while GPIO.input(GPIO_ECHO) == 0:
      StartTime = time.time()              # save Start Time
   while GPIO.input(GPIO_ECHO) == 1:
      StopTime = time.time()               # save time of arrival
    TimeElapsed = StopTime – StartTime     # time difference between start and arrival
   distance = (TimeElapsed * 17150)          #Multiply pulse duration by 17150 to get distance
       distance = round(distance, 2)        #Round to two decimal points
       print "Measured Distance =", distance, “cm”
         time.sleep(1)
      GPIO.cleanup()
```

Sending data to cloud using Raspberry Pi

Steps for sending data to cloud using Raspberry Pi

1. Decide which cloud service you want to use and check if they support raspberry pi.
2. Create an account, get the API keys.
3. Install an OS (preferably Raspbian), connect the sensors.
4. Connect the Raspberry Pi to the internet.
5. Browse for samples on the cloud service you're using.
6. Execute the script/program on your Pi.

It completely depends upon the cloud service you're planning to use.

General steps for sending data to cloud using Raspberry Pi

Adding sensors to devices and sending that sensor data to a public cloud offers many advantages; data can be stored, processed and decisions made with relatively little outlay on initial infrastructure. Having multiple sensors connected to a gateway that process and transmit data to a cloud platform is an efficient architecture to implement an IoT system.

1. Hardware and Software required

- Raspberry Pi
- Wifi, or Internet connection
- Python
- Raspbian Debian 9 Stretch, Debian 10 Buster.

2. Normalizing sensor values

The first step after receiving data from a sensor is to normalize the value to a standard range. For example, temperature sensors often output a value using a 16 bit ADC — so a number from 0 to 65,535 is transmitted. However, that number might represent two completely different temperature values from a 0 to 70-degree sensor compared to a -40 to 125-degree sensor. The received data point is converted to an actual temperature value (celsius or fahrenheit can be selected) on the gateway prior to transmission to the cloud.

3. JSON

After the data is normalised, it is put into a JavaScript Object Notation (JSON) format for transmission to the cloud. JSON is an open standard, human-readable, text-enabled way to transmit data and the most commonly used format for sending sensor data over the Internet. To keep the JSON message as short as possible, a short, unique identifier is used for each sensor in the system. On the cloud side, the identifier is expanded into a longer device address that allows users to easily select one sensor from thousands by location, type, or other defined parameters.

4. Provisioning and establishing a cloud connection

Before data is sent to the cloud for the first time, it is necessary to create a Google instance that is listening for the specific gateway. This is done using a unique identifier programmed into the gateway during the manufacturing process. — After the initial login process, the identifier is entered which creates the Google data node ready to receive sensor data from that gateway.

Powering up the gateway device causes a boot sequence which establishes an Internet connection, sets the clock on the Raspberry Pi, and then sends certificate information to the

Google cloud. Once a connection is established, sensor data is immediately read over the BLE connection, normalized, converted to JSON, encrypted, and then transmitted using Google IoT core (MQTT or HTTPS protocols).

5. Cloud activities

On the cloud side, a standard set of features is provided for displaying and processing the data. Once data is received, it is converted to an internal cloud message using a Publish/Subscribe (PUB/SUB) scheme that allows different cloud components to listen for messages from specific sources.

Dashboard

Sensor data is displayed in a customizable dashboard (below). Here both real-time and historical data can be displayed in a variety of user-defined formats selected from a library.

Event / Alerts

As data is received, it can be compared to pre-set conditions and alerts will be triggered if those conditions are met.

Note: The underlying algorithm is a pattern matching (Rete algorithm) scheme that allows for thousands of messages per second to be compared with thousands of possible events.

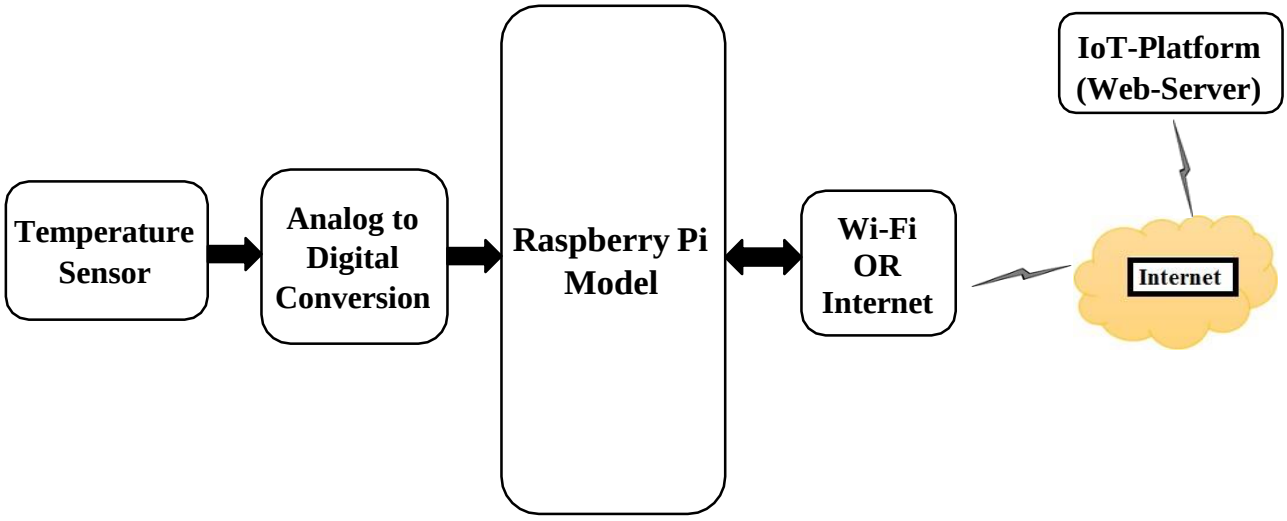
Events that might trigger an alert can include:

- Data meets a trigger value (temperature is over 30 degrees)
- Multiple data trigger points are met (temperature > x AND humidity > y)
- Data has not been received for x period of time
- Data has moved x standard deviations away from the long-term average
- Data is behaving abnormally (stuck on one value or jumping around)

Many more event / alert conditions can be programmed into the cloud engine.

Sending data to ThingSpeak cloud using Raspberry Pi

Example Temperature monitoring system using ThingSpeak cloud and Raspberry Pi



Raspberry Pi as a gateway device for uploading your sensor data on ThingSpeak server.

Step 1: Signup for Thingspeak

Go to www.thinspeak.com

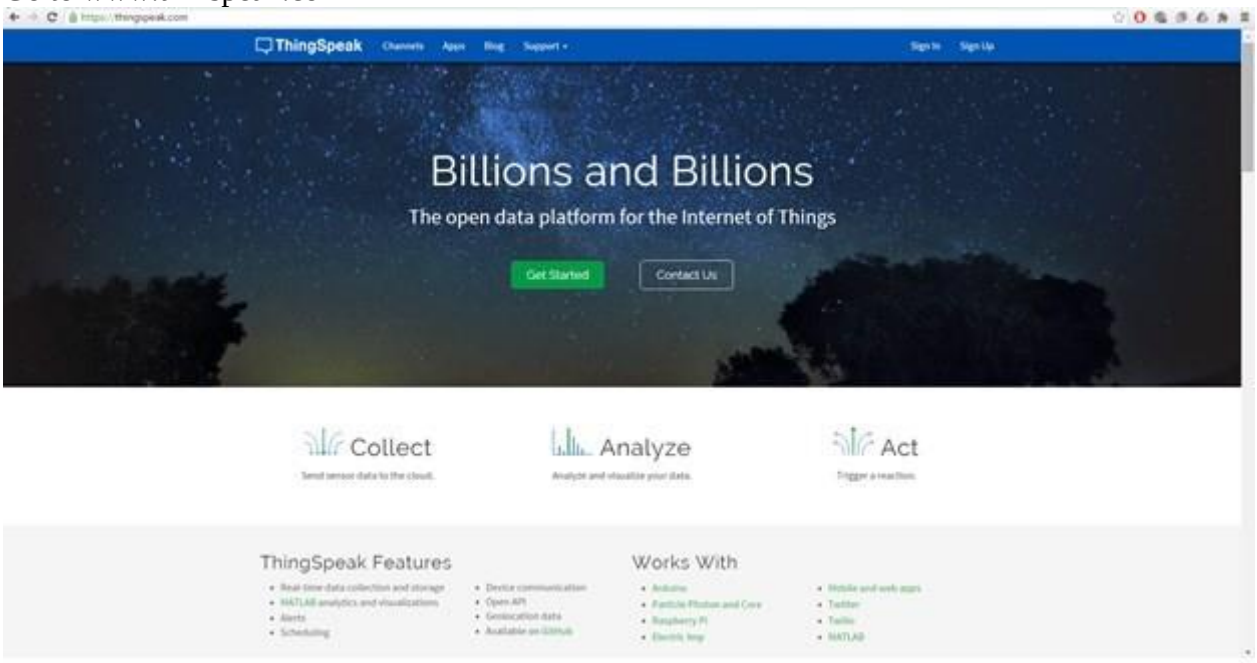


Fig1. Thingspeak Website

Click on “Sign Up” option and complete the details

Fig2. Create user account

Step 2: Create a Channel for Your Data

Once you Sign in after your account activation, Create a new channel by clicking "New Channel" button

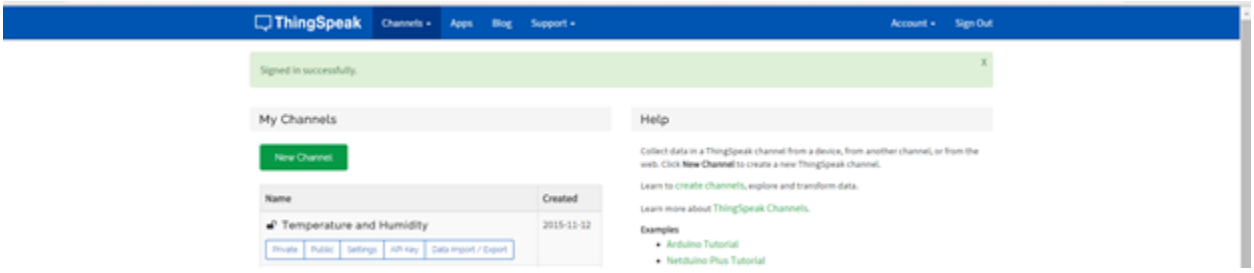


Fig3. Creating New Channel

After the "New Channel" page loads, enter the Name and Description of the data you want to upload. You can enter the name of your data (ex: Temperature) in Field1. If you want more Fields you can check the box next to Field option and enter the corresponding name of your data.

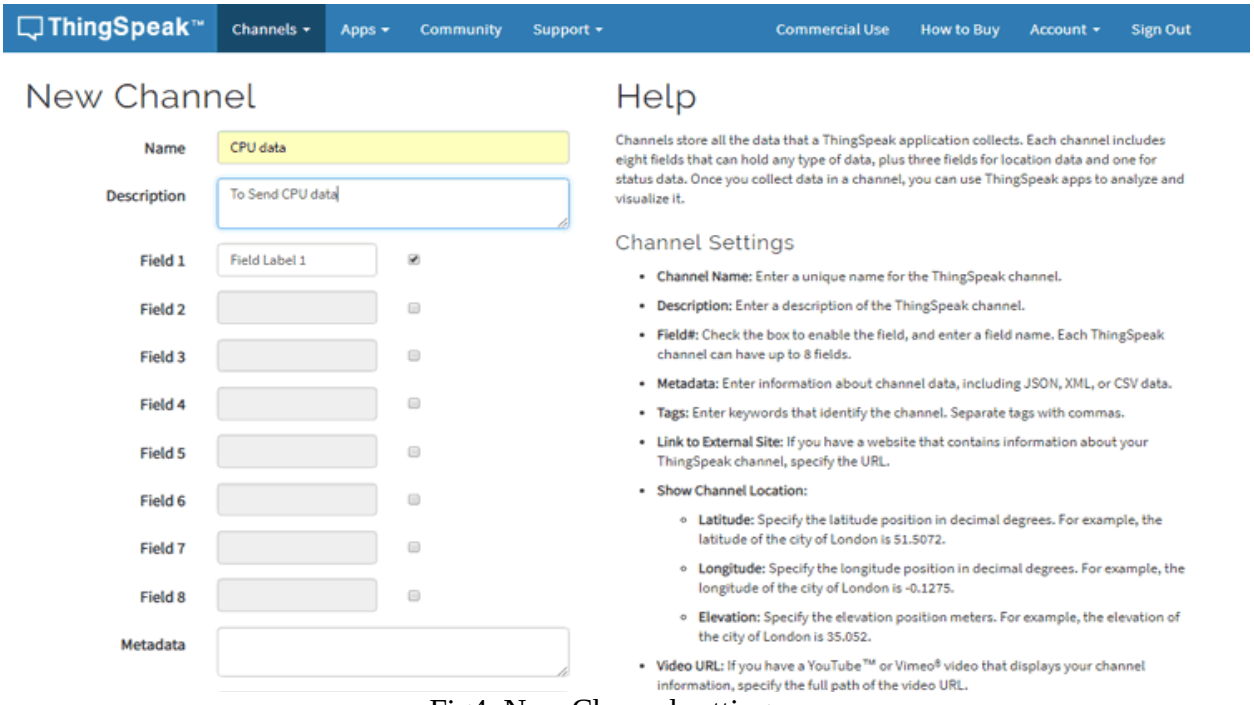


Fig4. New Channel settings

Click on “Save Channel” button to save all of your settings.

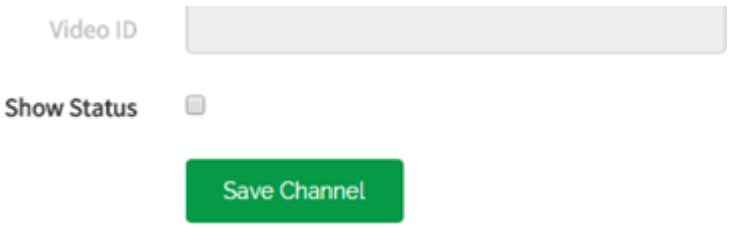


Fig5. Save settings for New Channel

Here we created two Fields, one is CPU Memory and one for CPU Temperature

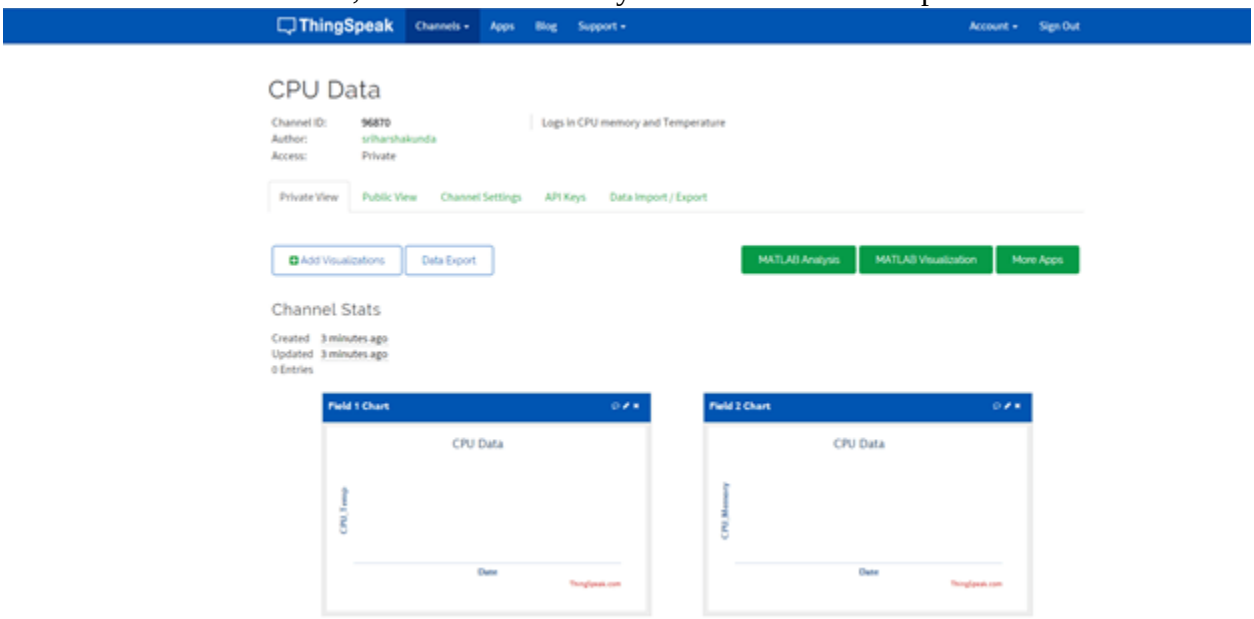


Fig6. Creating field charts to display data

Step 3: Get an API Key

To upload our data, we need an API key, which we will later include in a piece of python code to upload our sensor data to Thingspeak Website.

Click on “API Keys” tab to get the key for uploading your sensor data.

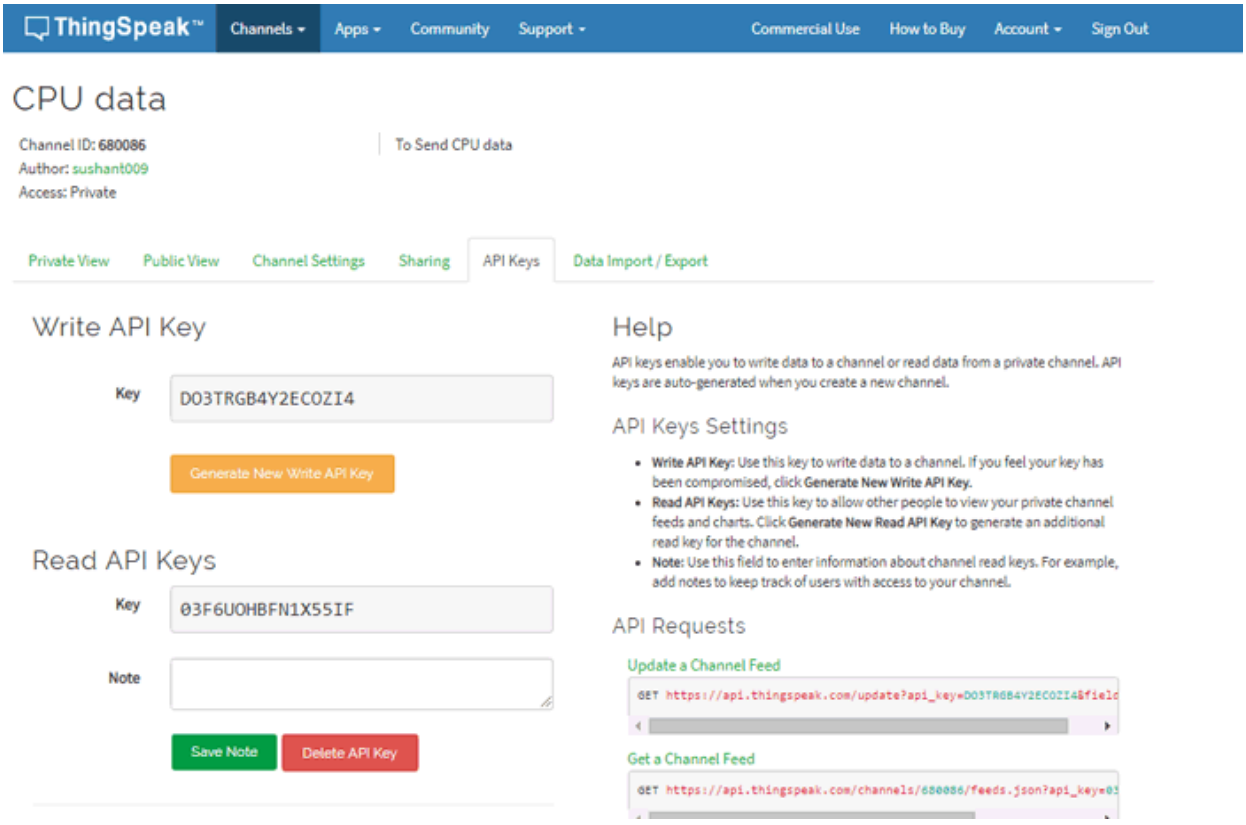


Fig7. Copy the Write API Key of the channel

(Note: The advantage of using Thingspeak compared to Xively or any other websites is that the convenience of using Matlab Analysis and Matlab Visualizations. This is a good option especially if you are doing some kind of research projects)

Once you have the "Write API Key". We are almost ready to upload our data, except for the python code.

Step 4: Modifying the Python Code

Go to https://github.com/sriharshakunda/Thingspeak_CPU_Python-Code

Download the code into your Raspberry Pi Home folder.

Open the CPU_Python.py file in a notepad.

Edit the line 19 by using CPU_Temp instead of temp.

Use your Write API Key to replace the key with your API Key

Save the file to overwrite changes

Step 5: Assuming you have python 2.7 and proper python libraries, go to the folder where you copied the CPU_Python.py file

Type python2.7 CPU_Python.py file

If the code runs properly you should see "200 ok" and something like "58.30" (CPU temperature value)

In case if there are any errors uploading the data, you will receive "connection failed" message

Step 6: Check Thinspeak API and Confirm data transfer

Open your channel and you should see the temperature uploading into thinspeak website.

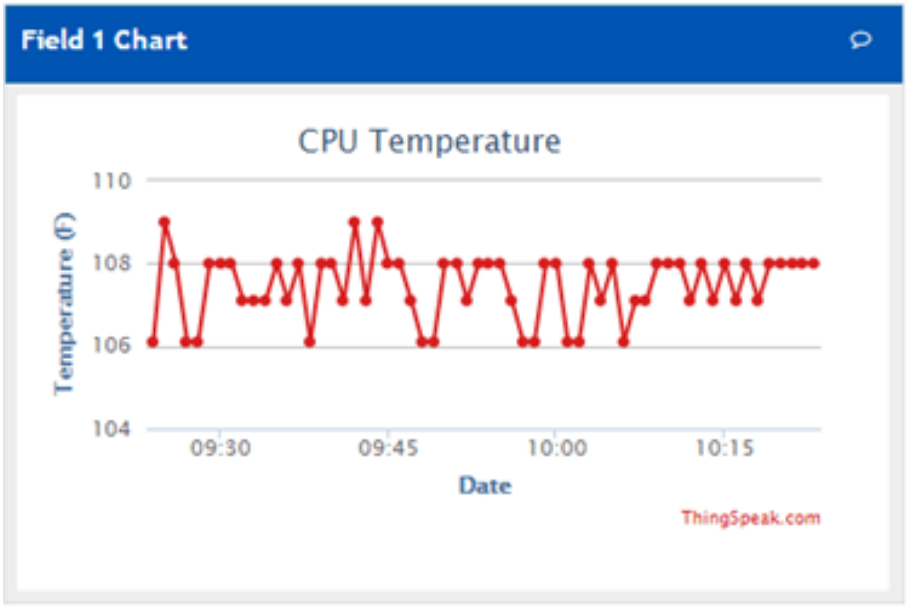
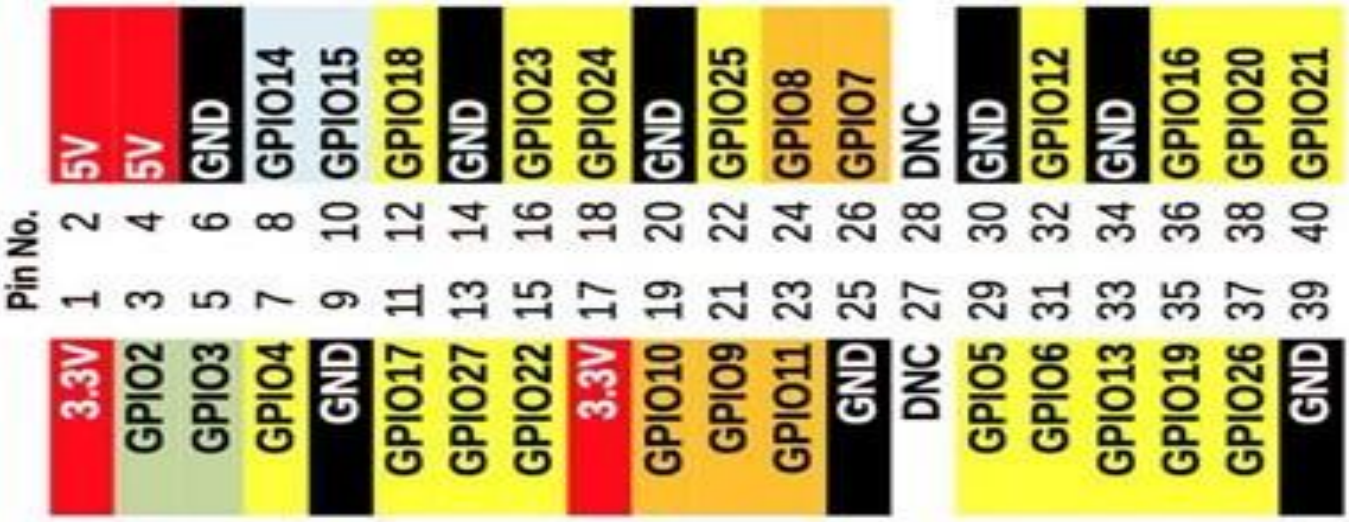


Fig9. CPU Temperature data displayed in Field Chart

Raspberry Pi GPIO Pin numbers



Unit V: Big Data - Data Storage and Analytics

Contents:

Types of data, What is Big Data (BD), Characteristics of Big data, Main components of Big Data Solution, Basic Architecture of BD Solution, Introduction to Hadoop, Introduction to data Analytics, Types of Data analytics, Statistical Models, Analysis of Variance, Data Dispersion, Contingence and Correlation, Regression Analysis, Precision and Error limits.

What is Data?

The quantities, characters, or symbols on which operations are performed by a computer, which may be stored and transmitted in the form of electrical signals and recorded on magnetic, optical, or mechanical recording media.

What is Big Data (BD)?

Big Data are data sets so large or so complex that the traditional data management tools are not able to store it or process it efficiently. However, there is a lot of potential value hidden in this data, so organizations are eager to harness it for a competitive advantage.

Big Data technologies and approaches are used to drive value out of such data in ways that traditional analytics tools and methods cannot.

Big data technologies enable organizations to gather, store, manage, and manipulate vast amounts and variety data at the right speed, at the right time, to gain the right information out of it.

Big Data is a phrase used to mean a massive volume of both structured and unstructured data that is so large it is difficult to process using traditional database and software techniques. In most enterprise scenarios the volume of data is too big or it moves too fast or it exceeds current processing capacity.

Big Data is also **data** but with a **huge size**. Big Data is a term used to describe a collection of data that is huge in size and yet growing exponentially with time. In short such data is so large and complex that none of the traditional data management tools are able to store it or process it efficiently.

Importance of big data

Companies use the big data accumulated in their systems to improve operations, provide better customer service, create personalized marketing campaigns based on specific customer preferences and, ultimately, increase profitability. Businesses that utilize big data hold a potential competitive advantage over those that don't since they're able to make faster and more informed business decisions, provided they use the data effectively.

For example, big data can provide companies with valuable insights into their customers that can be used to refine marketing campaigns and techniques in order to increase customer engagement and conversion rates.

Furthermore, utilizing big data enables companies to become increasingly customer-centric. Historical and real-time data can be used to assess the evolving preferences of consumers, consequently enabling businesses to update and improve their marketing strategies and become more responsive to customer desires and needs.

Big data is also used by medical researchers to identify disease risk factors and by doctors to help diagnose illnesses and conditions in individual patients. In addition, data derived from electronic health records (EHRs), social media, the web and other sources provides healthcare organizations and government agencies with up-to-the-minute information on infectious disease threats or outbreaks.

In the energy industry, big data helps oil and gas companies identify potential drilling locations and monitor pipeline operations; likewise, utilities use it to track electrical grids. Financial services firms use big data systems for risk management and real-time analysis of market data. Manufacturers and transportation companies rely on big data to manage their supply chains and optimize delivery routes. Other government uses include emergency response, crime prevention and smart city initiatives.

Big Data Characteristics

We differentiate Big Data characteristics from traditional data by one or more of the four V's: Volume, Velocity, Variety and variability.

1. Volume:

Volume is the amount of data generated that must be understood to make data-based decisions.

A text file is a few kilobytes, a sound file is a few megabytes while a full-length movie is a few gigabytes.

Example:

Amazon handles 15 million customer click stream user data per day to recommend products.

Extremely **large volume of data** is a major characteristic of **big data online training**

2). Velocity:

Velocity measures how fast data is produced and modified and the speed with which it needs to be processed. An increased number of data sources both machine and human generated drive velocity.

Example:

72 hours of video are uploaded to **YouTube** every minute this is the velocity. Extremely high velocity of data is another major big data characteristics

3) Variety:

Variety defines data coming from new source both inside and outside of an enterprise It can be structured, semi-structured or unstructured.

Structured data:

It is typically found in tables with columns and rows of data. The intersection of the row and the column in a cell has a value and is given a ‒key,‖ which it can be referred to in queries. Because there is a direct relationship between the column and the row, these databases are commonly referred to as relational databases. A retail outlet that stores their sales data (name of person, product sold, amount) in an Excel spreadsheet or CSV file is an example of structured data.

Example:

A Product table in a database is an example of Structured Data

Product_id	Product_name	Product_price
1	Pen	$5.95
2	Paper	$8.95

Semi-structured data also has an organization, but the table structure is removed so the data can be more easily read and manipulated. XML files or an RSS feed for a webpage are examples of semi-structured data.

Unstructured data:

Unstructured data generally has no organizing structure, and Big Data technologies use different ways to add structure to this data. Typical example of unstructured data is, a heterogeneous data source containing a combination of simple text files, images, videos etc

4) Variability

This refers to the inconsistency which can be shown by the data at times, thus hampering the process of being able to handle and manage the data effectively.

You can see that few values are missing in the below table

Department	Year	Minimum sales	Maximum sales
1	2010	?	1500
2	2011	10000	?

Data available can sometimes get messy and maybe difficult to trust. With wide variety in big data types generated, quality and accuracy are difficult to control.

Other important characteristics of Big Data are:

- **Exhaustive**
  Whether the entire system is captured or recorded or not.
- **Fine-grained and uniquely lexical**
  Respectively, the proportion of specific data of each element per element collected and if the element and its characteristics are properly indexed or identified.
- **Relational**
  If the data collected contains commons fields that would enable a conjoining, or meta-analysis, of different data sets.
- **Extensional**
  If new fields in each element of the data collected can be added or changed easily.
- **Scalability**
  If the size of the data can expand rapidly.
- **Value**
  The utility that can be extracted from the data.
- **Variability**
  It refers to data whose value or other characteristics are shifting in relation to the context they are being generated.

Advantages of Big Data (Features)

- One of the biggest advantages of Big Data is predictive analysis. Big Data analytics tools can predict outcomes accurately, thereby, allowing businesses and organizations to make better decisions, while simultaneously optimizing their operational efficiencies and reducing risks.
- By harnessing data from social media platforms using Big Data analytics tools, businesses around the world are streamlining their digital marketing strategies to enhance the overall consumer experience. Big Data provides insights into the customer pain points and allows companies to improve upon their products and services.
- Being accurate, Big Data combines relevant data from multiple sources to produce highly actionable insights. Almost 43% of companies lack the necessary tools to filter out irrelevant data, which eventually costs them millions of dollars to hash out useful data from the bulk. Big Data tools can help reduce this, saving you both time and money.
- Big Data analytics could help companies generate more sales leads which would naturally mean a boost in revenue. Businesses are using Big Data analytics tools to understand how well their products/services are doing in the market and how the customers are responding to them. Thus, the can understand better where to invest their time and money.
- With Big Data insights, you can always stay a step ahead of your competitors. You can screen the market to know what kind of promotions and offers your rivals are providing, and then you can come up with better offers for your customers. Also, Big Data insights allow you to learn customer behavior to understand the customer trends and provide a highly _personalized‘ experience to them.

Data types in Big Data

1. Structured data
2. Unstructured data
3. Semi-structured data

1. Structured data

Structured Data is used to refer to the data which is already stored in databases, in an ordered manner. It accounts for about 20% of the total existing data and is used the most in programming and computer-related activities.

There are two sources of structured data- machines and humans. All the data received from sensors, weblogs, and financial systems are classified under machine-generated data. These include medical devices, GPS data, data of usage statistics captured by servers and applications and the huge amount of data that usually move through trading platforms, to name a few.

Human-generated structured data mainly includes all the data a human input into a computer, such as his name and other personal details. When a person clicks a link on the internet, or even makes a move in a game, data is created- this can be used by companies to figure out their customer behavior and make the appropriate decisions and modifications.

Let‘s understand structured data with an example.

Top 3 players who have scored most runs in international T20 matches are as follows:

Player	**Country**	**Scores**	**No of Matches Played**
Brendon McCullum	New Zealand	2140	71
Rohit Sharma	India	2237	90
Virat Kohli	India	2167	65

2. Unstructured data

While structured data resides in the traditional row-column databases, unstructured data is the opposite- they have no clear format in storage. The rest of the data created, about 80% of the total account for unstructured big data. Most of the data a person encounters belong to this category- and until recently, there was not much to do to it except storing it or analyzing it manually.

Unstructured data is also classified based on its source, into machine-generated or human-generated. Machine-generated data accounts for all the satellite images, the scientific data from various experiments and radar data captured by various facets of technology.

Human-generated unstructured data is found in abundance across the internet since it includes social media data, mobile data, and website content. This means that the pictures we upload to Facebook or Instagram handle, the videos we watch on YouTube and even the text messages we send all contribute to the gigantic heap that is unstructured data.
Examples of unstructured data include text, video, audio, mobile activity, social media activity, satellite imagery, surveillance imagery – the list goes on and on.

The Unstructured data is further divided into –

- Captured
- User-Generated data

a. Captured data:

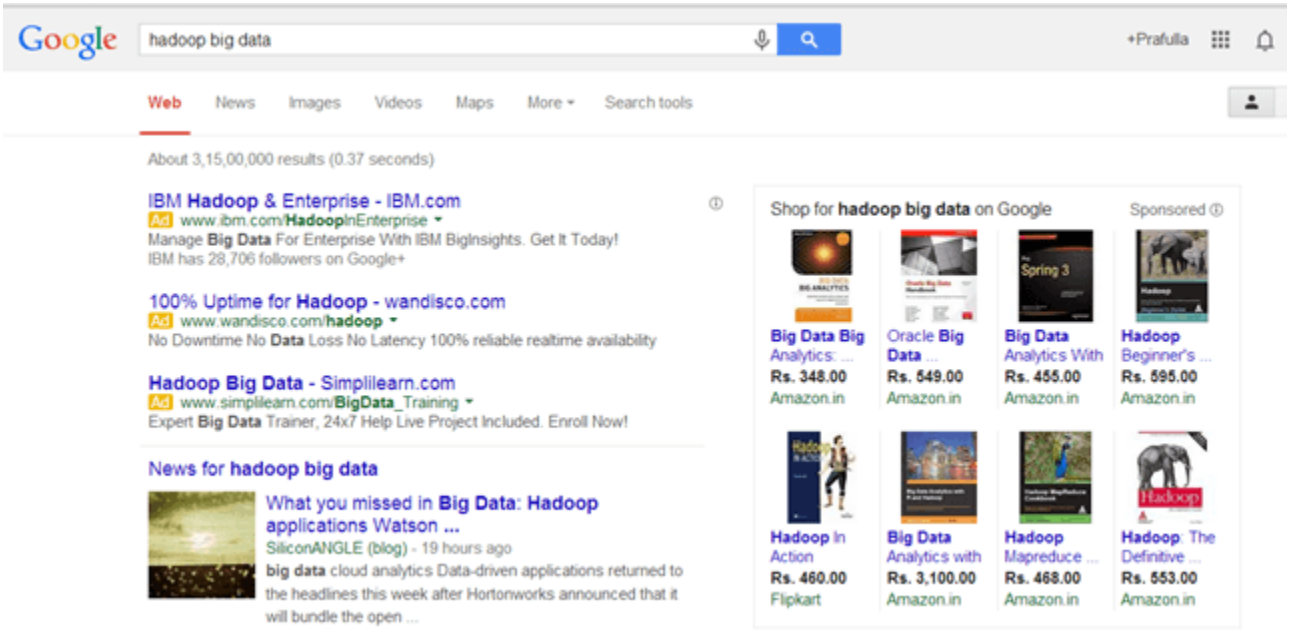
It is the data based on the user‘s behavior. The best example to understand it is GPS via smartphones which help the user each and every moment and provides a real-time output.

b. User-generated data:

It is the kind of unstructured data where the user itself will put data on the internet every movement. For example, Tweets and Re-tweets, Likes, Shares, Comments, on Youtube, Facebook, etc.

Examples of Un-structured Data

The output returned by 'Google Search'



3. Semi-structured data:

The line between unstructured data and semi-structured data has always been unclear since most of the semi-structured data appear to be unstructured at a glance. Information that is not in the traditional database format as structured data, but contains some organizational properties which make it easier to process, are included in semi-structured data. For example, NoSQL documents are considered to be semi-structured, since they contain keywords that can be used to process the document easily.

Big Data analysis has been found to have definite business value, as its analysis and processing can help a company achieve cost reductions and dramatic growth. So it is imperative that you do not wait too long to exploit the potential of this excellent business opportunity.

Examples of Semi-structured Data

Personal data stored in an XML file-

```
<rec><name>Prashant Rao</name><gender>Male><age>35</age></rec>
<rec><name>Seema R.</name>< gender >Female><age>41</age></rec>
<rec><name>Satish Mane</name>< gender >Male><age>29</age></rec>
<rec><name>Subrato Roy</name>< gender >Male><age>26</age></rec>
<rec><name>Jeremiah J.</name>< gender >Male><age>35</age></rec>
```

Difference between Structured, Semi-structured and Unstructured data

Factors	Structured data	Semi-structured data	Unstructured data
Flexibility	It is dependent and less flexible	It is more flexible than structured data but less than flexible than unstructured data	It is flexible in nature and there is an absence of a schema
Transaction Management	Matured transaction and various concurrency technique	The transaction is adapted from DBMS not matured	No transaction management and no concurrency
Query performance	Structured query allow complex joining	Queries over anonymous nodes are possible	An only textual query is possible
Technology	It is based on the relational database table	It is based on RDF and XML	This is based on character and library data

Main components of Big Data Solution

- ***Machine Learning***

It is the science of making computers learn stuff by themselves. In machine learning, a computer is expected to use algorithms and statistical models to perform specific tasks without any explicit instructions. Machine learning applications provide results based on past experience. For example, these days there are some mobile applications that will give you a summary of your finances, bills, will remind you on your bill payments, and also may give you suggestions to go for some saving plans. These functions are done by reading your emails and text messages.

- ***Natural Language Processing (NLP)***

It is the ability of a computer to understand human language as spoken. The most obvious examples that people can relate to these days is google home and Amazon Alexa. Both use NLP and other technologies to give us a virtual assistant experience. NLP is all around us without us even realizing it. When writing a mail, while making any mistakes, it automatically corrects itself and these days it gives auto-suggests for completing the mails and automatically intimidates us when we try to send an email without the attachment that we referenced in the text of the email, this is part of Natural Language Processing Applications which are running at the backend.

- ***Business Intelligence***

Business Intelligence (BI) is a method or process that is technology driven to gain insights by analyzing data and presenting it in a way that the end users (usually high-level executives) like managers and corporate leaders can gain some actionable insights from it and make informed business decisions on it.

- ***Cloud Computing***

If we go by the name, it should be computing done on clouds, well, it is true, just here we are not talking about real clouds, cloud here is a reference for the Internet. So we can define cloud computing as the delivery of computing services—servers, storage, databases, networking, software, analytics, intelligence and more—over the Internet (‑the cloud‖) to offer faster innovation, flexible resources, and economies of scale.

Basic Architecture of Big Data

Introduction to Big Data Architecture

When it comes to managing heavy data and doing complex operations on that massive data there becomes a need to use big data tools and techniques. When we say using big data tools and techniques we effectively mean that we are asking to make use of various software and procedures which lie in the big data ecosystem and its sphere. There is no generic solution that is provided for every use case and therefore it has to be crafted and made in an effective way as per the business requirements of a particular company. Thus there becomes a need to make use of different big data architecture as the combination of various technologies will result in the resultant use case being achieved. By establishing a fixed architecture it can be ensured that a viable solution will be provided for the asked use case.

What is Big Data Architecture?

- This architecture is designed in such a way that it handles the ingestion process, processing of

data and analysis of the data is done which is way too large or complex to handle the traditional database management systems.

- Different organizations have different thresholds for their organizations, some have it for a few hundred gigabytes while for others even some terabytes are not good enough a threshold value.
- Due to this event happening if you look at the commodity systems and the commodity storage the values and the cost of storage have reduced significantly. There is a huge variety of data that demands different ways to be catered.
- Some of them are batch related data that comes at a particular time and therefore the jobs are required to be scheduled in a similar fashion while some others belong to the streaming class where a real-time streaming pipeline has to be built to cater to all the requirements. All these challenges are solved by big data architecture.

Need of Big Data Architecture

1. You want to extract information from extensive networking or web logs.
2. You process massive datasets over 100GB in size. Some of these computing tasks run 8 hours or longer.
3. You are willing to invest in a big data project, including third-party products to optimize your environment.
4. You store large amounts of unstructured data that you need to summarize or transform into a structured format for better analytics.
5. You have multiple large data sources to analyze, including structured and unstructured.
6. You want to proactively analyze big data for business needs, such as analyzing store sales by season and advertising, applying sentiment analysis to social media posts, or investigating email for suspicious communication patterns – or all the above.

A big data architecture is designed to handle the assimilation, processing, and analysis of data that is too large or complex for traditional database systems.

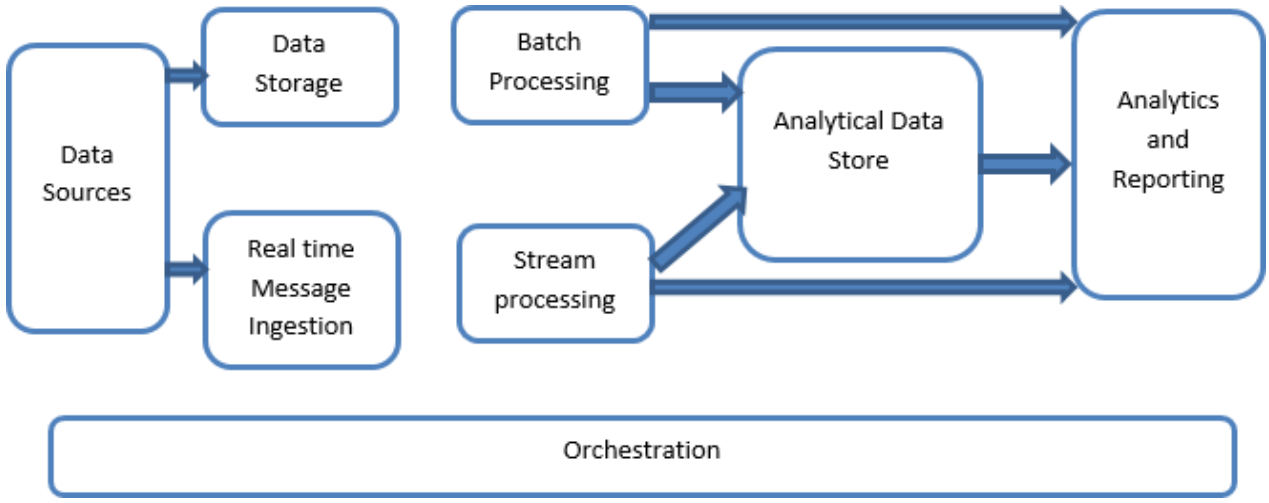


Figure: Big Data Architecture

Big data solutions typically involve one or more of the following types of workload:

- Batch processing of big data sources at rest.
- Real-time processing of big data in motion.
- Interactive exploration of big data.
- Predictive analytics and machine learning.

Most big data architectures include some or all of the following components:

- **Data sources**: All big data solutions start with one or more data sources. Examples include:
  - Application data stores, such as relational databases.
  - Static files produced by applications, such as web server log files.
  - Real-time data sources, such as IoT devices.
- **Data storage**: Data for batch processing operations is typically stored in a distributed file store that can hold high volumes of large files in various formats. This kind of store is often called a *data lake*. This generally forms the part where our Hadoop storage such as HDFS, Microsoft Azure, AWS, GCP storages are provided along with blob containers.
- **Batch processing**: Since the data is so huge in size, the architecture needs a batch processing, system to filter, aggregate, and process data for advanced analytics. Usually these jobs involve reading source files, processing them, and writing the output to new files. This requires batch processing. The batch processing is done in various ways by making use of Hive jobs or U-SQL based jobs or by making use of Sqoop or Pig along with the custom map reducer jobs which are generally written in any one of the Java or Scala or any other language such as Python.
- **Real-time message ingestion**: If the solution includes real-time sources, the architecture must include a way to capture and store real-time messages for stream processing. This might be a simple data store, where incoming messages are dropped into a folder for processing. However, many solutions need a message ingestion store to act as a buffer for messages, and to support scale-out processing, reliable delivery, and other message queuing semantics. Options include Azure Event Hubs, Azure IoT Hubs, and Kafka.
- **Stream processing**: After capturing real-time messages, the solution must process them by filtering, aggregating, and otherwise preparing the data for analysis. The processed stream data is then written to an output sink. Stream processing, on the other hand, is used to handle all that streaming data which is occurring in windows or streams and then writes the data to the output sink. This includes Apache Spark, Apache Flink, Storm, etc.
- **Analytical data store**: Many big data solutions prepare data for analysis and then serve the processed data in a structured format that can be queried using analytical tools. The analytical data store used to serve these queries can be a Kimball-style relational data warehouse, as seen in most traditional business intelligence (BI) solutions. Alternatively, the data could be presented through a low-latency NoSQL technology such as HBase, or an interactive Hive database that provides a metadata abstraction over data files in the distributed data store. Azure Synapse Analytics provides a managed service for large-scale, cloud-based data warehousing. HDInsight supports Interactive Hive, HBase, and Spark SQL, which can also be used to serve data for analysis.
- **Analysis and reporting**: The insights have to be generated on the processed data and that is effectively done by the reporting and analysis tools which makes use of their embedded technology and solution to generate useful graphs, analysis, and insights helpful to the businesses. Tools include Cognos, Hyperion, etc.
- **Orchestration**: Most big data solutions consist of repeated data processing operations, encapsulated in work flows, that transform source data, move data between multiple sources and sinks, load the processed data into an analytical data store, or push the results straight to a report or dashboard. To automate these workflows, you can use an orchestration technology such Azure Data Factory or Apache Oozie and Sqoop.

Benefits of Big Data

- **Better decision-making:** Use of machine learning in big data analysis gives better decision-making
- **Increased productivity:** Modern big data tools are allowing analysts to analyze more data, more quickly, which increases their personal productivity. In addition, the insights gained from those analytics often allow organizations to increase productivity more broadly throughout the

company.

- **Improved business processes:** Probably the biggest advantage of big data is it helps businesses to gain a huge competitive advantage. Apart from being able to understand, as well as, target customers better, analyzing big data can result in the improvement and optimization of certain facets of business operations. For instance, by mining big data retailers can not only explore patterns in consumption and production but can also promote better inventory management, improve the supply chain, optimize distribution channels, among others.
- **Improved customer service:** One of the most common goals among big data analytics programs is improving customer service. Today‘s businesses capture a huge amount of information from different sources like customer relationship management (CRM) systems, social media together with other points of customer contact. By analyzing this massive amount of information they get to know about the tastes and preferences of a user. And with the help of the big data technologies, they become able to create experiences which are more responsive, personal, and accurate than ever before.
- **Fraud detection:** This advantage of using big data comes from the implementation of machine learning technologies. It helps banks and other financial institutions to detect frauds like fraudulent purchases with credit cards often before even the cardholder gets to know about it.
- **Increased revenue:** When organizations use big data to improve their decision-making and improve their customer service, increased revenue is often the natural result.
- **Increased agility:** Many organizations are using their big data to better align their IT and business efforts, and they are using their analytics to support faster and more frequent changes to their business strategies and tactics.
- **Predictive analysis:** Predictive analysis will keep you ahead of your competitors. Big data can facilitate this by, as an example, scanning and analyzing social media feeds and newspaper reports. Big data also helps you do health-tests on your customers, suppliers, and other stakeholders to help you reduce risks such as default.
- **Greater innovation:** Big data analysis derives innovative solutions
- **Big Data is Reliable:** Things as simple as securing the correct contact data of customers through the review of multiple systems can save thousands of Dollars in incorrectly sent communications.

Drawbacks or disadvantages of Big Data

Following are the drawbacks or disadvantages of Big Data:

- Traditional storage can cost lot of money to store big data.
- Lots of big data is unstructured.
- Big data analysis violates principles of privacy.
- It can be used for manipulation of customer records.
- It may increase social stratification.
- Big data analysis is not useful in short run. It needs to be analyzed for longer duration to leverage its benefits.
- Big data analysis results are misleading sometimes.
- Speedy updates in big data can mismatch real figures.

Challenges of Big Data

- **Complexity**. Big data solutions can be extremely complex, with numerous components to handle data ingestion from multiple data sources. It can be challenging to build, test, and troubleshoot big data processes. Moreover, there may be a large number of configuration settings across multiple systems that must be used in order to optimize performance.
- **Skillset**. Many big data technologies are highly specialized, and use frameworks and languages that are not typical of more general application architectures. On the other hand, big data

technologies are evolving new APIs that build on more established languages. For example, the U-SQL language in Azure Data Lake Analytics is based on a combination of Transact-SQL and C#. Similarly, SQL-based APIs are available for Hive, HBase, and Spark.

- **Technology maturity**. Many of the technologies used in big data are evolving. While core Hadoop technologies such as Hive and Pig have stabilized, emerging technologies such as Spark introduce extensive changes and enhancements with each new release. Managed services such as Azure Data Lake Analytics and Azure Data Factory are relatively young, compared with other Azure services, and will likely evolve over time.
- **Security**. Big data solutions usually rely on storing all static data in a centralized data lake. Securing access to this data can be challenging, especially when the data must be ingested and consumed by multiple applications and platforms.
- **Need For Synchronization Across Disparate Data Sources**
  As data sets are becoming bigger and more diverse, there is a big challenge to incorporate them into an analytical platform. If this is overlooked, it will create gaps and lead to wrong messages and insights.
- **Acute Shortage Of Professionals Who Understand Big Data Analysis**
  The analysis of data is important to make this voluminous amount of data being produced in every minute, useful. With the exponential rise of data, a huge demand for big data scientists and Big Data analysts has been created in the market. It is important for business organizations to hire a data scientist having skills that are varied as the job of a data scientist is multidisciplinary. Another major challenge faced by businesses is the shortage of professionals who understand Big Data analysis. There is a sharp shortage of data scientists in comparison to the massive amount of data being produced.
- **Getting Meaningful Insights Through The Use Of Big Data Analytics**
  It is imperative for business organizations to gain important insights from Big Data analytics, and also it is important that only the relevant department has access to this information. A big challenge faced by the companies in the Big Data analytics is mending this wide gap in an effective manner.
- **Getting Voluminous Data Into The Big Data Platform**
  It is hardly surprising that data is growing with every passing day. This simply indicates that business organizations need to handle a large amount of data on daily basis. The amount and variety of data available these days can overwhelm any data engineer and that is why it is considered vital to make data accessibility easy and convenient for brand owners and managers.
- **Uncertainty Of Data Management Landscape**
  With the rise of Big Data, new technologies and companies are being developed every day. However, a big challenge faced by the companies in the Big Data analytics is to find out which technology will be best suited to them without the introduction of new problems and potential risks.
- **Data Storage And Quality**
  Business organizations are growing at a rapid pace. With the tremendous growth of the companies and large business organizations, increases the amount of data produced. The storage of this massive amount of data is becoming a real challenge for everyone. Popular data storage options like data lakes/ warehouses are commonly used to gather and store large quantities of unstructured and structured data in its native format. The real problem arises when a data lakes/ warehouse try to combine unstructured and inconsistent data from diverse sources, it encounters errors. Missing data, inconsistent data, logic conflicts, and duplicates data all result in data quality challenges.
- **Security And Privacy Of Data**
  Once business enterprises discover how to use Big Data, it brings them a wide range of possibilities and opportunities. However, it also involves the potential risks associated with big

data when it comes to the privacy and the security of the data. The Big Data tools used for analysis and storage utilizes the data disparate sources. This eventually leads to a high risk of exposure of the data, making it vulnerable. Thus, the rise of voluminous amount of data increases privacy and security concerns.

- **Dealing with data growth**
  As day by day data storage is increasing so dealing with growth of data is a big challenge
- **Recruiting and retaining big data talent**
- **Generating insights in a timely manner**
- **Securing big data**

Best practices

- **Leverage parallelism**. Most big data processing technologies distribute the workload across multiple processing units. This requires that static data files are created and stored in a splittable format. Distributed file systems such as HDFS can optimize read and write performance, and the actual processing is performed by multiple cluster nodes in parallel, which reduces overall job times.
- **Partition data**. Batch processing usually happens on a recurring schedule — for example, weekly or monthly. Partition data files, and data structures such as tables, based on temporal periods that match the processing schedule. That simplifies data ingestion and job scheduling, and makes it easier to troubleshoot failures. Also, partitioning tables that are used in Hive, U-SQL, or SQL queries can significantly improve query performance.
- **Apply schema-on-read semantics**. Using a data lake lets you to combine storage for files in multiple formats, whether structured, semi-structured, or unstructured. Use *schema-on-read* semantics, which project a schema onto the data when the data is processing, not when the data is stored. This builds flexibility into the solution, and prevents bottlenecks during data ingestion caused by data validation and type checking.
- **Process data in-place**. Traditional BI solutions often use an extract, transform, and load (ETL) process to move data into a data warehouse. With larger volumes data, and a greater variety of formats, big data solutions generally use variations of ETL, such as transform, extract, and load (TEL). With this approach, the data is processed within the distributed data store, transforming it to the required structure, before moving the transformed data into an analytical data store.
- **Balance utilization and time costs**. For batch processing jobs, it's important to consider two factors: The per-unit cost of the compute nodes, and the per-minute cost of using those nodes to complete the job. For example, a batch job may take eight hours with four cluster nodes. However, it might turn out that the job uses all four nodes only during the first two hours, and after that, only two nodes are required. In that case, running the entire job on two nodes would increase the total job time, but would not double it, so the total cost would be less. In some business scenarios, a longer processing time may be preferable to the higher cost of using underutilized cluster resources.
- **Separate cluster resources**. When deploying HDInsight clusters, you will normally achieve better performance by provisioning separate cluster resources for each type of workload. For example, although Spark clusters include Hive, if you need to perform extensive processing with both Hive and Spark, you should consider deploying separate dedicated Spark and Hadoop clusters. Similarly, if you are using HBase and Storm for low latency stream processing and Hive for batch processing, consider separate clusters for Storm, HBase, and Hadoop.
- **Orchestrate data ingestion**. In some cases, existing business applications may write data files for batch processing directly into Azure storage blob containers, where they can be consumed by HDInsight or Azure Data Lake Analytics. However, you will often need to orchestrate the ingestion of data from on-premises or external data sources into the data lake. Use an orchestration workflow or pipeline, such as those supported by Azure Data Factory or Oozie, to

achieve this in a predictable and centrally manageable fashion.

- **Scrub sensitive data early**. The data ingestion workflow should scrub sensitive data early in the process, to avoid storing it in the data lake.

Who is using Big Data?

- Banking
- Government
- Education
- Health care
- E-commerce
- Social Media

Introduction to Hadoop:

What is Hadoop?

Hadoop is an open source software programming framework for storing a large amount of data and performing the computation. Its framework is based on Java programming with some native code in C and shell scripts.

Hadoop is an Apache open source framework written in java that allows distributed processing of large datasets across clusters of computers using simple programming models. The Hadoop framework application works in an environment that provides distributed *storage* and *computation* across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

Hadoop is an open-source, Java-based framework from Apache which is used for storing, processing and analyzing data which are very huge in volume.

Hadoop is used for **batch/ offline processing**. It is a collection of software utilities which uses a network of many computers to **solve problems involving large amounts of data and computation**.

How Does Hadoop Work?

It is quite expensive to build bigger servers with heavy configurations that handle large scale processing, but as an alternative, you can tie together many commodity computers with single-CPU, as a single functional distributed system and practically, the clustered machines can read the dataset in parallel and provide a much higher throughput. Moreover, it is cheaper than one high-end server. So this is the first motivational factor behind using Hadoop that it runs across clustered and low-cost machines.

Hadoop runs code across a cluster of computers. This process includes the following core tasks that Hadoop performs −

- Data is initially divided into directories and files. Files are divided into uniform sized blocks of 128M and 64M (preferably 128M).

- These files are then distributed across various cluster nodes for further processing.
- HDFS, being on top of the local file system, supervises the processing.
- Blocks are replicated for handling hardware failure.
- Checking that the code was executed successfully.
- Performing the sort that takes place between the map and reduce stages.
- Sending the sorted data to a certain computer.
- Writing the debugging logs for each job.

Architecture of Hadoop:

The architecture of Hadoop involves a package of the file system and operating system abstractions which is called the **Hadoop Common package,** containing a **MapReduce engine** (processing part) and the **Hadoop Distributed File System** (storage part).
The Hadoop Common package contains the JAR (Java ARchive) files and scripts needed to start Hadoop.

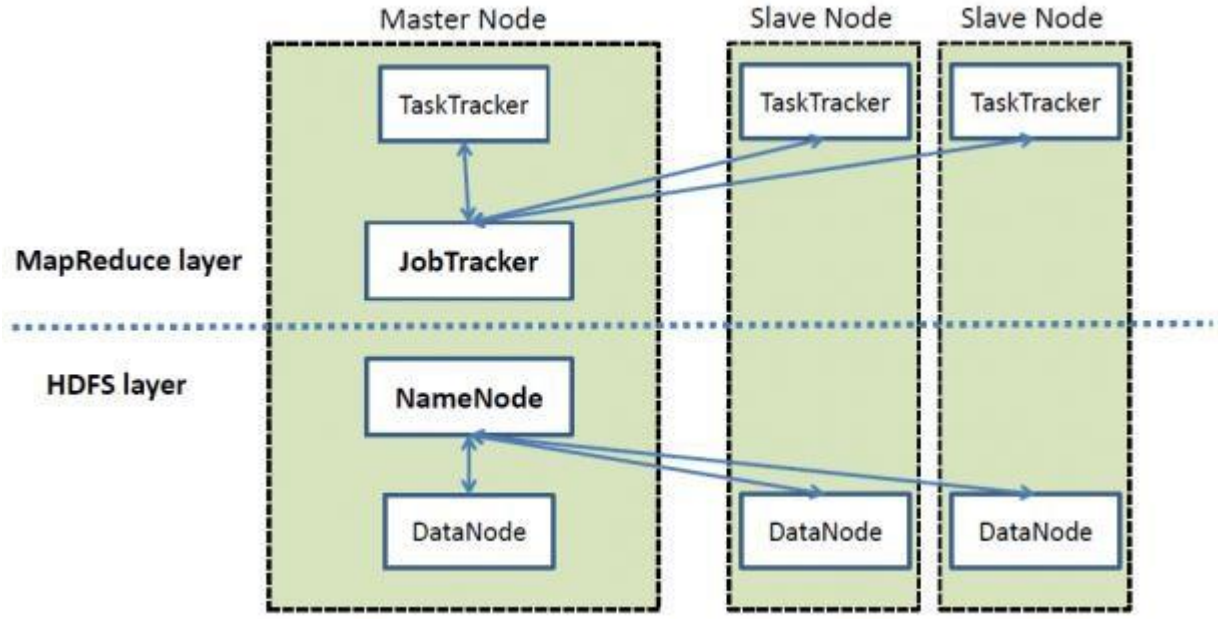


Fig.: High Level Architecture of Hadoop

Every Hadoop cluster consists of a single master and multiple worker nodes. The ***Master node*** has a Job Tracker, Task Tracker, Name Node and Data Node while the ***Slave (worker node)*** can act as

both a DataNode and TaskTracker. Also it is possible to have data-only and compute only worker nodes.
The standard startup and shutdown scripts require that *Secure Shell (SSH)* to be set up between nodes in the cluster. Apart from this Hadoop requires a *JRE (Java Runtime Environment)* of 1.6 or greater.

Hadoop has a Master-Slave Architecture for data storage and distributed data processing using MapReduce and HDFS methods.

NameNode:

NameNode represented every files and directory which is used in the namespace

DataNode:

DataNode helps you to manage the state of an HDFS node and allows you to interacts with the blocks

MasterNode:

The master node allows you to conduct parallel processing of data using Hadoop MapReduce.

Slave node:

The slave nodes are the additional machines in the Hadoop cluster which allows you to store data to conduct complex calculations. Moreover, all the slave node comes with Task Tracker and a DataNode. This allows you to synchronize the processes with the NameNode and Job Tracker respectively.

Core elements of Hadoop

There are four basic elements to Hadoop: HDFS; MapReduce; YARN; Common.

HDFS(Hadoop Distributed File System)

Hadoop works across clusters of commodity servers. Therefore there needs to be a way to coordinate activity across the hardware. Hadoop can work with any distributed file system, however the Hadoop Distributed File System is the primary means for doing so and is the heart of Hadoop technology. HDFS manages how data files are divided and stored across the cluster. Data is divided into blocks, and each server in the cluster contains data from different blocks. There is also some built-in redundancy.

YARN

YARN or Yet Another Resource Negotiator is the resource management layer of Hadoop. The basic principle behind YARN is to separate resource management and job scheduling/monitoring function into separate daemons. In YARN there is one global Resource Manager and per-application Application Master. An Application can be a single job or a DAG of jobs.
Inside the YARN framework, we have two daemons Resource Manager and Node Manager. The Resource Manager arbitrates resources among all the competing applications in the system. The job of Node Manager is to monitor the resource usage by the container and report the same to Resource Manager. The resources are like CPU, memory, disk, network and so on.

The ResourceManger has two important components – Scheduler and ApplicationManager

i. Scheduler

Scheduler is responsible for allocating resources to various applications. This is a pure scheduler as it does not perform tracking of status for the application. It also does not reschedule the tasks which fail due to software or hardware errors. The scheduler allocates the resources based on the requirements of the applications.

ii. Application Manager

Following are the functions of ApplicationManager

- Accepts job submission.
- Negotiates the first container for executing ApplicationMaster. A container incorporates elements such as CPU, memory, disk, and network.
- Restarts the ApplicationMaster container on failure.

Functions of ApplicationMaster:-

- Negotiates resource container from Scheduler.
- Tracks the resource container status.
- Monitors progress of the application.

We can scale the YARN beyond a few thousand nodes through YARN Federation feature. This feature enables us to tie multiple **YARN clusters** into a single massive cluster. This allows for using independent clusters, clubbed together for a very large job.

MapReduce:

MapReduce is the data processing layer of Hadoop. It is a software framework that allows you to write applications for processing a large amount of data. MapReduce runs these applications in parallel on a cluster of low-end machines. It does so in a reliable and fault-tolerant manner.

MapReduce job comprises a number of map tasks and reduces tasks. Each task works on a part of data. This distributes the load across the cluster. The function of Map tasks is to load, parse, transform and filter data. Each reduce task works on the sub-set of output from the map tasks. Reduce task applies grouping and aggregation to this intermediate data from the map tasks.

The input file for the MapReduce job exists on HDFS. The input format decides how to split the input file into input splits. Input split is nothing but a byte-oriented view of the chunk of the input file. This input split gets loaded by the map task. The map task runs on the node where the relevant data is present. The data need not move over the network and get processed locally.

JobTracker is a component of the MapReduce engine that manages how client applications submit MapReduce jobs. It distributes work to **TaskTracker** nodes. TaskTracker attempts to assign processing as close to where the data resides as possible.

Note that MapReduce is not the only way to manage parallel processing in the Hadoop environment.

What is Big Data and Why Use Hadoop?

Big data specifies datasets that are very big. It‘s a hoard of large datasets that cannot be processed by the traditional methods of computing. Big data is related to a complete subject rather than merely data that can be processed using various techniques, tools, and framework. Hadoop is an open-source frame, which is based on Java Programming and supports the storage and processing capabilities of extremely large datasets in a computing environment that is distributed across branches. Hadoop was developed by a team of computer scientists, which comprised of Mike

Cafarella and Doug Cutting in 2005, to support the distribution capabilities of search engines. There are pros & cons in hadoop, but compared to pros, cons are negotiable.

Key features that answer – Why Hadoop?

1. Flexible:

As it is a known fact that only 20% of data in organizations is structured, and the rest is all unstructured, it is very crucial to manage unstructured data which goes unattended. Hadoop manages different types of Big Data, whether structured or unstructured, encoded or formatted, or any other type of data and makes it useful for decision making process. Moreover, Hadoop is simple, relevant and schema-less! Though Hadoop generally supports **Java Programming**, any programming language can be used in Hadoop with the help of the **MapReduce technique**. Though Hadoop works best on **Windows** and **Linux**, it can also work on other operating systems like **BSD** and **OS X**.

2. Scalable

Hadoop is a scalable platform, in the sense that new nodes can be easily added in the system as and when required without altering the data formats, how data is loaded, how programs are written, or even without modifying the existing applications. Hadoop is an open source platform and runs on industry-standard hardware. Moreover, Hadoop is also fault tolerant – this means, even if a node gets lost or goes out of service, the system automatically reallocates work to another location of the data and continues processing as if nothing had happened!

3. Building more efficient data economy:

Hadoop has revolutionized the processing and analysis of big data world across. Till now, organizations were worrying about how to manage the non-stop data overflowing in their systems. Hadoop is more like a ‑Dam‖, which is harnessing the flow of unlimited amount of data and generating a lot of power in the form of relevant information. Hadoop has changed the economics of storing and evaluating data entirely!

4. Robust Ecosystem:

Hadoop has a very robust and a rich ecosystem that is well suited to meet the analytical needs of developers, web start-ups and other organizations. Hadoop Ecosystem consists of various related projects such as MapReduce, Hive, HBase, Zookeeper, HCatalog, Apache Pig, which make Hadoop very competent to deliver a broad spectrum of services.

5. Hadoop is getting more “Real-Time”!

Did you ever wonder how to stream information into a cluster and analyze it in real time? Hadoop has the answer for it. Yes, Hadoop‘s competencies are getting more and more real-time. Hadoop also provides a standard approach to a wide set of APIs for big data analytics comprising MapReduce, query languages and database access, and so on.

6. Cost Effective:

Loaded with such great features, the icing on the cake is that Hadoop generates cost benefits by bringing massively parallel computing to commodity servers, resulting in a substantial reduction in

the cost per terabyte of storage, which in turn makes it reasonable to model all your data. The basic idea behind Hadoop is to perform cost-effective data analysis present across world wide web!

7. Upcoming Technologies using Hadoop:

With reinforcing its capabilities, Hadoop is leading to phenomenal technical advancements. For instance, HBase will soon become a vital Platform for Blob Stores (Binary Large Objects) and for Lightweight OLTP (Online Transaction Processing). Hadoop has also begun serving as a strong foundation for new-school graph and NoSQL databases, and better versions of relational databases.

8. Hadoop is getting cloudy!

Hadoop is getting cloudier! In fact, cloud computing and Hadoop are synchronizing in several organizations to manage Big Data. Hadoop will become one of the most required apps for cloud computing. This is evident from the number of Hadoop clusters offered by cloud vendors in various businesses. Thus, Hadoop will reside in the cloud soon!

Advantages of Hadoop

- Ability to store a large amount of data.
- High flexibility.
- Cost effective.
- High computational power.
- Tasks are independent.
- **Open Source**: Hadoop is open-source in nature, i.e. its source code is freely available. We can modify source code as per our business requirements. Even proprietary versions of Hadoop like Cloudera and Horton works are also available.
- **Schema Independent:** Hadoop can work on different types of data. It is flexible enough to store various formats of data and can work on both data with schema (structured) and schema-less data (unstructured).
- **Scalable:** Hadoop is a storage platform that is highly scalable, as it can easily store and distribute very large datasets at a time on servers that could be operated in parallel.
- **Cost effective:** Hadoop is very cost-effective compared to traditional database-management systems.
- **Fast:** Hadoop manages data through clusters, thus providing a unique storage method based on distributed file systems. Hadoop‘s unique feature of mapping data on the clusters provides a faster data processing.
- **Flexible:** Hadoop enables enterprises to access and process data in a very easy way to generate the values required by the company, thereby providing the enterprises with the tools to get valuable insights from various types of data sources operating in parallel.
- **Failure resistant:** One of the great advantages of Hadoop is its fault tolerance. This fault resistance is provided by replicating the data to another node in the cluster, thus in the event of a failure, the data from the replicated node can be used, thereby maintaining data consistency.
- **High Throughput and Low Latency:** Throughput means amount work of done per unit time and Low latency means to process the data with no delay or less delay. As Hadoop is driven by the principle of distributed storage and parallel processing, Processing is done simultaneously on each block of data and independent of each other. Also, instead of moving data, code is moved to data in the cluster. These two contribute to High Throughput and Low Latency.
- **Data Locality:** Hadoop works on the principle of ‒Move the code, not data‖. In Hadoop, Data remains Stationary and for processing of data, code is moved to data in the form of tasks, this is known as Data Locality. As we are dealing with data in the range of petabytes, it becomes both difficult and expensive to move the data across Network, Data locality ensures that Data movement in the cluster is minimum.

- **Support for Multiple Languages:** Although Hadoop was mostly developed in Java, it extends support for other languages like Python, Ruby, Perl, and Groovy.
- **Support for Various File Systems**

  Hadoop is very flexible in nature. It can ingest various formats of data like images, videos, files, etc. It can process Structured and Unstructured data as well. Hadoop supports various file systems like JSON, XML, Avro, Parquet, etc.

Drawbacks or disadvantages of Hadoop

Following are the drawbacks or **disadvantages of Hadoop**:

- It is not suitable for small and real time data applications.
- Joining multiple data set operations are complex.
- It does not have storage or network level encryption.
- Cluster management is hard i.e. in cluster, operations like debugging, distributing software, collection logs etc. are too hard.
- When operated by a single master it will cause difficulty in scaling.
- Programming model is very restrictive.

What Is Big Data Analytics?

Big Data Analytics is a complete process of examining large sets of data through varied tools and processes in order to discover unknown patterns, hidden correlations, meaningful trends, and other insights for making data-driven decisions in the pursuit of better results.

Big data analytics refers to the strategy of analyzing large volumes of data, or big data. This big data is gathered from a wide variety of sources, including social networks, videos, digital images, sensors, and sales transaction records. The aim in analyzing all this data is to uncover patterns and connections that might otherwise be invisible, and that might provide valuable insights about the users who created it. Through this insight, businesses may be able to gain an edge over their rivals and make superior Big data analytics is the use of advanced analytic techniques against very large, diverse data sets that include structured, semi-structured and unstructured data, from different sources, and in different sizes from terabytes to zetta bytes.

Big data is a term applied to data sets whose size or type is beyond the ability of traditional relational databases to capture, manage and process the data with low latency. Big data has one or more of the following characteristics: high volume, high velocity or high variety. Artificial intelligence (AI), mobile, social and the Internet of Things (IoT) are driving data complexity through new forms and sources of data. For example, big data comes from sensors, devices, video/audio, networks, log files, transactional applications, web, and social media — much of it generated in real time and at a very large scale.

Types of Big data analytics

1. Prescriptive Analytics

The basis of this analytics is predictive analytics but it goes beyond the three mentioned above to suggest the future solutions. It can suggest all favorable outcomes according to a specified course of action and also suggest various course of actions to get to a particular outcome. Hence, it uses a strong feedback system that constantly learns and updates the relationship between the action and the outcome.

The computations include optimization of some functions that are related to the desired outcome. For example, while calling for a cab online, the application uses GPS to connect you to the correct driver from among a number of drivers found nearby. Hence, it optimizes the distance for faster arrival time. Recommendation engines also use prescriptive analytics.

The other approach includes simulation where all the key performance areas are combined to design the correct solutions. It makes sure whether the key performance metrics are included in the solution. The optimization model will further work on the impact of the previously made forecasts. Because of its power to suggest favorable solutions, prescriptive analytics is the final frontier of advanced analytics or data science, in today‘s term.

The most valuable and most underused big data analytics technique, prescriptive analytics gives you a laser-like focus to answer a specific question. It helps to determine the best solution among a variety of choices, given the known parameters and suggests options for how to take advantage of a future opportunity or mitigate a future risk. It can also illustrate the implications of each decision to improve decision-making. Examples of prescriptive analytics for customer retention include next best action and next best offer analysis.

- Forward looking
- Focused on optimal decisions for future situations
- Simple rules to complex models that are applied on an automated or programmatic basis
- Discrete prediction of individual data set members based on similarities and differences
- Optimization and decision rules for future events

2. Diagnostic Analytics

Diagnostic analytics is used to determine why something happened in the past. It is characterized by techniques such as drill-down, data discovery, data mining and correlations. Diagnostic analytics takes a deeper look at data to understand the root causes of the events. It is helpful in

determining what factors and events contributed to the outcome. It mostly uses probabilities, likelihoods, and the distribution of outcomes for the analysis.

In a time series data of sales, diagnostic analytics would help you understand why the sales have decrease or increase for a specific year or so. However, this type of analytics has a limited ability to give actionable insights. It just provides an understanding of causal relationships and sequences while looking backward.

A few techniques that uses diagnostic analytics include attribute importance, principle components analysis, sensitivity analysis, and conjoint analysis. Training algorithms for classification and regression also fall in this type of analytics

Data scientists turn to this technique when trying to determine why something happened. It is useful when researching leading churn indicators and usage trends amongst your most loyal customers. Examples of diagnostic analytics include churn reason analysis and customer health score analysis. Key points:

- Backward looking
- Focused on causal relationships and sequences
- Relative ranking of dimensions/variable based on inferred explanatory power)
- Target/dependent variable with independent variables/dimensions
- Includes both frequentist and Bayesian causal inferential analyses

3. Descriptive Analytics

This can be termed as the simplest form of analytics. The mighty size of big data is beyond human comprehension and the first stage hence involves crunching the data into understandable chunks. The purpose of this analytics type is just to summarize the findings and understand what is going on.

Among some frequently used terms, what people call as advanced analytics or business intelligence is basically usage of descriptive statistics (arithmetic operations, mean, median, max, percentage, etc.) on existing data. It is said that 80% of business analytics mainly involves descriptions based on aggregations of past performance. It is an important step to make raw data understandable to investors, shareholders and managers. This way it gets easy to identify and address the areas of strengths and weaknesses such that it can help in strategizing.

The two main techniques involved are data aggregation and data mining stating that this method is purely used for understanding the underlying behavior and not to make any estimations. By mining historical data, companies can analyze the consumer behaviors and engagements with their businesses that could be helpful in targeted marketing, service improvement, etc. The tools used in this phase are MS Excel, MATLAB, SPSS, STATA, etc.

This technique is the most time-intensive and often produces the least value; however, it is useful for uncovering patterns within a certain segment of customers. Descriptive analytics provide insight into what has happened historically and will provide you with trends to dig into in more detail. Examples of descriptive analytics include summary statistics, clustering and association rules used in market basket analysis. Key points:

- Backward looking
- Focused on descriptions and comparisons
- Pattern detection and descriptions
- MECE (mutually exclusive and collectively exhaustive) categorization
- Category development based on similarities and differences (segmentation)

4. Predictive Analytics

As mentioned above, predictive analytics is used to predict future outcomes. However, it is important to note that it cannot predict if an event will occur in the future; it merely forecasts what the probabilities of the occurrence of the event are. A predictive model builds on the preliminary descriptive analytics stage to derive the possibility of the outcomes.

The essence of predictive analytics is to devise models such that the existing data is understood to extrapolate the future occurrence or simply, predict the future data. One of the common applications of predictive analytics is found in sentiment analysis where all the opinions posted on social media are collected and analyzed (existing text data) to predict the person‘s sentiment on a particular subject as being- positive, negative or neutral (future prediction).

Hence, predictive analytics includes building and validation of models that provide accurate predictions. Predictive analytics relies on machine learning algorithms like random forests, SVM, etc. and statistics for learning and testing the data. Usually, companies need trained data scientists and machine learning experts for building these models. The most popular tools for predictive analytics include Python, R, Rapid Miner, etc.

The prediction of future data relies on the existing data as it cannot be obtained otherwise. If the model is properly tuned, it can be used to support complex forecasts in sales and marketing. It goes a step ahead of the standard BI in giving accurate predictions.

The most commonly used technique; predictive analytics use models to forecast what might happen in specific scenarios. Examples of predictive analytics include next best offers, churn risk and renewal risk analysis.

- Forward looking
- Focused on non-discrete predictions of future states, relationship, and patterns
- Description of prediction result set probability distributions and likelihoods
- Model application
- Non-discrete forecasting (forecasts communicated in probability distributions)

Why corporate sector/Business sector needs a big Data Strategy

Businesses and Big Data Analytics

Big Data analytics tools and techniques are rising in demand due to the use of Big Data in businesses. Organizations can find new opportunities and gain new insights to run their business efficiently. These tools help in providing meaningful information for making better business decisions.

The companies can improve their strategies by keeping in mind the customer focus. Big data analytics efficiently helps operations to become more effective. This helps in improving the profits of the company.

Big data analytics tools like Hadoop helps in reducing the cost of storage. This further increases the efficiency of the business. With latest analytics tools, analysis of data becomes easier and quicker. This, in turn, leads to faster decision making saving time and energy.

As the world becomes smarter and smarter, data becomes the key to competitive advantage, meaning a company‘s ability to compete will increasingly be driven by how well it can leverage data, apply analytics and implement new technologies. In fact, according to the International Institute for Analytics, by 2020, businesses using data will see $430 billion in productivity benefits over competitors who are not using data.

So, it‘s clear that data is now a key business asset, and it‘s revolutionizing the way companies operate, across most sectors and industries. In effect, every business, regardless of size, now needs to be a data business. And if every business is a data business, every business therefore needs a robust data strategy.

It all starts with strategy

Having a clear data strategy is absolutely vital when you consider the sheer volume of data that is available these days. I see too many businesses get so caught up in the Big Data buzz that they collect as much data as possible, without really considering what they want to do with all that data. While others are so overwhelmed by options that they bury their head in the sand. Neither represent a smart way to run a business.

Instead of starting with the data itself, every business should start with strategy. At this stage, it doesn‘t matter what data is out there, what data you‘re already collecting, what data your competitors are collecting, or what new forms of data are becoming available. Neither does it matter whether your business has mountains of analysis-ready data at your disposal, or next to none. A good data strategy is not about what data is readily or potentially available – it‘s about what your business wants to achieve, and how data can help you get there.

Therefore, if companies want to avoid drowning in data, they need to develop a smart strategy that focuses on the data they really need to achieve their goals. To be truly useful in a business sense, data must address a specific business need, help the organisation reach its strategic goals, and generate real value. This means you need to define the key challenges and business-critical questions that need answering, and then collect and analyse the data that will help you address them.

A lot of companies with data strategies nestled within different areas of the business, such as marketing and sales. That‘s not enough. Every business needs a company-wide data plan. Unfortunately, there is also still a widespread perception among business executives that data and analytics is purely an IT matter. And as with all IT matters, this means they don‘t really need to understand how it works. They simply need to know what it does – drive growth – and throw money at it. In my experience, data strategies that are driven by the IT team tend to focus on data storage, ownership and integrity rather than the business‘s long-term strategic goals and how data can help reach those goals. That‘s why the data strategy should be owned by the leadership team.

It is also important to remember that no one type of data is inherently better than any other kind. Using data strategically is about finding the best data for your company, and that may be very different to what‘s best for another company. With so much data available these days, the trick is to focus on finding the exact, specific pieces of data that will best benefit your organisation.

How Big data analysis helps the corporate sector?

Big data is a term that describes the large volume of data – both structured and unstructured – that inundates a business on a day-to-day basis. But it‘s not the amount of data that‘s important. It‘s what organizations do with the data that matters. Big data can be analyzed for insights that lead to better decisions and strategic business moves.

The new benefits that big data analytics brings to the table, however, are speed and efficiency. Whereas a few years ago a business would have gathered information, run analytics and unearthed information that could be used for future decisions, today that business can identify insights for immediate decisions. The ability to work faster – and stay agile – gives organizations a competitive edge they didn‘t have before.

The use of Big Data is becoming common these days by the companies to outperform their peers. In most industries, existing competitors and new entrants alike will use the strategies resulting from the analyzed data to compete, innovate and capture value.

Big data analytics helps organizations harness their data and use it to identify new opportunities. That, in turn, leads to smarter business moves, more efficient operations, higher profits and happier customers. In his report *Big Data in Big Companies*, IIA Director of Research Tom Davenport interviewed more than 50 businesses to understand how they used big data. He found they got value in the following ways:

- **Cost reduction.** Big data technologies such as Hadoop and cloud-based analytics bring significant cost advantages when it comes to storing large amounts of data – plus they can identify more efficient ways of doing business.
- **Faster, better decision making.** With the speed of Hadoop and in-memory analytics, combined with the ability to analyze new sources of data, businesses are able to analyze information immediately – and make decisions based on what they‘ve learned.
- **New products and services.** With the ability to gauge customer needs and satisfaction through analytics comes the power to give customers what they want. Davenport points out that with big data analytics, more companies are creating new products to meet customers‘ needs.

The key elements of a good data and analytics strategy

To create a robust data and analytics strategy, business leaders need to consider many factors. Here are the critical points.

- **Your data needs** – In order to find the right data for you, you must first define how you want to use data. You may need certain types of data for some goals and different types of data for others.
- **How you will source and gather the data** – Having identified what you are looking to achieve with data, you can now start to think about sourcing and collecting the best data to meet those needs. There are many ways to source and collect data, including accessing or purchasing external data, using internal data and putting in place new collection methods.
- **How that data will be turned into insights** – As part of any solid data strategy, you need to plan how you will apply analytics to your data to extract business-critical insights that can inform decision making, improve operations and generate value.
- **Technology infrastructure requirements** – Having decided how you want to use data, what kind of data is best for you, and how you might want to analyzed that data, the next step in creating a robust data strategy is considering the technology and infrastructure implications of those decisions. Specifically, this means deciding on the software or hardware that will take your data and turn it into insights.
- **Data competencies within the organisation** – In order to get the most out of data it is essential to cultivate certain skills. There are two main routes to developing data-related competencies in your organisation: boosting your in-house talent, and outsourcing the data analysis.

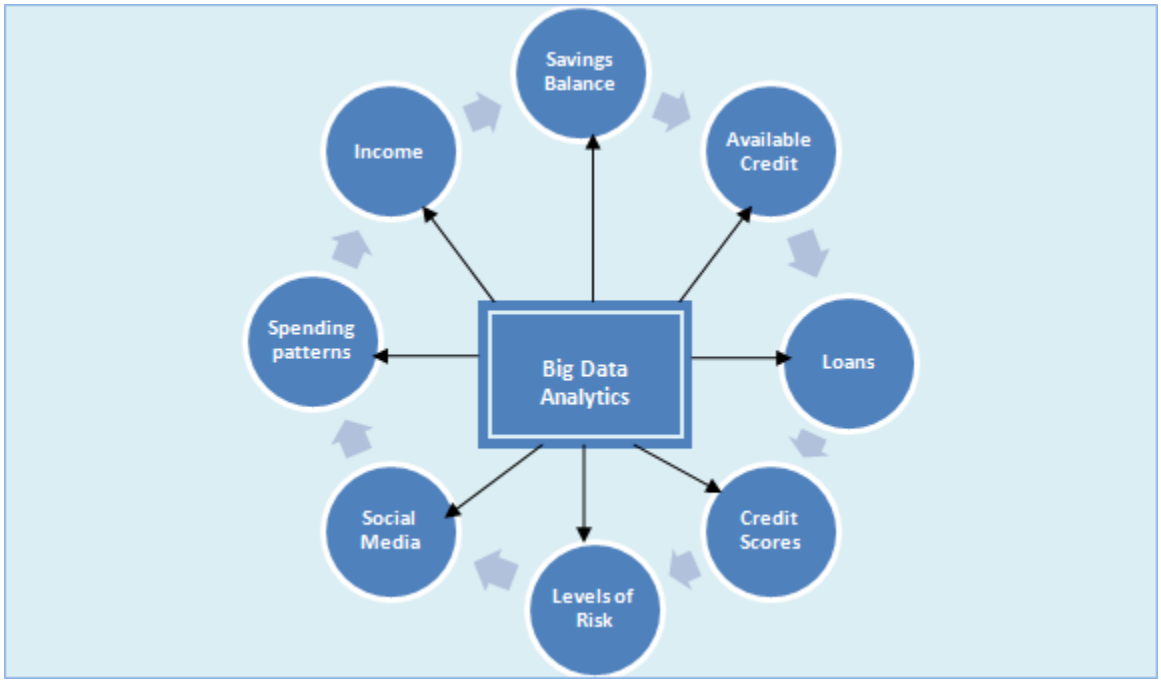
- **Data governance** – Collecting and storing data, especially personal data, brings serious legal and regulatory obligations. Therefore, it is vital any organisation factor data ownership, privacy and security issues into their data strategy. Ignoring these issues, or failing to properly address them, could see data go from a huge asset to a huge liability.

Real-time Benefits of Big Data Analytics(Use/Applications of Big Data)

There has been an enormous growth in the field of Big Data analytics with the benefits of the technology. This has led to the use of big data in multiple industries ranging from

- Banking
- Healthcare
- Energy
- Technology
- Consumer
- Manufacturing

There are many other industries which use big data analytics. Banking is seen as the field making the maximum use of Big Data Analytics.



The education sector is also making use of data analytics in a big way. There are new options for research and analysis using data analytics. The institutional data can be used for innovations by technical tools available today. Due to immense opportunities, Data analytics has become an attractive option to study for students as well.

The insights provided by the big data analytics tools help in knowing the needs of customers better. This helps in developing new and better products. Improved products and services with new

insights can help the firm enormously. This may help the customers too as they get better offerings satisfying their needs effectively. All in all, Data analytics has become an essential part of the companies today.

Statistical Models, Analysis of Variance, Data Dispersion, Contingence and Correlation, Regression Analysis, Precision and Error limits.

Unit VI: Technological Aggregation & Case Studies

Contents

Modern trends in IOT: Wearable, industrial standards. Case studies using IoTs, connected use cases in Real-life and smart cities, Case studies: Greenhouse monitoring, smart health care monitoring, smart home automation, smart car parking, Smart Agriculture Monitoring, air pollution monitoring, smart industrial automation.

https://appinventiv.com/blog/iot-trends/

What citizens expect from their cities?

https://community.singularitynet.io/t/what-do-you-think-citizens-want-from-smart-cities/1815

https://www.linkedin.com/pulse/10-things-citizens-expect-from-indias-20-smart-cities-arumugham/

https://www.scnsoft.com/blog/iot-for-smart-city-use-cases-approaches-outcomes

Modern trends in IOT

Modern Trends and challenges in IoT

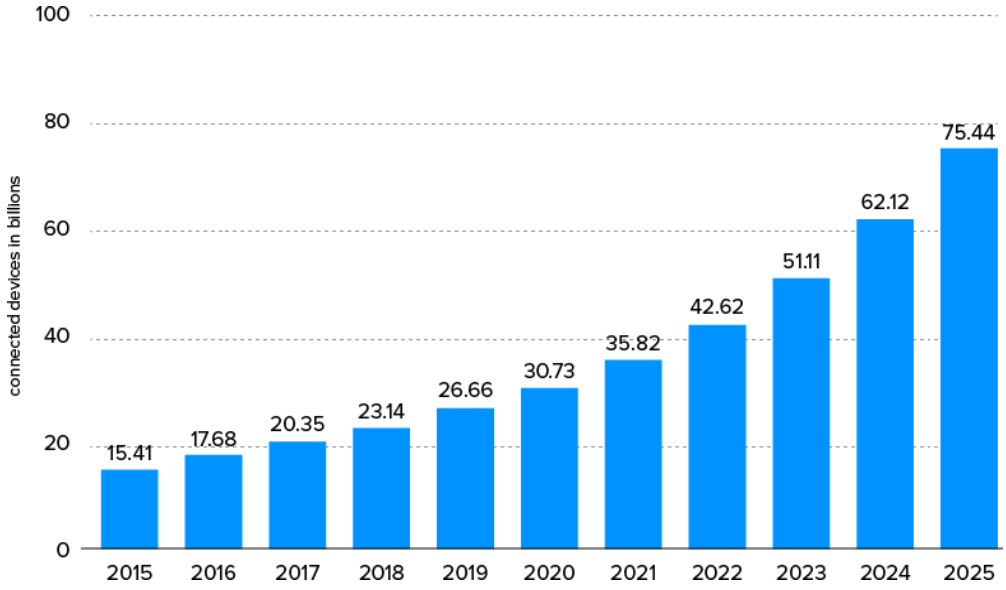
The Internet of Things can now be considered a mature technology with a multitude of options both in terms of connectivity, hardware and use cases already available on the market. 2019 was an exceptional year for the IoT, which experienced higher than expected growth in terms of connected devices. In fact, according to IoT Analytics, active IoT devices reached 9.5 billion in 2019 much more than the 8.3 billion previously estimated.

That the IoT is a leading sector is also attested by other very positive data, such as its penetration within companies. Vodafone highlights that today more than one-third of companies use IoT solutions to optimize processes, reduce operating costs, improve data collection, create new revenue streams or increase revenue from existing ones. It‘s also interesting to notice that very often these companies prefer to buy off the shelf solutions rather than developing their own ones in-house.

IoT – the concept which did not even exist a decade ago has today not just gotten mainstream but has also marked a presence, across industries, across the globe. With its market size poised to be 1.6 trillion by the time we reach 2025, entrepreneurs and businesses from all corners are finding opportunities to enter the segment.

1. Rise in Number & Types of IoT Devices

Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions)



2. Security.

Security remains a major concern for those developing and using IoT solutions today. This is comprehensible given the **more than 100 million attacks in the first half of 2019 alone**. A cyber security startup, Armis, is also one of the **first billionaire exits** in 2020. It is evident, hence, that this will be an area in which investment and the research of the necessary countermeasures will be a growing trend this year and in the years to come. Many experts also observe how the use of blockchain could be a key to achieve higher security in the IoT.

3. SaaS.

Software as a Service is gradually becoming the norm in the IoT world. Local and customized solutions leave room for Cloud solutions. The benefits are significant for both end customers and service providers. The former profit from a considerable cost reduction; the latter have the opportunity to develop increasingly sophisticated solutions and to sell them, directly or on marketplaces, on a larger scale. The recent release of our **Professional Public Servers** also follows this trend. Our SaaS LoRaWAN Network Server solution in fact was created to support IoT companies in their growth without unnecessary setup costs.

4. Edge Computing.

The reduction in cost and, on the other hand, the increased power of the devices used in the IoT make it possible to use the increased computing power of the devices to process data collected on the edge (on the device itself, before sending over the data) and allow huge bandwidth savings. In addition, in many cases, it ensures a greater compliance with privacy regulations.

5. Data Analytics

Increased adoption of IoT solutions results in increased data collection and transmission of all types of data. Managing and analyzing this data represents the real added value that IoT can bring. Data analysis, Machine Learning and Artificial Intelligence will therefore play an increasingly important and decisive role. Our direct experience also tells us that knowing how to analyse and understand data is also a crucial element for an efficient management of a network, especially if it is a large deployment. For this reason, we are developing solutions, shortly available, able to support our customers with these activities.

6. Smart City.

Cities are the core centers of our society, where about 60% of the world's population lives. In the urban environment, the services that people use can be significantly improved thanks to the IoT solutions already available on the market. Pioneer cities in this area are showing how much value the Internet of Things is generating and many other cities are therefore following their example so that smart city will certainly become mainstream in 2020.

7. Industrial Internet of Things (IIoT).

The manufacturing industry has now realized that Predictive Maintenance, Energy and Resources management solutions are increasing productivity, enabling greater optimization of resources and processes, reducing operating costs and improving operator safety. The percentage of Proofs of Concept (POCs) resulting in a commercial solution has more than doubled in the last 2 years and the path of digital transformation, involving also traditional industry sectors, is now well established.

8. Smart home.

The voice-recognition algorithms are now sufficiently accurate and performing. So, the devices equipped with voice user interface are spreading rapidly thanks to the affordable prices, due to the strong competition among the big players operating in this market. Their pervasive diffusion represents an important driver in the adoption of home automation and smart home solutions as these devices can become the access hub for dozens of services suitable for any home environment. Will 2020 be the year of smart homes? Maybe you should ask your voice assistant...

9. Healthcare.

Medical research is one of the sectors in which most investment is made, and innovations are not just about new drugs or biomedical implants. Chronic disease monitoring using smart devices, post-surgery monitoring, or early warning systems for patients at risk, smart tricorders and wearable devices, medical and diagnostic smart tools for clinicians, and devices to make hospital rooms "smart". According to Frost & Sullivan, the Internet of Medical Things (IoMT) is expected to grow up to $72.02 billion by 2021.

10. Environment.

2019 was the year in which the global population has become more aware of climate change and these issues have entered the agendas of governments and corporations also driven by greater public attention and pressure on these issues. The IoT can play a vital role in tackling these global challenges and preserving the planet. In fact, when designing IoT solutions, purely economic and higher productivity objectives are combined with environmental protection, waste reduction and better use of resources. This category also includes solutions for monitoring and protecting endangered animal species and their habitats, as well as disaster solutions (floods, fires, avalanches) prevention and management.

11. Global Connectivity.

Regardless of the transmission technology used to collect data from sensors and then process them, solutions to support processes of international or global reach such as asset tracking and management in food, manufacturing or medical applications require immediate and always-on connectivity to ensure consistency and process security. For companies that work globally or aspire to do so, having global connectivity will be a must. Something that network providers will have to deal with and resolve during the year that has just begun. We are in the process of doing that.

12. Smart Homes

Even people who discarded smart homes as devices for pretentious youngsters are now finding it difficult to ignore the capabilities the technology comes with. While it started with a steady growth, the demand for connected home devices will see a sharp rise in the years to come.

This demand, in turn, will increase the need for the manufacturing of connected electronic devices.

13. Powering Smart Store

The use of radio frequency identification tags has been powering smart stores. RFID makes inventory management a lot easier and fun as opposed to the use of conventional data capture systems.

IoT can also record how visitors spend their time on the store premises. It does this by tracking their movement and product interactions with the help of smart lighting.

Using data collected by such systems, store owners can make the needed improvements to their inventory and even the way products are displayed on their shelves.

14. Big Data, Analytics, and Machine Learning

Even now, distributed IoT analytics and data are finding applications in IoT networks. This method lets systems trigger alerts or action sans transferring volumes of data to network cores. This results in improved performance as networks operate at low latency. The integration of data streams with machine learning and AI engines is another trend. Potential applications include smart homes and elevator maintenance, to name a few. Find out more by reading the most current big data statistics.

Integrated analytics are now being embedded into solutions as providers seek to speed up data analysis. Such analytics are directly fed into machine learning apps. This design supports IoT devices, processes and infrastructure adaptation and optimization. IoT data will be sold as a

commodity, which will be targeted mostly at appliance manufacturers. Selling IoT data will be used to prove product viability. Recent machine learning statistics reflect these developments.

15. IoT in Manufacturing

IoT uses sensors, which can help managers identify machinery problems early on. Problems are identified before technicians are sent out to work on issues. Sensors are very handy for people handling preventive maintenance. IoT has been finding applications in smart factories. The use of wearables gives workers a clear picture of factory conditions and safety.

IoT and wearables connect factory floors to management. Wearables can connect to mobile devices. This improves time management and safety and allows for feedback. Innovations in preventive maintenance are coming every day. They will not stop anytime soon—all thanks to IoT. This development is clear among the most recent industrial IoT trends.

Trends in wearable technology

The Global wearable technology market is expected to grow at a 17.66% over the forecast period (2019-2024).

Wearable technology is an emerging trend that integrates electronics to the daily activities and fits into the changing lifestyles. It can be worn on any part of the body, but it is usually used as an accessory. The ability to connect to the Internet and enable data exchange between a network and a device is the driving factor that leads to the wearable technology trend.

Advancements in recent research causing increased innovation were instrumental in increasing the demand for the wearable market, which has led to new product categories like the smart fabrics, smart shoes, etc., that incorporate high-end technology and design in daily living. Lately, the focus has been on providing aesthetic design to devices to attract the customers.

The smart watch category is also experiencing a boom owing to additional features, like the brand that suits everyday lifestyle. Companies competing in this category are Apple, Asus, LG, Huawei, Samsung, and Sony. Strong brands, such as Apple and Fossil, are keep pricing consistent with the price bands of traditional watches to maintain the revenues. With Google's WearOS, many other premium watchmakers, such as TAG, Armani, etc., have entered the segment.

Although the wearable technology is still in its early stages, the niche segments are expected to have a longer-term perspective. Changing consumer lifestyles and shift toward fitness, along with the need for advanced technologies for healthcare and security applications, are expected to drive the market over the forecast period.

1. Fitness Trackers:

Fitness trackers are becoming more popular due to the increasing cases of health issues that mainly attribute to the sedentary lifestyle. Here **Xiaomi Mi Band 2S** that is the improved version is affordable and capable of capturing health data on a wider scope. With this gadget, you will be able to keep track of your health even when you are asleep. Moreover knowing the health data, trackers also send you real-time texts and the missed call notifications. Thereby you will not miss important calls and texts when you are exercising. At end of the day, you will evaluate your progress and get better on tactics to reduce fitness injuries.

2. Medical Wearables:

Medical Wearables are in an increasing demand because of its affordability and convenience. They help to take routine checkups that not only comes at a cost but also expensive. The latest development has seen that these gadgets are capable of collecting highly-accurate health data and connect remotely to the personal physician to check your health progress.

Wearable IoT in healthcare

With wearable apps and devices, doctors can easily track a patient's condition and determine critical insights, leading to a more advanced diagnosis process. Wearables can be a powerful solution to save both time and resources in hospitals. For example, Zephyr Any where‘s BioPatch, an FDA-approved device, can be attached to a patient's chest to observe their condition and deliver real-time data to their doctors. This device can send alerts to doctors' watches or smartphones, promoting more intensive care. Moreover, wearable IoT could effectively incorporate wheelchairs, artificial limbs and other supportive devices.

Wearable innovations extend to hearing aids as well. For instance, the ReSound LiNX2 allows a user to control their hearing through an iPhone app. With great inventions in wearable tech, it is easy to collect complex information that addresses several health-related concerns. Get ready to see more innovative mobile apps and wearables in the healthcare sector.

3. Smart Ring:

Smart ring is one of the latest trends in wearable technology. The gadget is popular among individuals and businessmen as they spend most of their time in meetings and wish to get notifications without much attention at their smart watches or other mobile devices. In addition, this wearable is very useful, while shopping. You can use it to access to your car. With this wearable, you can keep an eye on your biometric doings data like number of steps made during the day and the calories burnt. Different to the normal ring, you need not worry if you misplace it. You can track its position easily by GPS connection from other mobile devices.

4. More intelligent smart watches:

Today, smart watches work as supporting devices to enhance the mobile experience, but soon they will become smarter. In 2018, we can expect advanced watches that not only match smartphones but may replace the need for them. Though we can already monitor daily activities and health records using our smart watches, we may get information in a more usable form in the future. For example, we could determine glucose levels in our blood, link it to our schedule, monitor our stress levels throughout the day, and distinguish the people and meetings who give us the most stress.
Wearable apps and devices could also emerge as the most efficient solution for handling other IoT devices. The home and workplace are both moving toward intelligent devices, and this technology will undoubtedly demand a powerful app that helps the user control all the devices simultaneously.

5. Hearable Devices:

Similar to Fitness Trackers, Hearable Devices are popular in the market. These devices are used for multimedia applications, but a large number of hearable devices are used in medical applications. They are popular among patients with hearing impairment issues. Thereby hearable devices have earned a substantial momentum in the recent times. The unstable growth of the AI voice assistants has Apple, and Google racing to put your whole smartphone just in a headset.

6. Assault Protection Wearable:

Lately, there has been a rise in the figure of sexual assault cases especially in women. So to help them protect from any assault, Assault Protection Wearable is best to assist you. This wearable is camouflaged to the garment and is made to sense falls and force. The rape protection is linked to the phone via Bluetooth where you may select to either close the functionality in case of false alarm or turn on an emergency rape button if it is needed. In addition, your phone will send texts if you fail to close the app within 5 seconds confirming distress communiqué. On the other end, the contacted emergency assistance will be recording the evidence to present for legal action.

7. Smart Clothing:

Smart clothing, high tech clothing or smart fabrics, are clothing items that have been enhanced with technology to improve your life. High tech clothing may make life easier in some remarkable ways. It connects to a program on a secondary device by Bluetooth or Wi-Fi to let you enjoy life smartly. From increasing sleep quality, improving muscle recovery, recording fitness activity to recommending new workouts to the wearer, Smart Clothing has turned normal life into exciting one.

8. Military or Security Agencies

This is another area which makes a very serious business case for the wearable technology. Smart glasses, smart watches, sensor enabled rings, smart helmets are becoming increasingly popular among the security troops and they becomes a very light aid for the users in combat, they do not have to carry heavy equipment to track something or to get in sync with the team.

9. Enhanced customer data collection

Apart from improving the user experience, wearable technology will open new business opportunities for digital marketers. Popular wearable gadgets such as Google Glass and smart watches will allow marketers to easily collect essential information on buying habits and locations of the target audience. Entrepreneurs and app developers will also see new opportunities from the trend. Advanced wearable devices mean developers will need to build apps for new systems and come up with better data collection methods using wearables and mobiles.

We have already seen smart glasses providing more appealing and advanced user experiences. Tech giants Google, Intel and Microsoft are all striving to come up with glasses based on augmented reality. Soon we may even see high-tech smart clothes on the market. Undoubtedly, smart accessories and clothing will open up new business opportunities, and the trend continues to turn into a billion-dollar industry.

10. GPS technology

Navigation is another sector where there will be significant developments in the year ahead. A combination of wearables and real-time tracking can detect locations efficiently. Advanced wearable devices and smartphones today come with gyroscopes, compasses, accelerometers and motion trackers. With all this collected data, smartwatches and apps can determine cadence, speed, distance traveled and total calories burnt.

Accelerometers, compasses and gyroscopes can combine to build a joint motion sensor. Apple has already added motion-tracking coprocessors to its smartphone series. 2018 may be a crucial year for the revolutionary use of GPS technology.

11. Women's safety and wearable devices

This list wouldn't be complete without talking about wearable apps that will be used to protect women against violence and harassment. Per the National Sexual Violence Resource Center, 91 percent of victims of sexual assault in the United States are female, and 1 in 5 women will be raped in their lifetime. These numbers point to a serious problem that must be addressed from multiple perspectives. A few companies have already launched wearable apps and devices to help women find assistance more quickly or disorient a possible attacker. Such wearables can often camouflage into a woman's outfit, look like jewelry pieces or blend into key chains.

Smart sleepwear, **Smart active wear**, and **Smart casual wear** are examples of different types of high tech clothing that are latest trends in wearable technology.

Conclusion:

Truly health apps and wearables have boosted the scale of remote health-monitoring, letting for better communiqué between patients and doctors. Wearable sensors can take out medical information from the patient and send them to healthcare providers immediately. Healthcare apps make it achievable to view patient history, get a reminder for medication, and many more. Thus the use of wearable technology has become a requirement because of their convenience and affordability.

With the latest trends, these devices will help you surely in many ways. Not only as this technology eased healthcare services, but also it has made life easier. As the blog has made a precise note above, it is practical to believe on wearable technology. In regard to this, it needs you to upgrade to the latest trends in the technology and enjoy more benefits that associate with upgrades.

What do you think citizens want from smart cities?

While the definition of a smart city is still evolving, but currently it can be thought of as a city that can leverage information and communication technologies to enhance public services, citizen health and well-being and sustainability.

Smart cities leverage connected technologies and bring many benefits, let‘s examine a few of them?

Data-driven autonomous decision making

Advancements in connected devices that have access to _big data‘ allow smart cities to access information that‘s on a level not before realized. Data analytic strategies will enable the public sector to access and analyze a huge volume of real-time information, gaining far greater levels of understanding than currently possible. This will lead to an exponential growth in service level achievement and citizen satisfaction.

Enhanced citizen and government engagement

Expanding upon the current digitization of our public services such as digital local government gateways will make smart cities afar more attractive place for people to live and promote a connected citizen experience. Free easily accessible government data available to all will lead to an explosion of increased services levels from vendors, holographic interactive maps, city wide performance dashboards, Blockchain levels of transparency into city budgeting, live-streamed interconnected city media, all congeal together to form the emergent state of a city being _smart‘ and creating a closer bond with the citizens that populate them.

Safer communities

Taking advantage of technological advances such as smart traffic control systems that can use license plate recognition, violence detectors, live interconnected crime prevention centres, next-gen emergency service response will create a zero or near zero crime rates.

Reduced environmental footprint

With the rise of smart cities and the increased consumption that comes with them, the cities of the future will employ a number of strategies to cope and reduce the negative effects on the environment. Energy efficient buildings, atmospheric air sensors, and renewable energy sources created within the city itself will provide these smart cities with the tools needed to shrink their ecological impact while becoming self-sustaining.

Improved transportation

Smart public transportation systems are expected to rise exponentially over the next five years, interconnected transportation systems will drastically enhance efficiencies throughout a city.

A cities transportation system could be considered the arteries and veins of the city and with the implementation of technologies such as intelligent traffic control systems that can optimize traffic flow, relieving congestion during peak travel times and allow emergency services to navigate unabated will have a massive effect on the efficiency of the city itself.

The number of autonomous vehicles on our roads is sset to expload, and as this technology becomes mainstream there will be no need of traffic signals or road markings at all as all the trafic flow can be controlled by smart traffic control systems.

Increased digital equity

Smart city technology has the ability to create a stable environment for citizens who all have access to high-speed internet services important smart technologies should be affordable and accessible for all citizens. We already have an abundance of free public Wi-Fi hotspots strategically placed throughout a city, but in the city of the future internet will be omnipresent and free to all residents.

Economic development opportunities

Investment in smart city technology means massive expansion in citywide and indeed national GDP growth. Opportunities for companies of all sizes to team up with local governments to invest in smart city infrastructure and initiatives.

Increased efficiency of public utilities

With a limited supply of natural resources available to meet human demand coupled with a population growth of people wishing to live within urban environments, smart technologies are provide the tools needed to effectively conserve and reduce consumption of vital resources of water and electricity while moving us away from oil and gas reliance. Smart sensors and utility grids will allow quick identification of any leaks or over usage, enabling fast repairs to any damaged pipework and reassign power.

Improved infrastructure

Ageing transportation infrastructure is a major cost to maintain and repair. Smart predictive analytic technologies can identify areas that need to be fixed before there is an infrastructure failure. These smart interconnected technologies can communicate and transmit data showing any adverse structural changes in buildings and report back the need for inspections or maintenance. This usage of smart technology represents a massive opportunity for cities to save huge amounts of money and resources not to mention saving lives.

Increased employment engagement

Deploying smart and autonomous technologies will increase companies ability to understand what their customers really want and react in real time. It will also reduce the manual tasks workers currently face every day affording more time citizens can dedicate to more worthwhile persutes such as spending time with their families, education, sport and leisure time.

Access to Better Public Transport

Growing urbanisation has increased the number of private vehicles on public roads, leading to massive traffic congestion in almost all Indian cities. All selected Smart Cities have undertaken to develop or strengthen their public transportation networks to encourage their increased use and thereby reduce the use of private vehicles. The Smart Cities propose to provide easy access to public transport and enhance mobility by use of ICT (Information & Communications Technology) solutions. Public transport will aid faster, easier and cheaper commuting, and the modal shift from private to public transport will be instrumental in significantly mitigating inner city congestion.

Putting Pedestrians First

_Pedestrianisation‘ and Non-Motorized Transport are integral factors in the proposed _smartening up‘ programme of cities like Pune, Belagavi, Udaipur and Chennai, the latter also being the first city to implement a Non-Motorized Transport Policy in India. Citizens will enjoy wide footpaths with public seating at regular intervals and easy mobility for the differently-abled. Car-free Sundays will make streets available for citizens to interact and engage in street activities.

Since pedestrians are the victims in a large number of major road accidents, on-street parking will be managed and organized to increase their safety. Also, cycle sharing and feeder systems will help citizens achieve better Last Mile connectivity, which is the major hurdle for the successful functioning of public transport. Citizens will have the option to use cycles to commute to their destinations from the public transport mode.

Availability of Adequate Parking

Intelligent parking management, a part of many of the winning Smart City proposals, will help citizens find parking with ease and even pre-book their parking slot along with online payment modes. On-street parking management will be a reliable revenue source for the cities, and can be used to further strengthen their public transportation systems. Managing on-street parking will reduce traffic congestion, increase the effective carriageway width available for vehicles and reduce fuel consumption and pollution, among other benefits. The cities which have identified this as a prime winning proposition in the Smart City contest are Bhubaneshwar, Davanagere, Indore, Udaipur, Guwahati and Chennai.

Reduced Traffic Congestion

Intelligent Traffic Management Systems to manage city traffic via various ICT solutions have been considered by Bhubaneswar, Surat, Ahmedabad, Davanagere, Indore, Udaipur and Chennai. Citizens will enjoy easier transport modes and routes, and also have smart phone access to estimated travel time to their destination by Passenger Real Time information on arrival of buses,

trains and e-rickshaws. Traffic Signaling Prioritization of BRTS buses and video surveillance will further ensure safety and prevent traffic violations.

Safer Living

Safety of residents is another aspect that has been assured in Coimbatore, Kakinada, Udaipur, Guwahati and Chennai. Initiatives such as LED Street lighting will boost pedestrians‘ safety, as will video surveillance via a Common Control Centre – which will simultaneously help reduce traffic violations and ensure efficient on-street parking management.

Hassle-Free Civic Services

Another big initiative that the identified Smart Cities are targeting is e-governance, meaning a single platform from where citizens can access all and any details and also help them get all services done. Most importantly, this will enable citizens‘ engagement in all aspects of city functioning, as the data sharing or transparency between government and citizens will act as a forum for citizens to understand exactly how their city is functioning. Some of the smart solutions such as integrated fare cards, smart unified city governance, ‗one city one website‘, GIS (Geographic Information System) mapping and Wi-Fi hotspots have been considered by Bhubaneswar, Kochi, Ahmedabad, Vishakhapatnam, Davanagere, New Delhi Municipal Council (NDMC), Belagavi, Ludhiana and Bhopal.

Safety Against Natural Disasters

Natural disasters are, almost by definition, impossible to prevent; however human interventions in terms of precautionary measures can help in minimizing losses to life and property to a great extent. The coastal areas are often badly affected by cyclones and flooding, so cities like Chennai and Vishakhapatnam will concentrate on ICT-based disaster management techniques like sensors, weather forecasts, zero flooding zones, storm water management, etc. to make them safer places to live in.

Neighborhood Sanitation

Solid Waste Management through smart solutions for clean roads and a healthy environment is considered as an important factor by cities of Jaipur, Jabalpur, Indore and Kakinada. Recycling of waste will produce renewable energy, ensure safe disposal of solid waste, prevent soil and environmental pollution and reduce depletion of resources.

Easy Access to All Basic Infrastructure

Smart Cities aim to maintain basic infrastructure with best quality and 100% efficiency. The efficiency of the utilities in our cities has been an elusive factor till date, thanks to inadequate monitoring and responsiveness. Electricity, sewerage, storm water drainage and water supply will be strengthened in the Smart Cities with a smart layer of ICT applications. Citizens in Pune, Kochi, Solapur, NDMC, Kakinada and Belagavi will benefit from ICT-enabled initiatives such as zero loss monitored by Smart Meters, LED street lighting, 24x7 water supply by source augmentation, waste water recycling and sensors to detect sewer system leakages. Pune is focusing on healthcare for low income households and providing training in digital literacy, and Sholapur is incentivizing conservation of water. Chennai aims to create water sources using desalination plants and recycling water to use for various purposes.

City Beautification

All identified Smart Cities are focusing on developing more _lung spaces' within the city. Smart components like cycling, street furniture, jogging tracks, designated spaces hawkers, etc. will enhance the aesthetics of the city. Green spaces will get a new dimension with new soft and hard landscapes equipped with Wi-Fi hotspots, providing ideal areas for citizens to relax, exercise and interact. All this also plays a major role in creating a healthy and a sustained environment.

In Short…

Smart City citizens stand to benefit significantly if the smart cities initiatives are implemented effectively. Implementation through the formation of SPVs is integral to success of these proposals and will determine the success of Smart Cities Mission as a whole; failure to consider this route seriously can prove to be a major stumbling block.

Smart City citizens can expect their city authorities and the involved nodal agencies to work efficiently towards finalizing and implementing the committed proposals, keeping them involved through the citizen's engagement process, and to see their respective cities compete successfully with others in attracting investments.

IoT use cases for smart cities

IoT-enabled smart city use cases span multiple areas: from contributing to a healthier environment and improving traffic to enhancing public safety and optimizing street lighting. Below, we provide an overview of the most popular use cases that are already implemented in smart cities across the globe.

Road traffic

Smart cities ensure that their citizens get from point A to point B as safely and efficiently as possible. To achieve this, municipalities turn to IoT development and implement smart traffic solutions.

Smart traffic solutions use different types of sensors, as well as fetch GPS data from drivers‘ smart phones to determine the number, location and the speed of vehicles. At the same time, smart traffic lights connected to a cloud management platform allow monitoring green light timings and automatically alter the lights based on current traffic situation to prevent congestion. Additionally, using historical data, smart solutions for traffic management can predict where the traffic could go and take measures to prevent potential congestion.

For example, being one of the most traffic-affected cities in the world, Los Angeles has implemented a smart traffic solution to control traffic flow. Road-surface sensors and closed-circuit television cameras send real-time updates about the traffic flow to a central traffic management platform. The platform analyzes the data and notifies the platform users of congestion and traffic signal malfunctions via desktop user apps. Additionally, the city is deploying a network of smart controllers to automatically make second-by-second traffic lights adjustments, reacting to changing traffic conditions in real time.

Smart parking

With the help of GPS data from drivers‘ smartphones (or road-surface sensors embedded in the ground on parking spots), smart parking solutions determine whether the parking spots are occupied or available and create a real-time parking map. When the closest parking spot becomes free, drivers receive a notification and use the map on their phone to find a parking spot faster and easier instead of blindly driving around.

Public transport

The data from IoT sensors can help to reveal patterns of how citizens use transport. Public transportation operators can use this data to enhance traveling experience, achieve a higher level of safety and punctuality. To carry out a more sophisticated analysis, smart public transport solutions can combine multiple sources, such as ticket sales and traffic information.

In London, for instance, some train operators predict the loading of train passenger cars on their trips in and out of the city. They combine the data from ticket sales, movement sensors, and CCTV cameras installed along the platform. Analyzing this data, train operators can predict how each car will load up with passengers. When a train comes into a station, train operators encourage passengers to spread along the train to maximize the loading. By maximizing the capacity use, train operators avoid train delays.

Utilities

IoT-equipped smart cities allow citizens to save money by giving them more control over their home utilities. **IoT enables different approaches to smart utilities:**

- ***Smart meters & billing***

With a network of smart meters, municipalities can provide citizens with cost-effective connectivity to utilities companies‘ IT systems. Now, smart connected meters can send data directly to a public utility over a telecom network, providing it with reliable meter readings. Smart metering allows utilities companies to bill accurately for the amount of water, energy and gas consumed by each household.

- ***Revealing consumption patterns***

A network of smart meters enables utilities companies to gain greater visibility and see how their customers consume energy and water. With a network of smart meters, utilities companies can monitor demand in real time and redirect resources as necessary or encourage consumers to use less energy or water at times of shortage.

- ***Remote monitoring***

IoT smart city solutions can also provide citizens with utility management services. These services allow citizens to use their smart meters to track and control their usage remotely. For instance, a householder can turn off their home central heating using a mobile phone. Additionally, if a problem (e.g., a water leakage) occurs, utilities companies can notify householders and send specialists to fix it.

Street lighting

IoT-based smart cities make maintenance and control of street lamps more straightforward and cost-effective. Equipping streetlights with sensors and connecting them to a cloud management solution helps to adapt lighting schedule to the lighting zone.

Smart lighting solutions gather data on illuminance, movement of people and vehicles, and combine it with historical and contextual data (e.g., special events, public transport schedule, time of day and year, etc.) and analyze it to improve the lighting schedule. As a result, a smart lighting solution ‒tells‖ a streetlight to dim, brighten, switch on or switch off the lights based on the outer conditions.

For instance, when pedestrians cross the road, the lights around the crossings can switch to a brighter setting; when a bus is expected to arrive at a bus stop, the streetlights around it can be automatically set brighter than those further away, etc.

Waste management

Most waste collection operators empty containers according to predefined schedules. This is not a very efficient approach since it leads to the unproductive use of waste containers and unnecessary fuel consumption by waste collecting trucks.

IoT-enabled smart city solutions help to optimize waste collecting schedules by tracking waste levels, as well as providing route optimization and operational analytics.

Each waste container gets a sensor that gathers the data about the level of the waste in a container. Once it is close to a certain threshold, the waste management solution receives a sensor record, processes it, and sends a notification to a truck driver‘s mobile app. Thus, the truck driver empties a full container, avoiding emptying half-full ones.

Environment

IoT-driven smart city solutions allow tracking parameters critical for a healthy environment in order to maintain them at an optimal level. For example, to monitor water quality, a city can deploy a network of sensors across the water grid and connect them to a cloud management platform. Sensors measure pH level, the amount of dissolved oxygen and dissolved ions. If leakage occurs and the chemical composition of water changes, the cloud platform triggers an output defined by the users. For example, if a Nitrate (NO3-) level exceeds 1 mg/L, a water quality management solution alerts maintenance teams of contamination and automatically creates a case for field workers, who then start fixing the issue.

Another use case is monitoring air quality. For that, a network of sensors is deployed along busy roads and around plants. Sensors gather data on the amount of CO, nitrogen, and sulfur oxides, while the central cloud platform analyzes and visualizes sensor readings, so that platform users can view the map of air quality and use this data to point out areas where air pollution is critical and work out recommendations for citizens.

Public safety

For enhancing public safety, IoT-based smart city technologies offer real-time monitoring, analytics, and decision-making tools. Combining data from acoustic sensors and CCTV cameras deployed throughout the city with the data from social media feed and analyzing it, public safety solutions can predict potential crime scenes. This will allow the police to stop potential perpetrators or successfully track them.

For example, more than 90 cities across the United States use a gunshot detection solution. The solution uses connected microphones installed throughout a city. The data from microphones passes over to the cloud platform, which analyzes the sounds and detects a gunshot. The platform measures the time it took for the sound to reach the microphone and estimates the location of the gun. When the gunshot and its location are identified, cloud software alerts the police via a mobile app.

Iterative approach to implementing smart city solutions

The range of smart city applications is highly diverse. What they have in common is the approach to implementation. Whether municipalities plan to automate waste collection or improve street lighting, they should start with the foundation – a basic smart city platform. If a municipality prefers to expand the range of smart city services in future, it will be possible to upgrade the existing architecture with new tools and technologies without having to rebuild it.

Here is a six-step implementation model to follow for creating an efficient and scalable IoT architecture for a smart city.

Stage 1: basic IoT-based smart city platform

To be able to scale, smart city implementation should start with designing a basic architecture – it will serve as a springboard for future enhancements and allow adding new services without losing functional performance. A basic IoT solution for smart cities includes four components:

- **The network of smart things**

A smart city – as any IoT system – uses smart things equipped with sensors and actuators. The immediate goal of *sensors* is to collect data and pass it to a central cloud management platform. *Actuators* allow devices to act - alter the lights, restrict the flow of water to the pipe with leakage, etc.

- **Gateways**

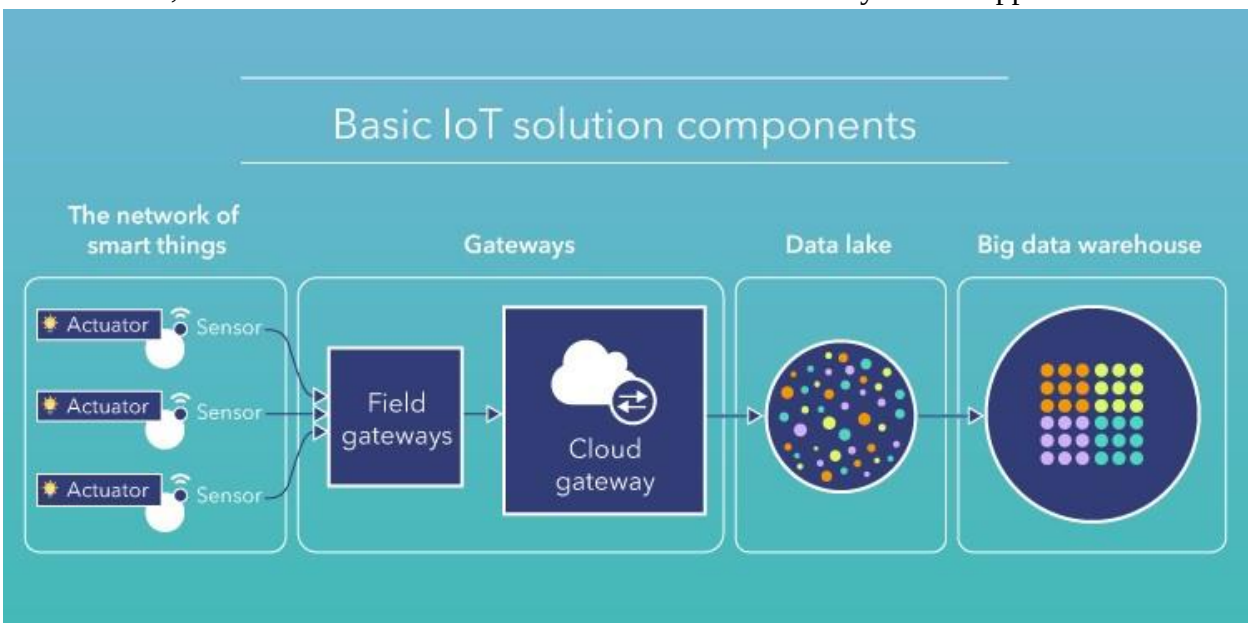
Any IoT system comprises two parts – a ‒tangible‖ part of IoT devices and network nodes and a cloud part. The data cannot simply pass from one part to the other. There must be doors – *field gateways*. Field gateways facilitate data gathering and compression by preprocessing and filtering data before moving it to the cloud. *The cloud gateway* ensures secure data transmission between field gateways and the cloud part of a smart city solution.

- **Data lake**

The main purpose of a data lake is to store data. Data lakes preserve data in its raw state. When the data is needed for meaningful insights, it‘s extracted and passed over to the big data warehouse.

- **Big data warehouse**

A big data warehouse is a single data repository. Unlike data lakes, it contains only structured data. Once the value of data has been defined, it‘s extracted, transformed and loaded into the big data warehouse. Moreover, it stores contextual information about connected things, e.g., when sensors were installed, as well as the commands sent to devices‘ actuators by control applications.



Stage 2: Monitoring and basic analytics

With data analytics, it is possible to monitor devices‘ environment and set rules for control applications (we cover them at stage 4) to carry out a particular task.

For example, analyzing the data from soil moisture sensors deployed across a smart park, cities can set rules for the electronic valves to close or open based on the identified moisture level. The data collected with sensors can be visualized on a single platform dashboard, allowing users to know the current state of each park zone.

Stage 3: Deep analytics

Processing IoT-generated data, city administrations can go beyond monitoring & basic analytics and identify *patterns* and hidden *correlations* in sensor data. Data analytics uses advanced techniques like machine learning (ML) and statistical analysis. ML algorithms analyze historical sensor data stored in the big data warehouse to identify trends and create predictive models based on them. The models are used by control applications that send commands to IoT devices‘ actuators. Here is how it applies in practice.

Unlike a traditional traffic light that is programmed to display a particular signal for a definite period, a smart traffic light can adapt signal timings to the traffic scenario. ML algorithms are applied to historical sensor data to reveal traffic patterns and adjust signal timings, helping to improve average vehicle speed and avoid congestions.

Stage 4: Smart control

Control applications ensure better automation of smart city objects by sending *commands* to their *actuators*. Basically, they ‒tell‖ actuators what to do to solve a particular task. There are *rule-based* and *ML-based* control applications. Rules for rule-based control applications are defined manually, while ML-based control applications use models created by ML algorithms. These models are identified based on data analysis; they are tested, approved and regularly updated.

Stage 5: Instant interacting with citizens via user applications

Along with the possibility of automated control, there should always be an option for users to influence the behavior of smart city applications (for example, in case of emergency). This task is carried out by user applications.

User applications allow citizens to connect to the central smart city management platform to monitor and control IoT devices, as well as receive notifications and alerts. For example, using GPS data from drivers‘ smartphones, a smart traffic management solution identifies a traffic jam. To prevent even bigger congestion, the solution automatically sends a notification to the drivers in the area, encouraging them to take a different route.

At the same time, employees at a traffic control center who use a desktop user app receive a _congestion alert.‘ To relieve the congestion and re-route part of the traffic, they send a command to the traffic lights‘ actuators to alter the signals.

Stage 6: Integrating several solutions

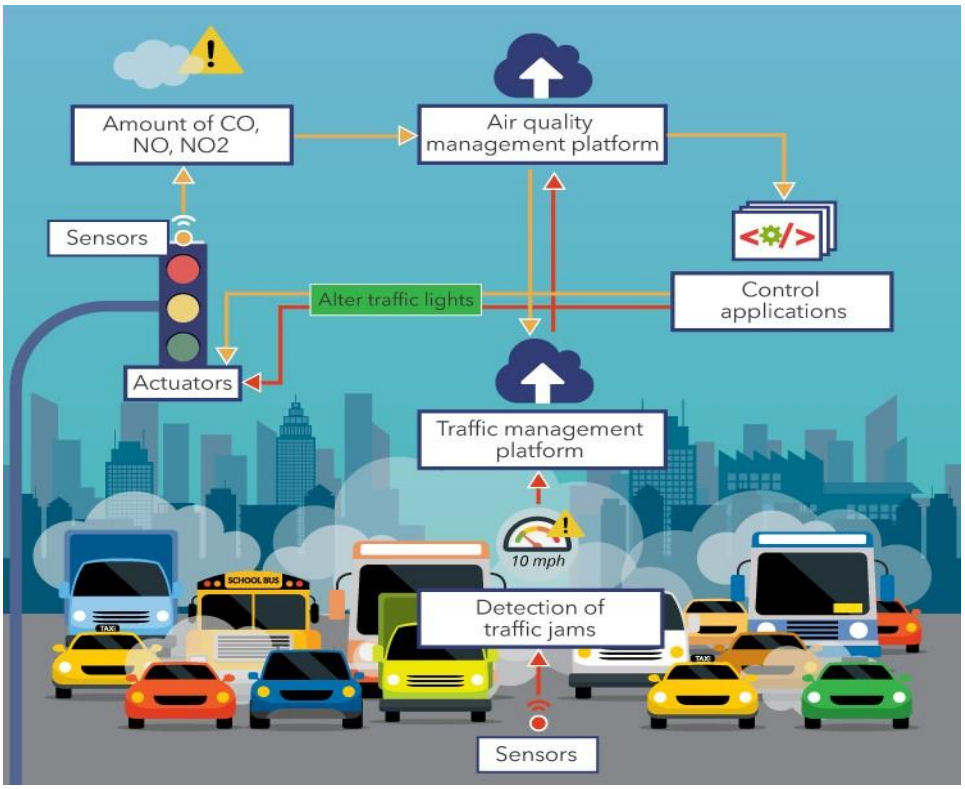
Achieving ‒smartness‖ is not a one-time action – it is a continuous process. Implementing IoT-based smart city solutions today, municipalities should think of services they might like to implement tomorrow. It implies not only increasing the number of sensors but, more importantly, the number of functions. Let‘s illustrate this functional scalability with the example of a smart city solution for traffic monitoring.

A city deploys a traffic management solution to detect traffic jams in real time and manage traffic lights to reduce traffic in the areas with intensive traffic. After some time, the city decides to ensure city traffic doesn‘t harm the environment and integrates the traffic management solution

with a smart air quality monitoring solution. Cross-solution integration allows controlling both traffic and air quality in the city dynamically.

Example 1: Traffic light control and Air Pollution Monitoring

For that, traffic lights or street lights along the roads can be equipped with sensors that monitor air quality. Sensors measure the amount of CO, NO, and NO_2 in the air and pass data records to a central air quality management platform for processing. If the amount of harmful gases in the air is critical, control applications apply rules or use models to take an output action, e.g., ‗alter traffic lights.' Before that, there is a need to make sure that altering traffic lights won't cause accidents or blockages in other areas. It is possible due to the integration of the traffic management solution to the air quality management solution. The traffic management platform performs real-time analysis and identifies if it is possible to alter the traffic lights. If altering the lights is acceptable, control applications send a command to the traffic lights' actuators, which execute the command.



Example 2:

INTRODUCTION

The Smart City represents nowadays hot topic in terms of improving living conditions. The Internet of Things (IoT) is currently considered as a basic communication infrastructure for smart cities, where machines communicate automatically between each other. The biggest advantage is the cooperation of many different communication technologies and devices within one functional system, where a big amount of information and data are shared and used in a secure and smart way.

EXISTING SYSTEM

In the existing system, garbage is collected by the corporation by weekly once or by 2 days once. Though the garbage strikes and overflows the garbage bin and spread over the roads and pollutes the environment which produces air pollution causes several diseases.

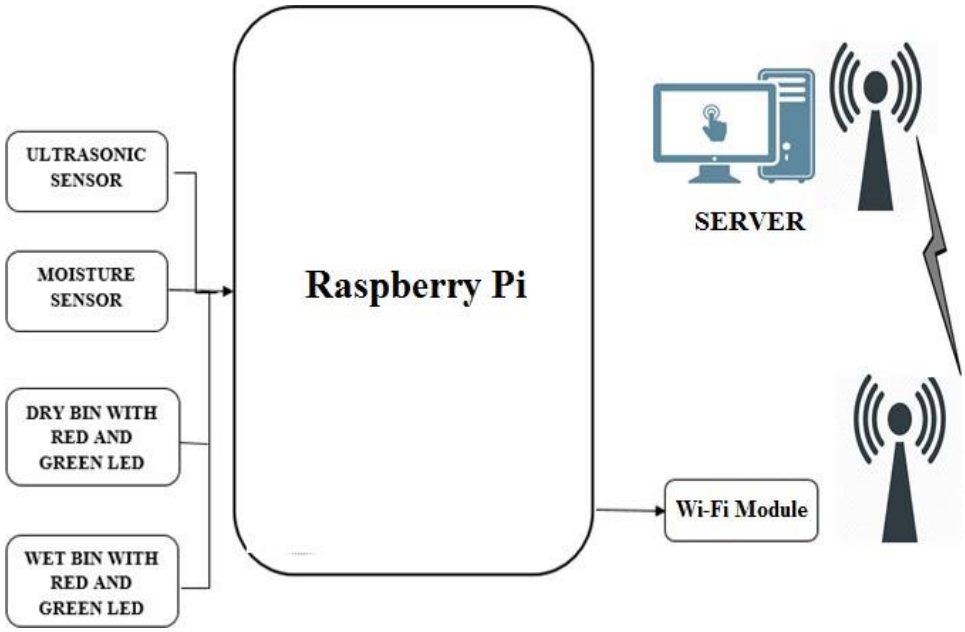
PROPOSED SYSTEM

In this Proposed System consists of a mechanical setup that separates the waste and Manages the waste in a different type of box by using automation. In this system, we detect the dry and wet waste also and this kind of waste we separate by automatically. By having such a system in multiple places every thing is monitored and controlled by centralized node using IoT technology

Ultrasonic Sensor – Distance measurement

Humidity sensor – Moisture

BLOCK DIAGRAM



BLOCK DIAGRAM DESCRIPTON

Raspberry Pi is interfaced with the ultrasonic sensor and Moisture Sensor, so it acts as a heart of the system. The ultrasonic sensor gives the distance that is How far the waste object is present in the bin and humidity sensor gives it is dry or wet waste. All the category of waste is separated by 2 separate bins

HARDWARE TOOLS

- Raspberry Pi
- Ultrasonic sensor
- Moisture Sensor

SOFTWARE TOOLS

- Raspbian Jessie OS
- Language: Python

CONCLUSION

The article introduced the upcoming IoT infrastructure for smart cities and put it in the context of municipal waste management. Given overview showed that it is not yet enough discussed the possibility of using genetic algorithms as an optimization method for waste collection. This system result is based on the idea of IoT infrastructure, which should provide enough information to handle this Smart City issue more efficiently.

Business model scenario for Internet of things

Business models continue growing and innovate from ancient times till this date.

- Ancient time models: _barter resource with one another‘.
- Later models
- Purchase, add value and sale,
- Use of currency

Business Model Growing Continuously

- Plan, purchase raw material, manufacture a product on bigger scale, distribute and sale and profit‘
- Innovations in model continues

Business Model Taking Into Account Many Factors

- Competitive advantage
- Experience curves
- Value chain
- Theory of portfolio of products and services,
- Core competencies of business organization and generic strategies

Business Model Definition

- A conceptual structure, supporting the viability of a business, including its purpose, its goals and its ongoing plans for achieving them.
- ―A business model is an abstract representation of an organisation. Representation may be conceptual, textual, and/or graphical.‖ (Wikipedia)

Abstract Representations in Business Model

- Representation for the all core interrelated architectural, co-operational, and financial arrangements
- Includes many activities, present and future
- Includes core products and/or services the organization offers, or will offer
- Architecture includes organizational infrastructure and technological architecture

Term Business Model

- Refers to _uses of a range of informal and formal descriptions to represent core aspects of a business
- Business process, strategy, practices, and operational processes, policies including culture

Business Model Focus

- Not only on financial goals but also on the business sustainability or
- Establishing a corporate culture when offering value to customers.

Key Resources at Input Stages in Business Model Scenarios For The IoT

- Sensors, M2M, sensor networks data
- Data using web APIs for multiple information sources data
- Open sources data and corporate databases

- Mobile services network information data
- Knowledge databases

Real-time Monitoring Applications in Business Model Scenarios for the IoT

- Using Real-time M2M, IoT sensors and sensors network data and generating events and monitor the systems
- Real-time anomalies and fault detection
- Maintenance scheduling
- Fault or anomaly detection.
- Event analytics enabling insight knowledge into the business processes
- Enabling the distributed processes
- Enabling the efficient processes
- Streamlining the business processes.
- **Predictive maintenance and service**:
  Current data from sensors hooked up to your critical machinery is compared real-time to historical data to detect and report any possible chances of malfunction. You can now predict and prevent network connectivity and downtime issues by taking early corrective actions. The result is lower maintenance costs and higher productivity.
- **Operations & Consumer Insights powered by Big Data:**
  The IoT has led to a veritable explosion of data generated from each of the devices in a connected network. This data can be fed to analytics and decision support tools to deduce valuable inferences. Big data analysis can yield unexpected insights in day to day operations as well as customer and market facing operations and help shape strategic business decisions.
- **Supply Chain Management:** IoT can bring high levels of efficiency into supply chain management. You can track shipments in real time through a combination of RFID sensors, tracking devices and communication networks (3G, 4G, internet etc). It can help optimize operations and avoid delays by developing transit routes that save shipment time. It can transmit real-time weather and traffic information, so that your deliveries are always on-time.
- **Inventory management:** IoT can be used to monitor stock levels and place procurement orders in advance. The data generated from various points of delivery and sale can be used to forecast demand, optimize pricing and understand usage patterns. Marketing and sales personnel can use this data to identify areas of potential growth and plan further action.

Business model scenario for Internet of things
Building blocks of a Business Model Canvas

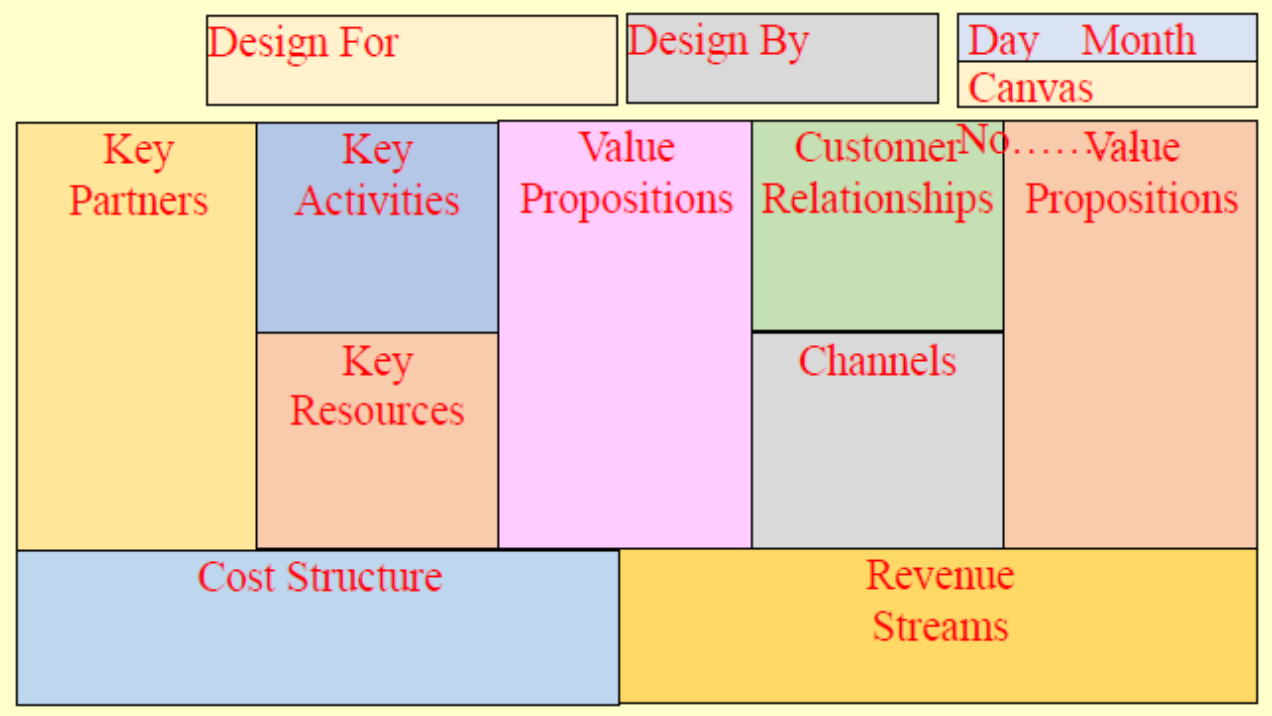


Fig. Building blocks of a Business Model Canvas

1. Key partners: Strategic alliances between competitors or non-competitors to optimize the operations and reduce risks of a business model
2. Key activities: Key activities execute a company‘s value proposition
3. Key resources: Key resources for sustaining and supporting the business and necessary for creation of the value to customers.
4. Value propositions: Products and services offered, their features such as performance, efficiency, accessibility, price, cost, convenience, usability, and design, and how differing from competitors
5. Customer relationships: Identified type of relationship of the company to be created with their customers and targeted segments
6. Customer segments: Identified sets of customers, segments, client-groups and diverse groups based on the value propositions offered
7. Channels: Effective, fast, efficient and cost-effective channels to deliver its value proposition to its targeted customers
8. Cost structure: Cost constituents to be considered in offering the value propositions and services, and considerations of scopes of economies in the operations
9. Revenue stream: Identified types of income sources, such as income from sales of product and physical goods

The business model framework depicted in Figure includes four main perspectives of the business model, namely the value proposition, the customer, financials and the infrastructure. The components are not stand-alone but mutually influence each other.

The *value proposition* specifies what is actually delivered to the customer. This goes beyond the product or service offered. It describes which customer needs are satisfied and details what other quantitative (e.g., price or speed of service) and qualitative aspects (e.g., brand, design, cost/risk reduction) contribute to the offered value. In the Internet of Things we consider raw data about physical objects as well as any aggregated or processed information a core component of the value proposition.
The *customer perspective* includes the *customer segments* addressed by the company, such as related channels and customer relationships. The customer segments define the different groups of people that are served. Different types of customer segments can be distinguished: mass market vs. niche market, segmented vs. diversified or multisided platforms. Multisided platforms will exist, if two or more interdependent customer segments are served by the company (e.g. credit card companies). The company can reach its customers, respectively customer segments through different *channels*. These can be direct or indirect and owned by the company itself or by partners. Channels can be aligned to the different phases of the lifecycle, such as creating awareness for the value proposition, evaluation of the value proposition through the customer, purchase, delivery and after sales. *Customer relationships* are often determined by the channels used. Relationships
can range from very loose (self-service, automated services) to highly engaged (personal assistance, communities, co-creation).
The *financial perspective* comprises the costs as well as the revenues. The *revenue structure* depicts the sources and ways of revenue generation. Here, too, different types of revenue streams can be distinguished: asset sale, usage fee, subscription fee, lending / renting / leasing, licensing, brokerage fee, and advertising. The *cost structure* describes the most important costs (variable and fixed) inherent to the business model. The business model can be rather value or cost driven (cost leadership vs. differentiation strategy). Companies can use economies of scale or economies of scope to create a successful business model.
Key partners, *key activities* and *key resources* can be referred to as the *infrastructure components*. The *key resources* are the assets required to make the business model work. Key resources can be physical, intellectual, human or financial. The *key activities* describe the most important actions to be performed by the company in order to create, offer and market the value proposition. These can be producing, problem solving or developing and maintaining a platform, respectively network. *Key partners* are the network of suppliers and collaboration partners (strategic alliances, outsourcing partners, co-creation) the business model depends on.

Value Creation in the Internet of Things

Value creation means creation of a _smart tracking and logistic service‘ from the sensed IDs of RF IDs communication on internet, data analytics, data visualization and mobile communication for provisions of SMS to receiver and delivery confirmation

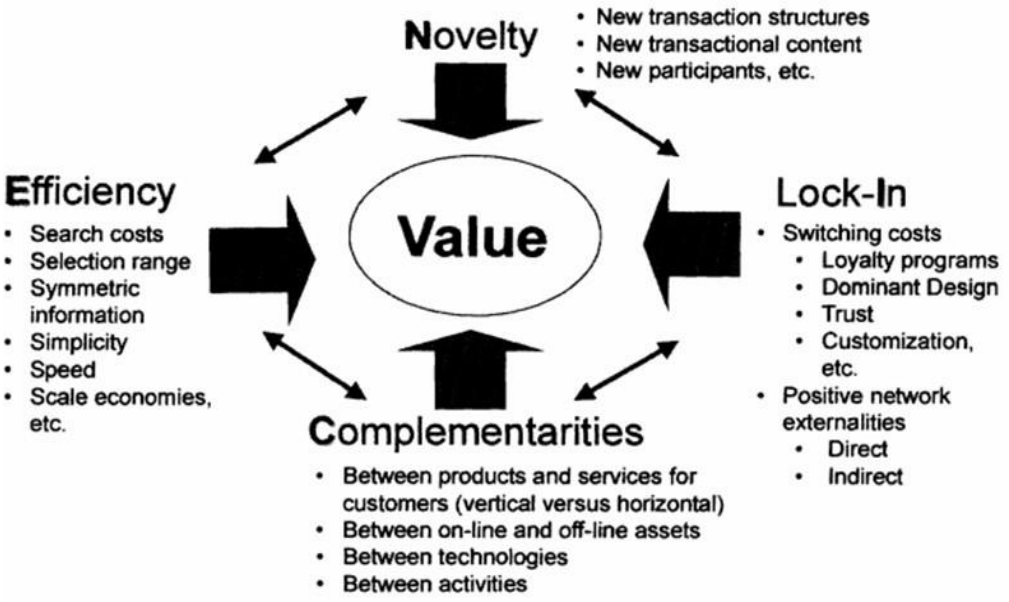
Value creation is the expansion of relationship enabled through internet and creation of new behaviors as a result.
Value creation is the heart of any business model. It involves performing activities that enhance the value of company‘s product or service (offering) and encourages customer willingness to pay.
When broadly defined, value creation is increasingly being recognized as a better management goal than strict financial measures of performance, many of which tend to place cost-cutting that produces short-term results ahead of investments that enhance long-term competitiveness and growth. As a result, some experts recommend making value creation the first priority for all

employees and all company decisions. "If you put value creation first in the right way, your managers will know where and how to grow; they will deploy capital better than your competitors; and they will develop more talent than your competition," Ken Favaro explained in Marakon Commentary. "This will give you an enormous advantage in building your company's ability to achieve profitable and long-lasting growth."

Features of Value Creation

- IoT enables addressing the emergent needs and real time needs using predictive analytics.
- Information convergence creates new experiences for current for current products. Information enable innovative services.
- IoT enables offering of product and services which can updated using the internet (such through as Wi-Fi) and create synergic value for the product.
- IoT enables value capture and thus recurrent revenue.
- Adds personalization and context, and uses networked products/services.
- Faster ecosystem functioning where multiple companies establish loose relationships among themselves or establish relationships with big company‘s

Sources of value creation in e-business



Efficiency By making each individual transaction more efficient, a business can decrease overall transaction costs. It has been found that the use of IT lowers transaction costs and by doing so, the transactions will create more value for the firm. Comparing businesses in traditional markets, relative to those in virtual markets, the efficiency can be elevated by reducing information asymmetry between buyers and sellers. The Internet can facilitate information flow to keep information up-to-date to be accessed conveniently by other parties. By improving one‘s information a firm can reduce both opportunistic behavior of suppliers, as well as the search and bargaining costs of customers. With the Internet‘s wide adoption, the transaction efficiency is further increased by faster and more well informed decision-making. Studies have been made

comparing offline and online transaction costs of car auctions, where it has been found that the transaction costs are significantly lower in online markets. The Internet also makes it possible for multiple transactions to occur simultaneously without the use of personnel resources, which further increases the efficiency of businesses. The Internet can be seen as a highly dense network, and as such can be compared to clusters of offline firms. Multiple studies have shown that denser offline networks also reduce transaction costs, due to similar reasons of increased efficiency such as better information flow.

Complementarities By complementarities refer to accompanying products or services that together provide more value than separately. A complementor in that case is a player whose product together with a firm‘s own product are valued more by customers than what would be the case of the firm's product alone. Complementarities are also highlighted in the resource based view of the firm as well as in network theory between participants. Complementarities are often split into two categories, vertical or horizontal complementarities. After-sales services are considered vertical, while bundles of items such as one-stop shopping is considered horizontal. By combining the resources and capabilities of different firms into an offer, more value can be created. Complementarities can also increase the efficiency from a customer perspective, such as airline companies also providing weather forecasts and currency exchange rates when booking an airline ticket.

Lock-in The potential to create value is also increased the more inclined customers are to repeat transactions. The transaction costs associated with a transaction by an established customer are lower than that of a new customer, while repeat transactions also lead to increased transaction volume. By locking-in customers, partners are further motivated to maintain and improve partnerships with the firm. Lock-ins can be achieved by implementing switching costs for customers, partners and competitors. Multiple e-businesses now use click-analysis, buying patterns, recommendations and reminder e-mails for restocking on previously bought items to keep customers, as well as to make their shopping as quick and convenient as possible. Buy using these types of data analysis on purchasing behaviors of customers, the next shopping experience will be tailored to each specific customer, thus creating a positive feedback loop, which motivates customers to use the same supplier multiple times. Furthermore, many e-businesses have developed communities around their business where customers can interact with one another. This increases customer loyalty and can in turn increase the lock-in effect. Other companies have managed for customers themselves to create value.

Novelty The introduction of ‒creative destruction‖ by Schumpeter illustrated how innovations can be sources of value. Extensive literature has regarded innovation in offline markets as sources of new value, such as manufacturing processes, product innovation and marketing, and recently increasing attention has been paid to business model innovation. E-businesses have provided great examples of innovating the business model. Amit and Zott provide examples of eBay, the first company to successfully introduce customer-to-customer auctions, and Pricelines.com who introduced reverse market listings. These companies created value by restructuring the business model and the way traditional transactions were made. Many ebusinesses simply eliminate much of the transaction costs associated with offline markets, such as Über (application-based taxi service which eliminates parts of the previous booking-process), PedidosYa (allowing customers to order food online with delivery by connecting a range of restaurants to their digital platform)

Value creation Model using information driven IoT value chain and using production/manufacture driven value chain.

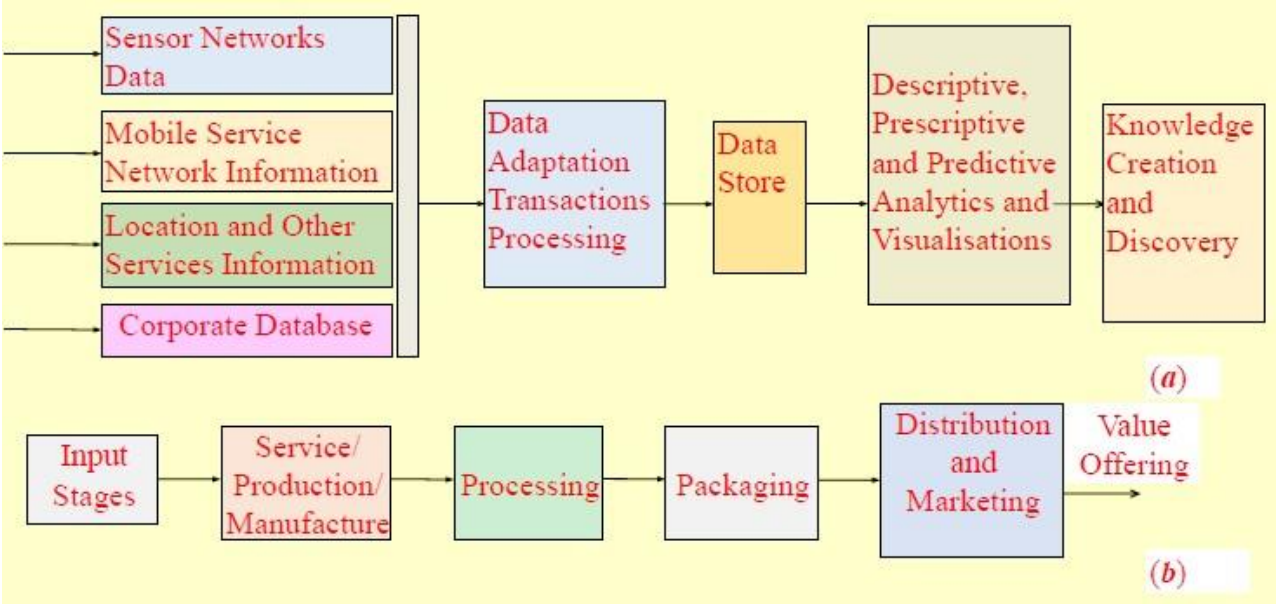


Fig. (a) and (b): Value creation using information driven IoT value chain and using production/manufacture driven value chain.

Value Chain means series of actions for creating value start with data collecting using APIs for the sensors/sensor networks/M2M data or from multiple information sources

The chain includes the actions using web APIs, open data, data from mobile-services network and corporate databases.

Fig (a) shows value proposition in production/manufacturing-driven value chain Fig. (b) shows value creation on deploying a value chain using information-driven IoT value chain in production/manufacturing/service.

IoT Cloud Platforms and Business Value Creation

This section illustrates how IoT is being utilized in different sectors of industries. Further, a hypothetical case study is performed involving electric bulb, waste bin, and tractor to show how these ―things‖ can give more business values other than their original and ordinary functional values.

Though IoT clouds vary in many factors, such as real-time data capture, data visualization, cloud type, data analytics, device configuration, API protocols, cost, and big data, the main purpose of IoT cloud is to facilitate intelligent integration and accessibility of the things/devices and associated surrounding ecosystem thus bridging the gap in the physical digital divide.

The following list includes several existing IoT cloud service providers that enable end-to-end IoT based analytics, storage, and real-time processing support to the customers and business farms: Xively, ThingSpeak, Plotly, Exosite, GroveStreams, Temboo, ThingWorx, Carriots, Nimbits, KAA, IBM IoT, Oracle Open IoT, Microsoft Research Lab of Things, SensorCloud, Ayla‘s IoT

Cloud Fabric, Arrayent Connect TM, Aer Cloud, thethings, io, SeeControl IoT, and Jasper Control Center.

The opportunities for the IoT cloud providers are of multifold. Following figure illustrates various areas where corporate investments are currently being sought for.

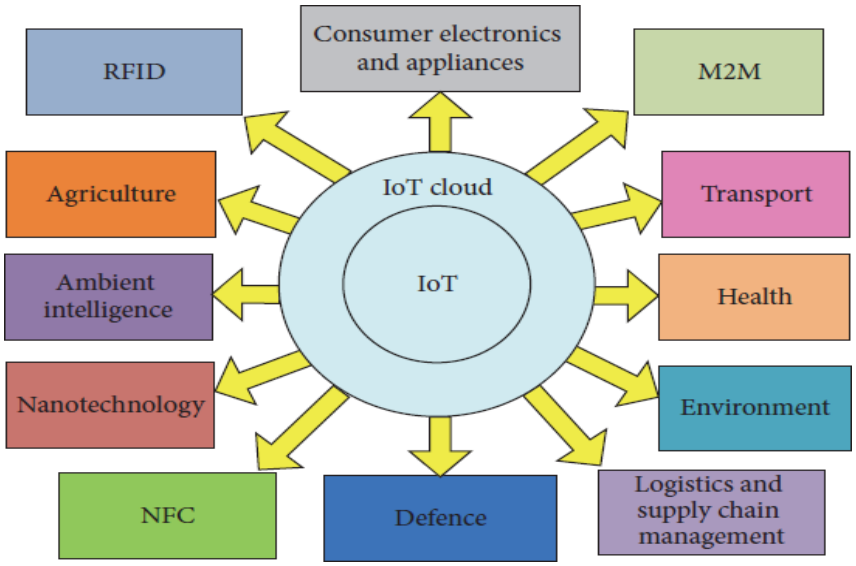


Figure: IoT cloud based investments.

It starts with leveraging a common plat form for consumer-electronics and appliances design sector, while heterogeneous Machine-to-Machine (M2M) communication platform is cumulating the digital telephony among the devices, ubiquitous transportation systems are being implemented at various smart cities, smart health care facilitation gives the fully connected ambience to the patients, and environment monitoring is deliberately getting huge attention.

Similarly, defense, security, and military forces seem to be keen on getting full exploration with IoT cloud using NFC (Near Field Communication) and innovative nanotechnology solutions at the war site. Smart agriculture systems are being popularized among the global farmers that use the intelligence of IoT cloud platform to result in a beginning of a new era of precision cumoptimized agriculture for better yield. Ambient intelligence is another area where IoT cloud is momentarily getting itself.

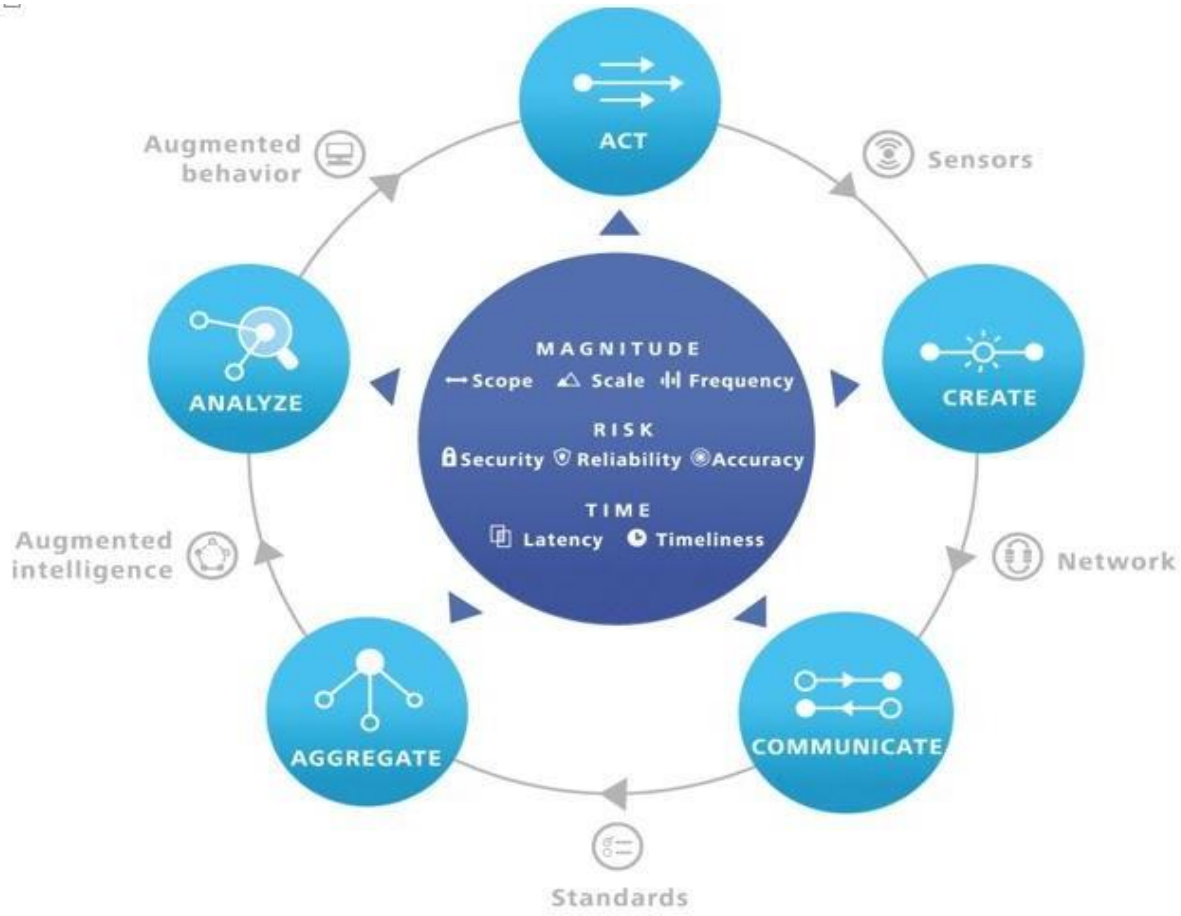
The original business value of IoT cloud lies on its actual process of dissemination with things/devices to the extraction of information for some specific purposes. Value creation primarily depends on the IoT cloud stack which will be discussed in next section. Following figure presents the logical schema of such value creation. It illustrates that IoT cloud based solutions typically combine physical things/devices with information technology in the form of available hardware and software. IoT stack plays the crucial role in resolving these issues. IoT cloud stack is placed on top of IoT stack which results in thing-based functionalities either physical or local influence. Intelligent and assisted IoT cloud services empower user/customer/corporate farms to solicit in decision making, revenue generation, and inductive cum incumbent processes.

The stages of information value creation

Stage	Definition
Create	The use of sensors to generate information about a physical event or state

Communicate	The transmission of information from one place to another
Aggregate	The gathering together of information created at different times or from different sources
Analyze	The discernment of patterns or relationships among phenomena that leads to descriptions, predictions, or prescriptions for action
Act	Initiating, maintaining, or changing a physical event or state

We capture the stages (that is, *Create*, *Communicate*, *Aggregate*, *Analyze*, *Act*) through which information passes in order to create value with the Information Value Loop, shown in figure



Information value drivers

Value driver	Definition
Magnitude	Factors that determine the amount of information that informs action
Scale	Corresponding to ‑volume,‖ this is the number of instances of the same action that inform subsequent action. One can dispatch trucks knowing the location of one truck in a fleet or knowing the locations of all the trucks in a fleet.
Scope	Corresponding to ‑variety,‖ this is the number of different dimensions of an action on which information informs subsequent action. One can dispatch trucks knowing the

	location of a truck, or knowing that truck‘s location, speed, and direction.
Frequency	Corresponding to ―velocity,‖ this is the interval between opportunities to adapt action based on new information. One can update truck dispatches knowing the truck locations once per hour, or knowing them once per minute.
Risk	Factors that determine the probability that information will create value in the manner expected
Security	Is the information used only by those with the necessary authorization? If thieves also know the location of one‘s trucks, the information may well lead to a net reduction in value due to higher rates of theft.
Reliability	Is the information consistently generated as expected? If the other value drivers of information are unpredictable, it is more difficult to make optimal use of that information.
Accuracy	Does the information capture the actual value of what it represents? If the information on the location of the truck misrepresents the truck‘s actual location, dispatch instructions based on that information will be less valuable.
Time	Factors that determine how quickly value can be created from information
Timeliness	Is the information available for use at the most opportune moments? Dispatch schedules that are updated as the trucks reach their routes‘ halfway point are more valuable than those updated after the trucks have returned to the depot.
Latency	Does the information capture the state of the world as it is, or as it was? Knowing trucks‘ locations 30 minutes ago is less valuable than knowing their locations 30 seconds ago.

IoT for agriculture

What Is Smart Agriculture? The Definition and Market Size

Smart agriculture, on the other hand, is mostly used to denote the application of IoT solutions in agriculture.

Although smart agriculture IoT, as well as industrial IoT, aren‘t as popular as consumer connected devices, the market is still very dynamic. The adoption of IoT solutions for agriculture is constantly growing. The market is still developing, there are still ample opportunities for businesses willing to join in. Building IoT products for agriculture within the coming years can set you apart as an early adopter and, as such, help you pave the way to success.

Why should you consider building an IoT application for agriculture in the first place?

The Benefits of Smart Farming: How IoT Is Shaping Agriculture

Technologies and IoT have the potential to transform agriculture in many aspects. Namely, there are five ways IoT can improve agriculture:

- **Data, tons of data, collected by smart agriculture sensors,** e.g. weather conditions, soil quality, crop‘s growth progress or cattle health. This data can be used to track the state of your business in general, as well as staff performance, equipment efficiency, etc.
- **Better control over the internal processes and, as a result, lower production risks**. The ability to foresee the output of your production allows you to plan for better product

distribution. If you know exactly how much crops you are going to harvest, you can make sure your product won‘t lie around unsold.

- **Cost management and waste reduction thanks to the increased control over production**. Being able to see any anomalies in the crop growth or livestock health, you will be able to mitigate the risks of losing your yield.
- **Increased business efficiency through process automation**. By using smart devices, you can automate multiple processes across your production cycle, e.g. irrigation, fertilizing, or pest control.
- **Enhanced product quality and volumes**. Achieve better control over the production process and maintain higher standards of crop quality and growth capacity through automation.
- **Cleaner process.** The same is relevant to pesticides and fertilizers. Not only do IoT-based systems for precision farming help producers save water and energy and, thus, make farming greener, but also significantly scale down on the use of pesticides and fertilizer. This approach allows getting a cleaner and more organic final product compared to traditional agricultural methods.
- **Agility.** One of the benefits of using IoT in agriculture is the increased agility of the processes. Thanks to real-time monitoring and prediction systems, farmers can quickly respond to any significant change in weather, humidity, air quality as well as the health of each crop or soil in the field. In the conditions of extreme weather changes, new capabilities help agriculture professionals save the crops.

IoT Use Cases in Agriculture

There are many types of IoT sensors and IoT applications that can be used in agriculture:

Monitoring of Climate Conditions

Probably the most popular smart agriculture gadgets are weather stations, combining various smart farming sensors. Located across the field, they collect various data from the environment and send it to the cloud. The provided measurements can be used to map the climate conditions, choose the appropriate crops, and take the required measures to improve their capacity (i.e. precision farming).

Precision Farming

Precision farming is a process or a practice that makes the farming procedure more accurate and controlled for raising livestock and growing of crops. The use of IT and items like sensors, autonomous vehicles, automated hardware, control systems, robotics, etc in this approach are key components.

Precision agriculture in the recent years has become one of the most famous applications of IoT in agricultural sector and a vast number of organizations have started using this technique around the world.

The products and services offered by IoT systems include soil moisture probes, VRI optimization, and virtual optimizer PRO, and so on. VRI (Variable Rate Irrigation) optimization is a process that maximizes the profitability on irrigated crop fields with soil variability, thereby improving yields and increasing water use efficiency.

Agriculture Drones

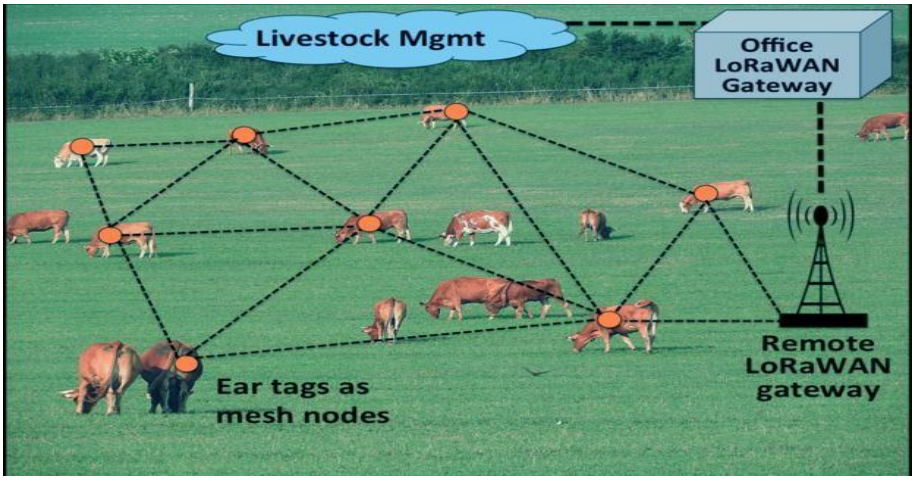
Agricultural drones are a very good example of IoT applications in Agriculture. Agriculture industries today, have become one of the major industries where drones can incorporate. Two types of drones, that is, ground-based and aerial-based drones are being incorporated in agriculture in many ways such as, for crop health assessment, irrigation, planting, and soil & field analysis.

The benefits that the usage of drones brings to the table include, ease of use, time-saving, crop health imaging, integrated GIS mapping, and the ability to increase yields. The drone technology will give a high-tech makeover to the agriculture industry by making use of strategy and planning based on real-time data collection and processing.

The farmers through drones can enter the details of what field they want to survey. Select an altitude or ground resolution from which they what data of the fields. From the data collected by the drone, useful insights can be drawn on various factors such as plant counting and yield prediction, plant health indices, plant height measurement, canopy cover mapping, nitrogen content in wheat, drainage mapping, and so on. The drone collects data and images that are thermal, multispectral and visual during the flight and then lands at the same location it took off initially.

Livestock Monitoring

IoT applications help farmers to collect data regarding the location, well-being, and health of their cattle. This information helps them in identifying the condition of their livestock. Such as, finding animals that are sick so, that they can separate from the herd, preventing the spread of the disease to the entire cattle. The feasibility of ranchers to locate their cattle with the help of IoT based sensors helps in bringing down labor costs by a substantial amount.



One example of an IoT system in use by a company is JMB North America. Which is an organization that provides cow monitoring solutions to cattle producers? Out of the many solutions provided, one of the solutions is to help the cattle owners observe their cows that are pregnant and about to give birth. From them, a battery that is sensor powered is expelled when its water breaks. An information is then sent to the herd manager or the rancher. The sensor thus enables farmers will more focus.

Smart Greenhouses

Greenhouse farming is a technique that enhances the yield of crops, vegetables, fruits etc. Greenhouses control environmental parameters in two ways; either through manual intervention or a proportional control mechanism. However, since manual intervention has disadvantages such as production loss, energy loss, and labor cost, these methods are less effective. A smart greenhouse through IoT embedded systems not only monitors intelligently but also controls the climate. Thereby eliminating any need for human intervention.

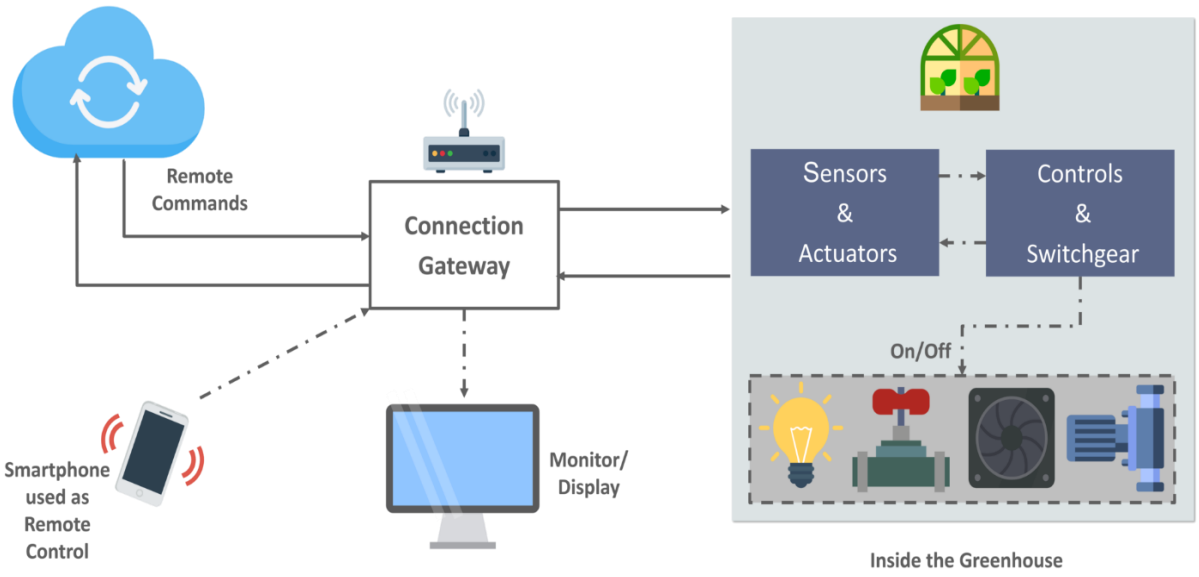
Different sensors that measure the environmental parameters according to the plant requirement are used for controlling the environment in a smart greenhouse. Then, a cloud server create for remotely accessing the system when it connects using IoT.

Inside the greenhouse, the cloud server helps in the processing of data and applies a control action. This design provides optimal and cost-effective solutions to the farmers with minimal and almost no manual intervention.

IoT based Green House

A greenhouse farming technique enhances the yield of crops by *controlling environmental parameters*. However, manual handling results in production loss, energy loss, and labor cost, making the process less effective.

A greenhouse with embedded devices not only makes it easier to be monitored but also, enables us to control the climate inside it. Sensors measure different parameters according to the plant requirement and send it to the cloud. It, then, processes the data and applies a control action.



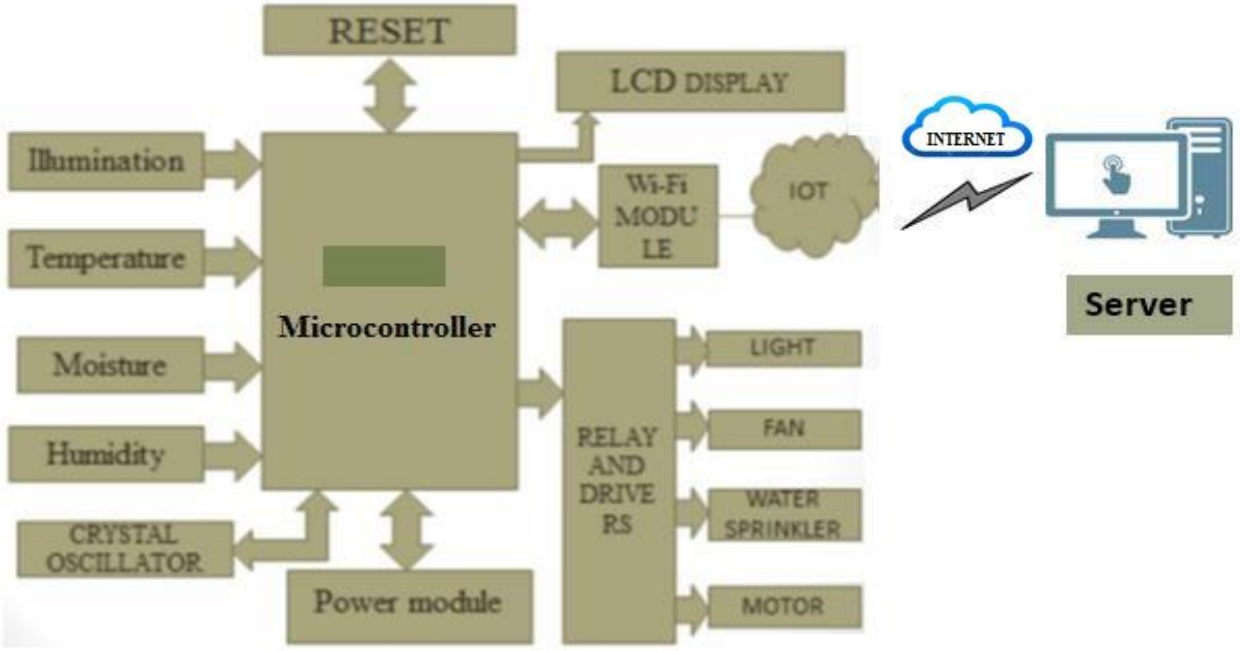


Fig: Block diagram of greenhouse Monitoring system.

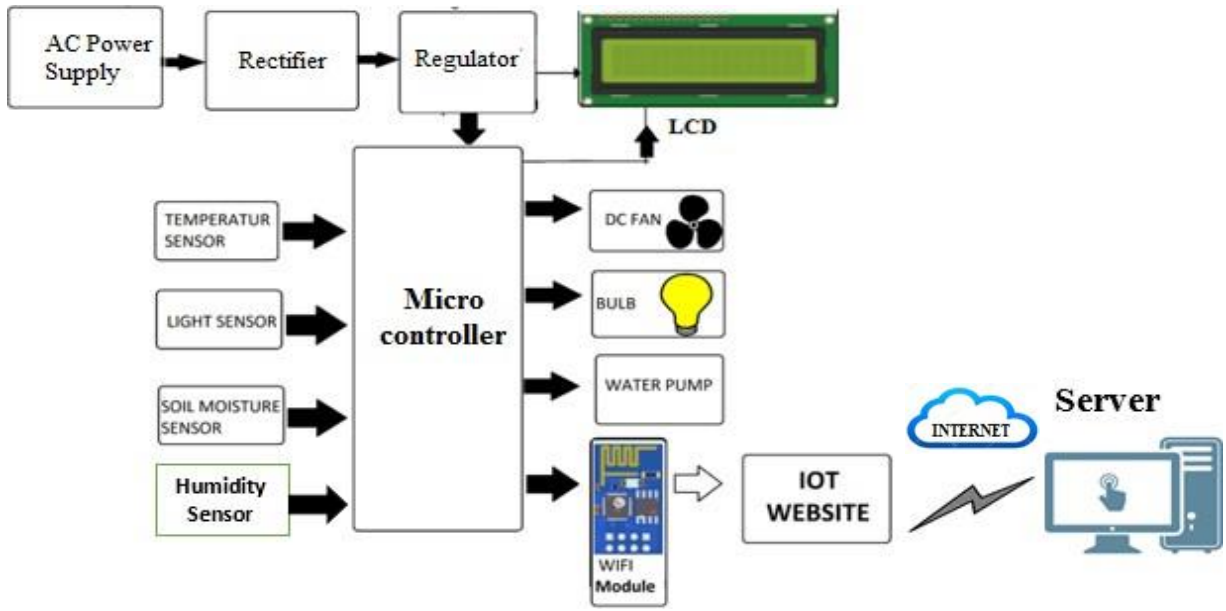


Fig: Block diagram of greenhouse Monitoring system.

A green house is where plants such as flowers and vegetables are grown. Greenhouses warm-up during the day when sun-rays penetrates through it, which heats the plant, soil and structure. Green houses help to protect crops from many diseases, particularly those that are soil borne and splash onto plants in the rain. Greenhouse effect is a natural phenomenon and beneficial to human being. Numerous farmers fail to get good profits from the greenhouse crops for the reason that they can‘t manage two essential factors, which determines plant growth as well as productivity. Green house

temperature should not go below a certain degree, High humidity can result to crop transpiration, condensation of water vapor on various greenhouse surfaces, and water evaporation from the humid soil. To overcome such challenges, this greenhouse monitoring and control system comes to rescue.

This project demonstrates the design and implementation of a various sensors for greenhouse environment monitoring and controlling. This greenhouse control system is powered by microcontroller it consists of temperature sensor, light sensor, soil moisture sensor, LDR sensor, LCD display module, 12v DC fan, Bulb and pump. Temperature sensor, senses the level of temperature, if it goes high DC fans gets on and when the temperature goes low the fan gets off. Soil moisture sensor, senses the water level as the level decreases the pumps gets on. In the absence of light, the LDR sensor senses and the bulb starts glowing. By this way it will become easy to monitor and control the system.

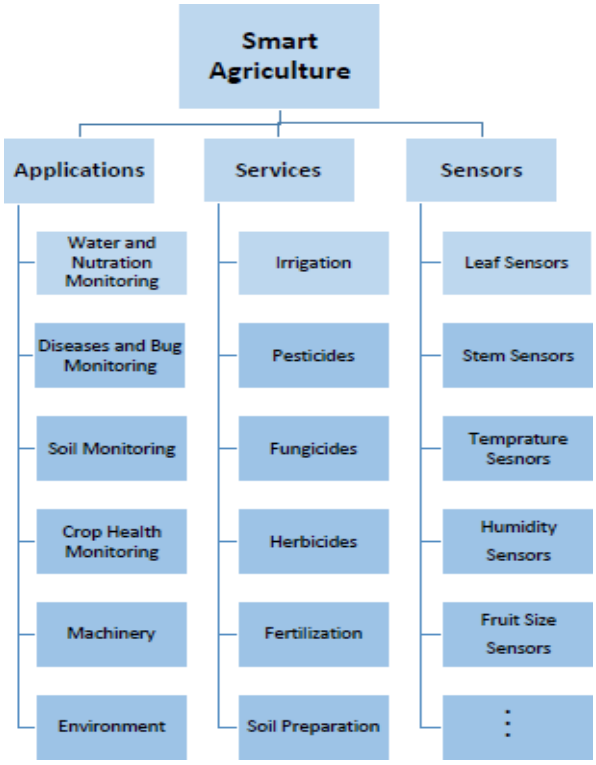
The proposed design grabs data (such as temperature, humidity and moisture) from environments (such as greenhouse) and sends to remote server via modem by using internet connection and also could operate remote commands. Data on the remote server could be viewed via both web and smart phone [19]. Also, these stations could be distance management centers. Throughputs and measurement times should be recorded on a database. Here we are using two WIFI modules, one is to upload the records to the cloud whereas the other one is dedicated for controlling purpose. Besides, remote server should keep track of records to check values whether they are in limits. When a value overflows outside of the boundaries, like if temperature increases dramatically, remote server should send alert to subscriber‘s smart phone and turn on the cooling fan automatically. When user receives alert, he could also step in and prevent emergency situations manually [13]. The controlling webserver consist of mode selection buttons, sensed values and GPIO access. When the device is switched to automatic mode, the system will work as an intelligent system. That means, if the parameter value is exceeds the limit it will automatically trigger an event. When the manual mode is selected, we need to turn on and off each device individually.

Smart Agriculture Monitoring System using IoT

What is Smart Agriculture Monitoring System?

Smart agriculture monitoring system or simply smart farming is an emerging technology concept where data from several agricultural fields ranging from small to large scale and its surrounding are collected using smart electronic sensors. The collected data are analyzed by experts and local farmers to draw short term and long term conclusion on weather pattern, soil fertility, current quality of crops, amount of water that will be required for next week to a month etc.

We can take smart farming a step further by automating several parts of farming, for example smart irrigation and water management. We can apply predictive algorithms on microcontrollers or SoC to calculate the amount of water that will be required today for a particular agriculture field. Say, if there was rain yesterday and the quantity of water required today is going to be less. Similarly if humidity was high the evaporation of water at upper ground level is going to be less, so water Using IoT we can directly send the collected data to a central server in real time. Since we have automated the date collection, the data integrity is assured and since the data processing is done using computers, experts may get advanced analytical software tools to draw most accurate predictions.



Block Diagram of Smart Farming Agriculture

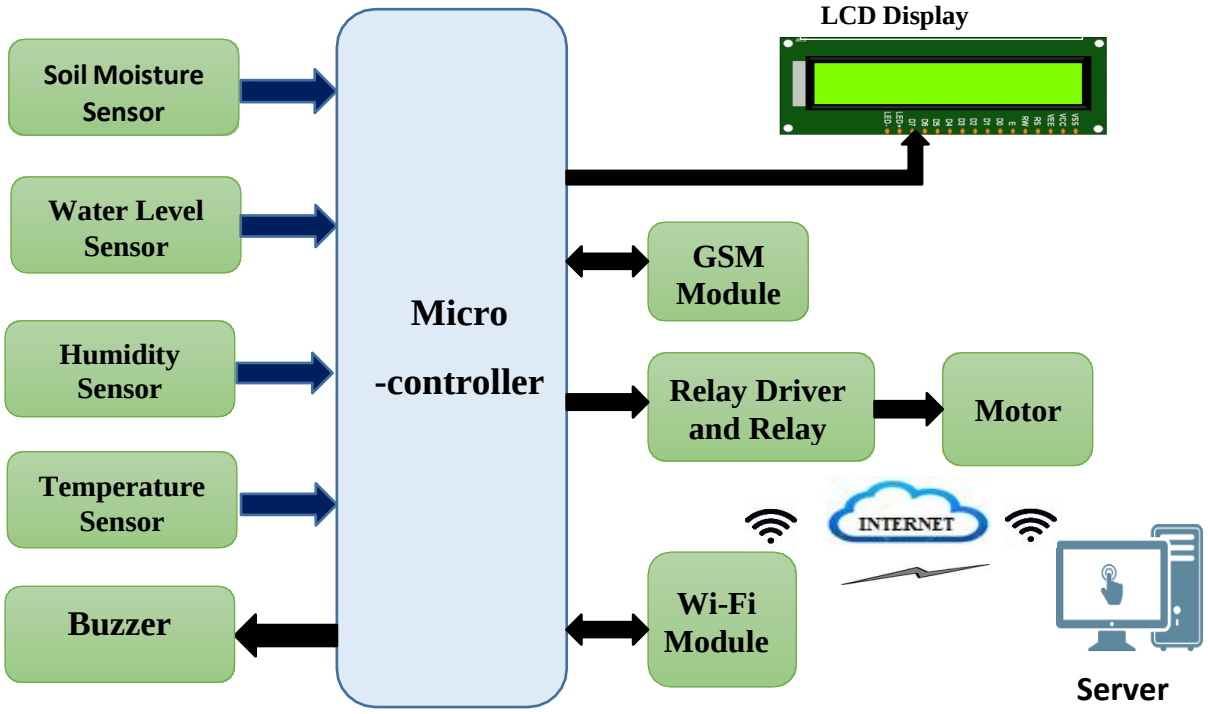


Figure: Block Diagram of Smart Farming Agriculture

Description of the Smart Agriculture

The world population is increasing at a daunting pace. Providing the basic necessities of life for such a huge population is a great challenge. The most basic requirement for any human being is good nutrition. However, due to the increasing population, the old and traditional farming methods are proving insufficient for providing food in bulk quantities. Fortunately, by making use of the latest agricultural techniques as well as smart electronics technology we can increase efficiency and productivity too far higher levels which can ensure our food security. For the purpose of increasing the efficiency and productivity of agricultural crops, an IOT based smart agriculture monitoring project using Arduino is proposed. The system consists of three sensors for the measurement of various parameters crucial for the crop. These sensors include a temperature sensor, a water level sensor, and a soil moisture sensor.

The system also contains a GSM modem and a WiFi module. The project also includes output devices such as a dc motor, relays and a buzzer. There are three sensors present in the circuit which are used for measuring the ambient temperature, water level of the crop and the soil moisture level. Based on the monitoring of these sensor values the smart agriculture monitoring system provides air and water to the crop. The data from the sensors is sent to an Arduino controller which stores and processes this data and then sends it to the IOT platform as well as GSM module. A WiFi module is interfaced with the Arduino which sends the sensor values to the remote IOT platform using WiFi connection. The GSM modem receives the sensor values from the Arduino board and sends these values to the user via SMS after every 5 minutes. Some output devices are also connected to the Arduino outputs. These devices include DC motor, relays, and buzzer. If any of the sensor values crosses a certain predefined threshold then the buzzer is turned on to notify the user. One relay is connected to the fan while the other is connected to the water pump. If the ambient temperature is too hot then the fan is turned on by the controller in order to maintain the desired temperature for the crop. If the soil moisture level is low then the water pump is turned on by the Arduino controller to provide water to the crops. The crop status can be monitored remotely by means of a remote IOT platform.

Description of the block diagram

The project block diagram consists of the following sub-blocks and devices:

The **temperature sensor** measures the ambient temperature of the crop.

The **water level sensor** measures the water level of the crop.

Soil moisture sensor measures the level of moisture in the soil.

GSM modem is used to send SMS notifications to the user at an interval of 5 minutes.

The WiFi module is used to send the sensor values to the remote server via a WiFi connection and IOT protocols.

Microcontroller communicates with the GSM modem and WiFi module, gathers data from the sensors and activates the output devices.

Relays are used to turn on the fan as well as the water pump in order to maintain the temperature and moisture level of the crop.

The buzzer is turned on when any of the sensor values crosses a certain threshold

Air Pollution Monitoring using IoT

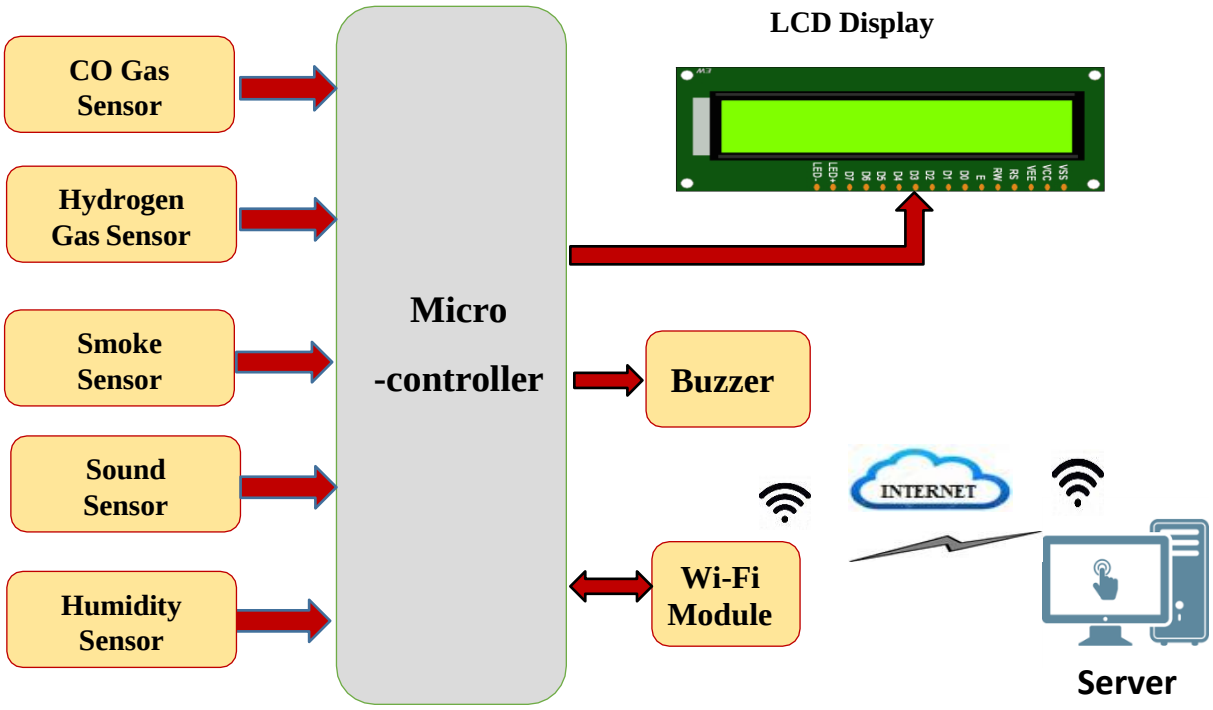


Fig.: Air Pollution Monitoring using IoT

Description of the System

With an ever-increasing number of industrial units and transport vehicles, the problem of air and sound pollution is becoming severe by each passing day. Therefore, it has now become extremely important to keep the air and noise pollution levels under check, especially in urban centers. To achieve this goal, some type of air and sound pollution monitoring system needs to be developed.

To counter the problem of air and sound pollution monitoring, we propose a microcontroller and IOT based air and sound pollution monitoring system project. To measure the air and noise pollution levels, the monitoring system employs a CO2 sensor, hydrogen sensor, smoke sensor and humidity sensor. The measured sensor data is then transmitted to a remote web server via a Wi-Fi connection. The most significant aspect of this microcontroller based Air pollution monitoring system is the IoT connectivity, which allows the remote and real-time monitoring of air and sound pollution levels. By effective monitoring of pollution levels, measures can be taken to bring those levels down.

Operation of the System

The IOT based air and sound pollution monitoring system using microcontroller consists of sensors, controller, output device and Wi-Fi communication system. The system contains five different sensors. The CO2 sensor measures the carbon emission levels in the air. The methane sensor measures the level of methane gas in the air which is exhausted from the industrial units and vehicles. The microphone measures the level of sound pollution in the atmosphere. The smoke

sensor used for sensing smoke in air and humidity sensor used to measure humidity. All the data collected from these sensors is continuously fed to a controller which is a microcontroller board in our case. The microcontroller converts this sensor data into a convenient form and transmits to a remote web server by making use of IOT – the internet of things communication protocols.

Apart from that, the sensor data is also displayed on an LCD display in real-time by the microcontroller. On the IOT web server-side, an optional graphical user interface can be created, which visualizes the sensor data in a convenient way. By virtue of this IOT based pollution monitoring system project, air and pollution levels can be constantly monitored from a remote location, may it be anywhere in the world and steps can be taken in order to reduce the pollution levels.

Description of the block diagram

The block diagram of the microcontroller and IOT based air and noise pollution monitoring system. The system consists of five snesors. All five sensor blocks send their data to the microcontroller. Microcontroller constitutes the main block in our system block diagram. It is the most heart of the IOT air and sound pollution monitoring system. The sensors devices that send their data into the microcontroller. The output of the microcontroller is connected to two output blocks. These output blocks are the IOT web server block and the LCD display screen block.

The data from the sensors is processed by the microcontroller and is then displayed on the LCD output device. The same air pollution sensor data which is displayed on the LCD is also communicated to the remote web server by means of onboard Wi-Fi of the microcontroller board. This sensor data can then be monitored from a remote location using the IOT website.

Applications, advantages, and enhancements of the project

- The IOT based air and sound pollution monitoring system using microcontroller can be used as a subsystem for smart cities. This microcontroller based pollution monitoring system can be used to monitor the real-time air and sound pollution levels in any area.
- This raspberry pi air and noise pollution monitoring system over IOT can be implemented using low cost components and can be built at a reasonable price.
- This monitoring system is completely automated and requires very little human intervention and maintenance.
- This monitoring system can be adopted by the government and can be helpful in devising environment friendly policies.
- The size of the monitoring system can be made more compact by using other controllers in place of the microcontroller board.
- More environmental sensors can be added into the system in order to get more precise data about the air quality and ambient sound.

Environmental Monitoring using IoT:

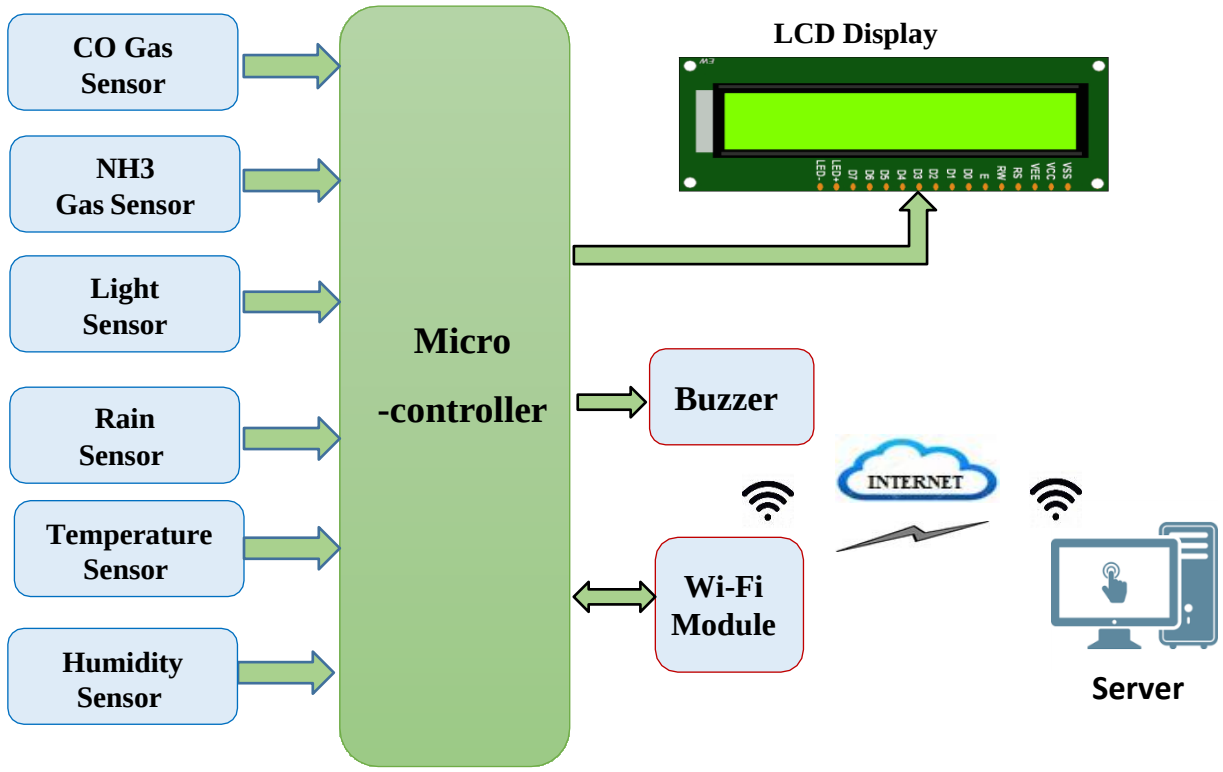


Figure: Block diagram of Environmental Monitoring System using IoT

The primary focus is to design an IoT based monitoring system which keeps track of environmental parameters such as temperature, humidity, air pollution, sun light intensity and rain. The system works in real time and captures the data for analysis.

The only way to get the real sense of the various environmental parameters is through the use of the sensors. The design of the sensor network is crucial part of the system as the end user would rely on data provided by these sensors only. Correctly designed and accurately calibrated sensor network will serve the purpose. The system consists of different sensors like CO, NH3, light, humidity, rain and temperature. The output of sensors is given to microcontroller which convert it in to digital and process theses data. The values of sensors are displayed on LCD as well as sent to Wi-Fi module and to internet server. From these user can monitor the different values of sensor which are related to environment.

Block Diagram Description:

The block diagram consists of six sensors CO, NH3, light, humidity, rain and temperature. The function of sensor is to convert physical parameters into electrical signal. The heart of the system is a microcontroller, which reads analog value from sensors and convert it into digital and performs processing on this digital data. The value of the sensors are displayed on LCD display.

The same value of sensor are given to Wi-Fi module and hence data is sent over the internet. Then user can monitor the environmental condition from anywhere.

IoT for Health Care (Smart Health Care System using IoT)

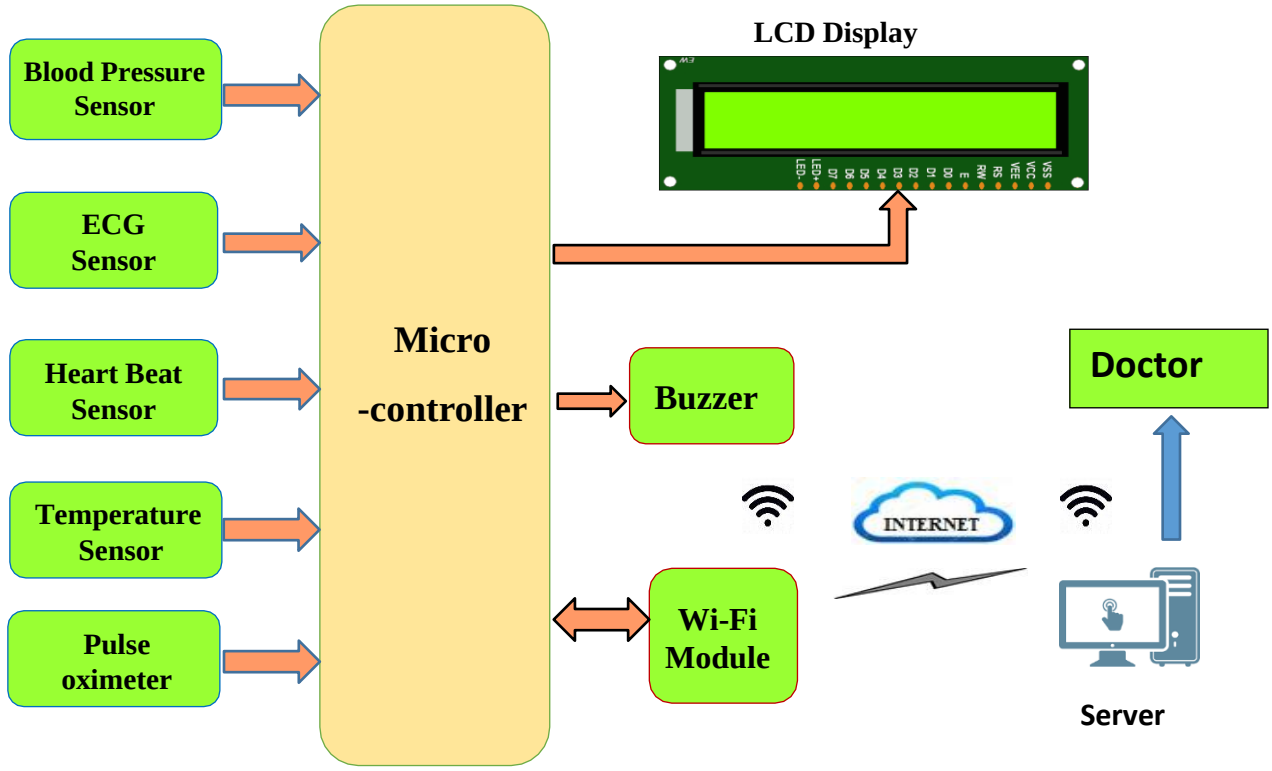


Fig.: Block diagram of IoT for Health Care (Smart Health Care System using IoT)

Health is one of the global challenges for humanity. In the last decade the healthcare has drawn considerable amount of attention. The prime goal was to develop a reliable patient monitoring system so that the healthcare professionals can monitor the patients, who are either hospitalized or executing their normal daily life activities. Recently, the patient monitoring systems is one of the major advancements because of its improved technology.

Currently, there is need for a modernized approach. In the traditional approach the healthcare professionals play the major role. They need to visit the patient‘s ward for necessary diagnosis and advising. There are two basic problems associated with this approach. Firstly, the healthcare professionals must be present on site of the patient all the time and secondly, the patient remains admitted in a hospital, bedside biomedical instruments, for a period of time. In order to solve these two problems, the patients are given knowledge and information about disease diagnosis and prevention. Secondly, a reliable and readily available patient monitoring system (PMS) is required. In order to improve the above condition, we can make use of technology in a smarter way. In recent years, health care sensors along with microcontroller play a vital role. Wearable sensors are in contact with the human body and monitor his or her physiological parameters. We can buy

variety of sensors in the market today such as ECG sensors, temperature sensors, pulse monitors etc. The cost of the sensors varies according to their size, flexibility and accuracy.

In this system we are measuring patient‘s parameters (ECG, temperature, heart rate, pulse, etc) with different available sensors. These sensors collected data i.e. biometric information is given to microcontroller and then it is transferred to server.

Block Diagram Description:

The block diagram consists of five sensors heart bit, pulse Oximeter, blood pressure, ECG and temperature. The function of sensor is to convert physical parameters into electrical signal. The heart of the system is a microcontroller, which reads analog value from sensors and convert it into digital and performs processing on this digital data. The value of the sensors are displayed on LCD display. The same value of sensor are given to Wi-Fi module and hence data is sent over the internet. Then doctor of concern patient or any doctor can monitor the environmental condition from anywhere. Hence we can have a smart health monitoring using IoT.

IoT for Home Automation (Smart Home Automation by using IoT)

In the competitive environment and fast world, home automation technology is required for every person. IoT conceptualizes the idea of remotely connecting and monitoring real world objects through the internet. When it comes to our home, this concept can be aptly incorporate to make it smarter, safer and comfortable life. This IoT home automation system focuses on building a smart wireless home automation using android application over internet. The system used to create a virtual, but practically usable, android home automation system by control and monitoring the status (ON/OFF of the home appliances) using multiple ways such as the internet, electrical switch etc. The android mobile is used to send the commands to the Arduino to control all the home appliances. The advantage of this technology is low cost, user friendly, easy installation and also wastage of electrical power is reduced by proper monitoring of home appliances.

–Smart‖ word is becoming very popular these days. We have smart phones, smart TVs, smart homes and so on. The main idea behind the –smart‖ devices is that they can operate to some extent interactively and autonomously thereby helping people in managing the appliances freely and smartly. Based on this ideas, a smart home automated system is designed by implementing related software and hardware‘s. The system is build using Internet of Things based smart home automated system to remotely control the home appliances using Wi-Fi. A low cost Wi-Fi module is used to build Smart Units. The user can remotely operate home appliances like lights, fan, door lock etc. through internet. The lights in any room can be controlled from any place in the house, within the wifi range, through a Android Smart phone

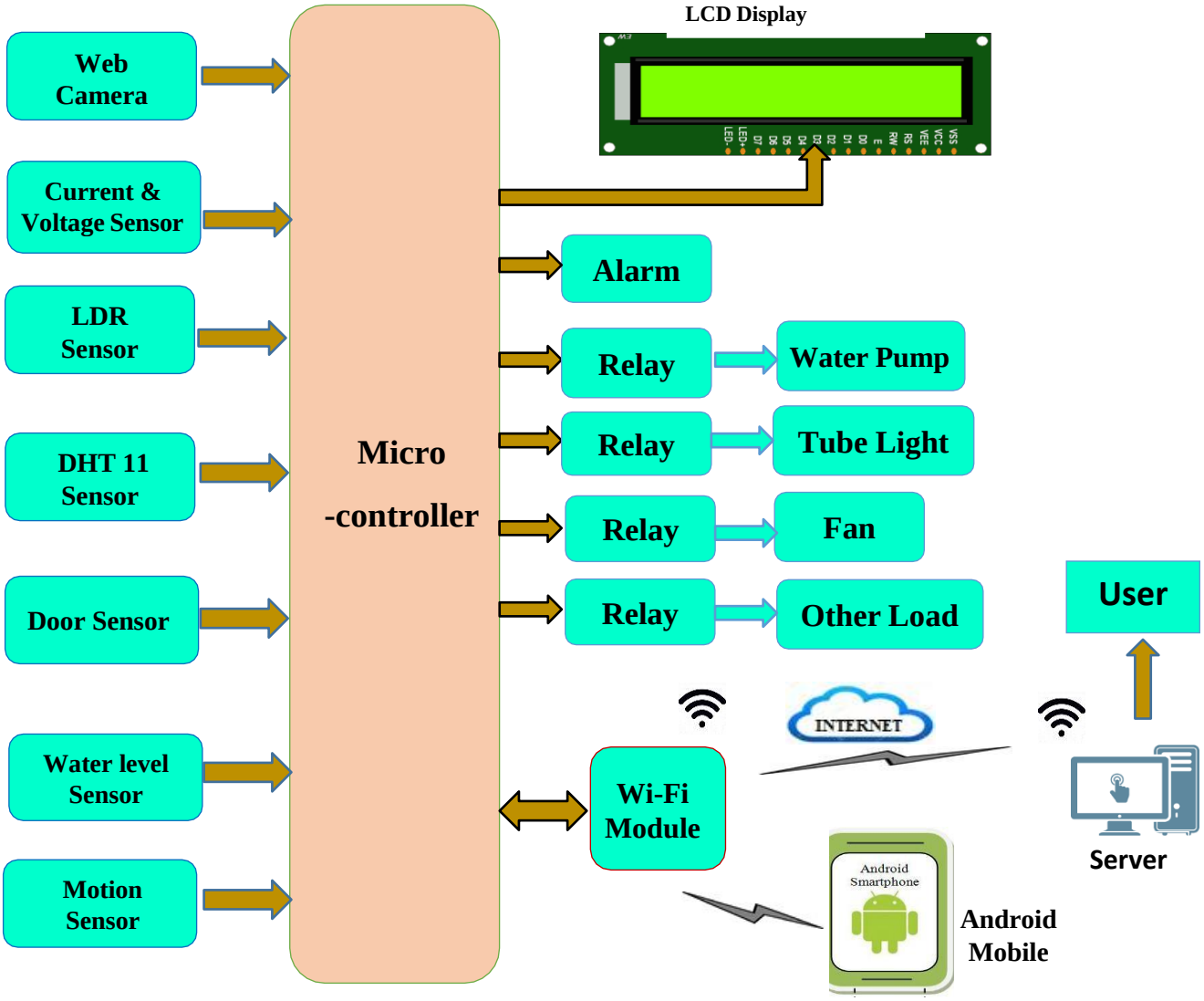


Figure: Smart Home Automation using IoT

Home automation has become more and more popular in recent years. It aims at helping people manage the home appliances freely and build an autonomous environment in home. The aim of this system is the home automation with full security and controlling the home appliances using wireless communication as Wi-Fi. The system is smart home system with the implementation of related software and hardware. To assure security the motion sensor (PIR) is used to detect the motion and vibration to prevent from theft. It alerts the people by alarm/buzzer and starts to record it through camera. The temperature and humidity of the each room is monitored and maintained at room temperature using temperature and humidity sensors which activates the exhaust fan to maintain the temperature. The water level sensor is used to fill the overhead water tank without wasting the water. For these control purposes the Arduino mega 2560 and ESP8266 can be used because the arduino has the advantages of ease understandability and easily modifiable. The arduino board is specially designed circuit board for programming and prototyping with ATMEL microcontroller.

Android Smart phone is used to sense the status of appliances and also for controlling these appliances. The Android app has been designed and generated making use of the open source web application software to monitor and control the domestic appliances using any Android Smart phone. The Android app created provides graphical user interface supports two options to control appliances: (i) the user can use icons or graphical buttons created for respective home appliances and (ii) they can use specific voice commands to control those home appliances using Android Smart phone.

Industrial Automation using IoT (Smart Industrial Automation using IoT)

What is the Industrial IoT, or IIoT?

The Industrial Internet of Things is the use of connected smart devices in industrial applications for purposes such as automation, remote monitoring and predictive maintenance. The IIoT is a more robust version of the Internet of Things, or IoT which is the realm of connected devices in commercial and consumer applications.

In Industrial IoT use cases, smart devices may be deployed in construction vehicles, supply chain robotics, solar and wind power, agricultural sensor systems, smart irrigation, and more. These IIoT applications tend to have one thing in common: they are deployed in challenging environments. There are light industry IIoT use cases such as meters, and heavy industry use cases such as conveyors used in the mining industry, where devices can be subject to a wide range of environmental factors, from extreme heat and cold to moisture and vibration. The Industrial Internet of Things is about deploying smart machines to capture and move data, sense changes in temperature, flow or volume, automate procedures for efficiency, accuracy and safety, deliver data into the right hands for analysis and decision making, and ensure that all of those processes happen on time, reliably and securely.

Defining the Industrial Internet of Things

Let‘s begin with a short IIoT definition: The Industrial Internet of Things (IIoT) refers to the vast number of machines and devices – or ‗things‘ – a business uses that are now connected to the Internet. Protecting this critical operational data, secure IIoT forms a closed, private Internet network where the devices can communicate and share data with other people, systems, and things. This data can be used to improve existing work practices and business processes as well as creating entirely new ways of doing things.

When you define the industrial internet of things, it‘s very important that you understand there‘s more than just the industrial IoT devices. IIoT forms an ecosystem where people, applications and devices interact. That‘s why most large industrial IoT solutions are based around a central industrial IoT platform that can manage every aspect of the industrial IoT network and the data flowing through it.

The **Industrial Internet of Things** (**IIoT**) refers to interconnected sensors, instruments, and other devices networked together with computers' industrial applications, including manufacturing and energy management. This connectivity allows for data collection, exchange, and analysis, potentially facilitating improvements in productivity and efficiency as well as other economic benefits

IIoT versus IoT

Although the internet of things and the industrial internet of things have many technologies in common, including cloud platforms, sensors, connectivity, machine-to-machine communications and data analytics, they are used for different purposes.

- IoT applications connect devices across multiple verticals, including agriculture, healthcare, enterprise, consumer and utilities, as well as government and cities. IoT devices include smart appliances, fitness bands and other applications that generally don't create emergency situations if something goes amiss.
- IIoT applications, on the other hand, connect machines and devices in such industries as oil and gas, utilities and manufacturing. System failures and downtime in IIoT deployments can result in high-risk situations or even life-threatening situations. IIoT applications are also more concerned with improving efficiency and improving health or safety, versus the user-centric nature of IoT applications.

How IIoT works

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

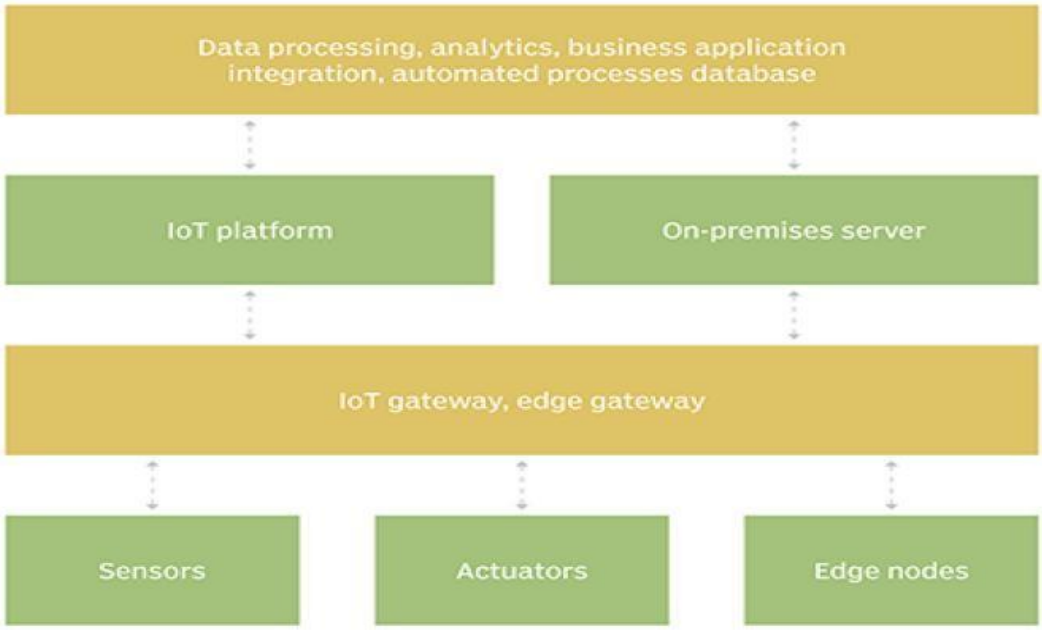
Intelligent assets that can sense, communicate and store information about themselves;

Public and/or private data communications infrastructure

Analytics and applications that generate business information from raw data; and People.

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

IOIT Architecture



Industrial IoT Services

1. Digital/connected factory:

IoT enabled machinery can transmit operational information to the partners like original equipment manufacturers and to field engineers. This will enable operation managers and factory heads to remotely manage the factory units and take advantage of process automation and optimization. Along with this, a digitally connected unit will establish a better line of commands and help identify key result areas (KRAs) for managers.

2. Facility management:

The use of IoT sensors in manufacturing equipment enables condition-based maintenance alerts. There are many critical machine tools that are designed to function within certain temperature and vibration ranges. IoT Sensors can actively monitor machines and send an alert when the equipment deviates from its prescribed parameters. By ensuring the prescribed working environment for machinery, manufacturers can conserve energy, reduce costs, eliminate machine downtime and increase operational efficiency.

3. Production flow monitoring:

IoT in manufacturing can enable the monitoring of production lines starting from the refining process down to the packaging of final products. This complete monitoring of the process in (near) real-time provides scope to recommend adjustments in operations for better management of operational cost. Moreover, the close monitoring highlights lags in production thus eliminating wastes and unnecessary work in progress inventory.

4. Inventory management:

IoT applications permit the monitoring of events across a supply chain. Using these systems, the inventory is tracked and traced globally on a line-item level and the users are notified of any significant deviations from the plans. This provides cross-channel visibility into inventories and managers are provided with realistic estimates of the available material, work in progress and estimated the arrival time of new materials. Ultimately this optimizes supply and reduces shared costs in the value chain.

5. Plant Safety and Security:

IoT combined big data analysis can improve the overall workers‘ safety and security in the plant. By monitoring the Key Performance Indicators (KPIs) of health and safety, like the number of injuries and illness rates, near-misses, short- and long-term absences, vehicle incidents and property damage or loss during daily operations. Thus, effective monitoring ensures better safety. Lagging indicators, if any, can be addressed thus ensuring proper redressal health, safety, and environment (HSE) issues.

6. Quality control:

IoT sensors collect aggregate product data and other third-party syndicated data from various stages of a product cycle. This data relates to the composition of raw materials used, temperature and working environment, wastes, the impact of transportation etc. on the final products. Moreover, if used in the final product, the IoT device can provide data about the customer

sentiments on using the product. All of these inputs can later be analyzed to identify and correct quality issues.

7. Packaging Optimization:

By using IoT sensors in products and/or packaging, manufacturers can gain insights into the usage patterns and handling of product from multiple customers. Smart tracking mechanisms can also trace product deterioration during transit and impact of weather, road and other environment variables on the product. This will offer insights that can be used to re-engineer products and packaging for better performance in both customer experience and cost of packaging.

8. Logistics and Supply Chain Optimization:

The Industrial IoT (IIoT) can provide access to real-time supply chain information by tracking materials, equipment, and products as they move through the supply chain. Effective reporting enables manufacturers to collect and feed delivery information into ERP, PLM and other systems. By connecting plants to suppliers, all the parties concerned with the supply chain can trace interdependencies, material flow and manufacturing cycle times. This data will help manufacturers predict issues, reduces inventory and potentially reduces capital requirements.

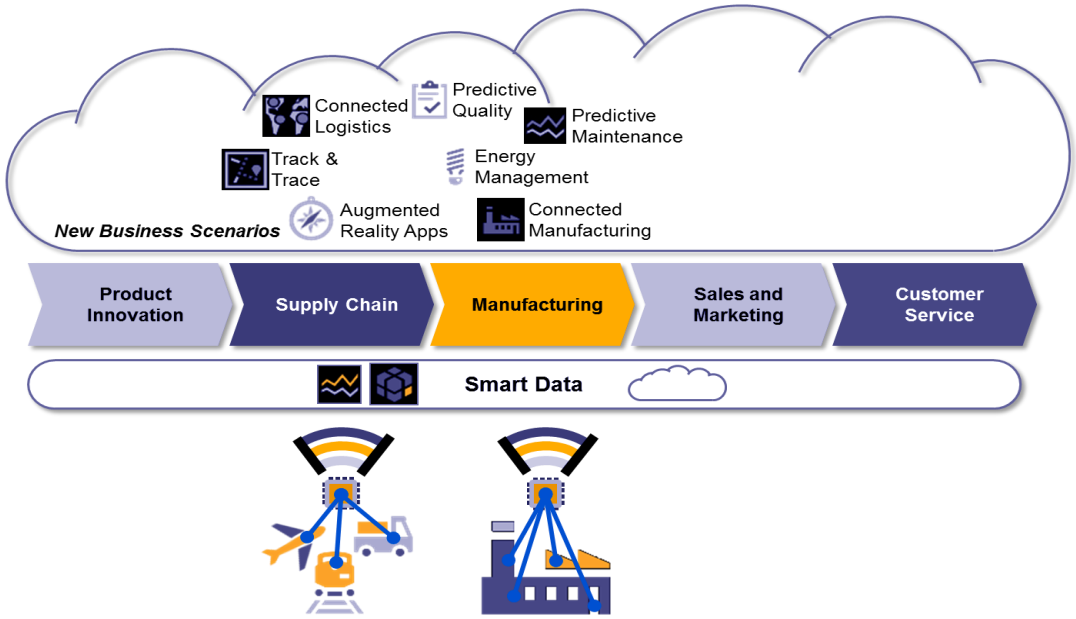
9. Power Management

IoT can offer better solutions for power management in industries. Specific sensors can detect environment and trigger to turn on/o_ control of lights, air conditioners, humidity controls, liquid flow etc… for efficient power management.

Advantages of Industrial Internet of Things

- Improved accuracy
- Product and process optimization
- Predictive maintenance and analysis
- Higher efficiency
- Remote accessibility and monitoring
- Enhanced security
- Scalability of network
- Reduced down time for machines and process
- Power savings
- Cost effectiveness

Industrial IOT Applications- Logistics & Supply Chain Optimization:



Example of Smart Industrial Automation:

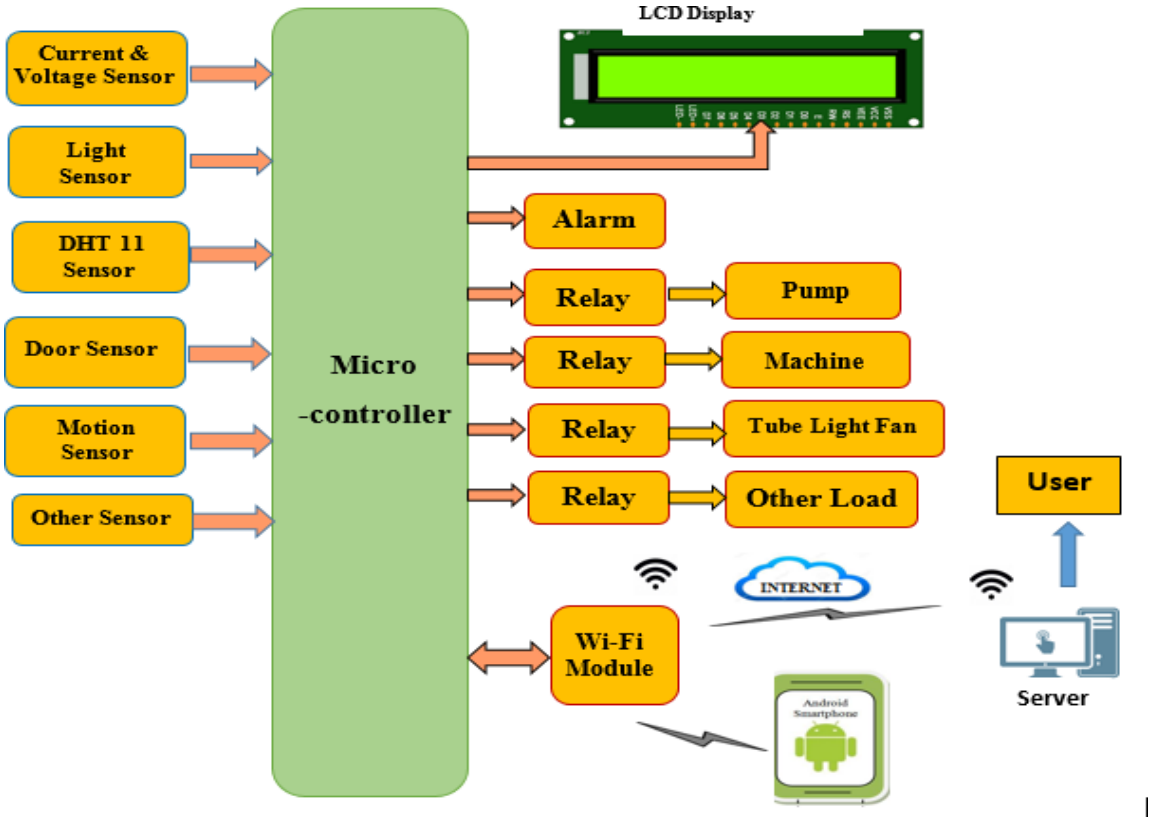


Fig.: Block diagram of Industrial Automation

Smart Energy Meter and Advanced Metering Infrastructure

Smart Energy Meter

Wireless energy management and analysis is an embedded system implemented for smart metering. In country like India, energy management is done mostly with door –door billing system or user have to pay online at only gram panchayat or municipality where authorized person will do payment procedure for user and take the amount. Here prepaid scheme is implemented for the user-end. User will get the electricity equivalent to balance in his account. All the activity done by server implemented over a here and sum illegal activity at the user to end like meter box lock broken can be determined etc

We can see a person standing in front of our house from electricity board, whose duty is to read the energy meter and handover the bills to the owner of that house every month. This is nothing but meter reading. According to that reading we have to pay the bills. The main drawback of this system is that person has to go area by area and he has to read the meter of every house and handover the bills. Many times errors like extra bill amount, or notification from electric board even though the bills are paid are common errors. To overcome this drawback we have come up with an idea which will eliminate the third party between the consumer and service provider, even the errors will be overcome.

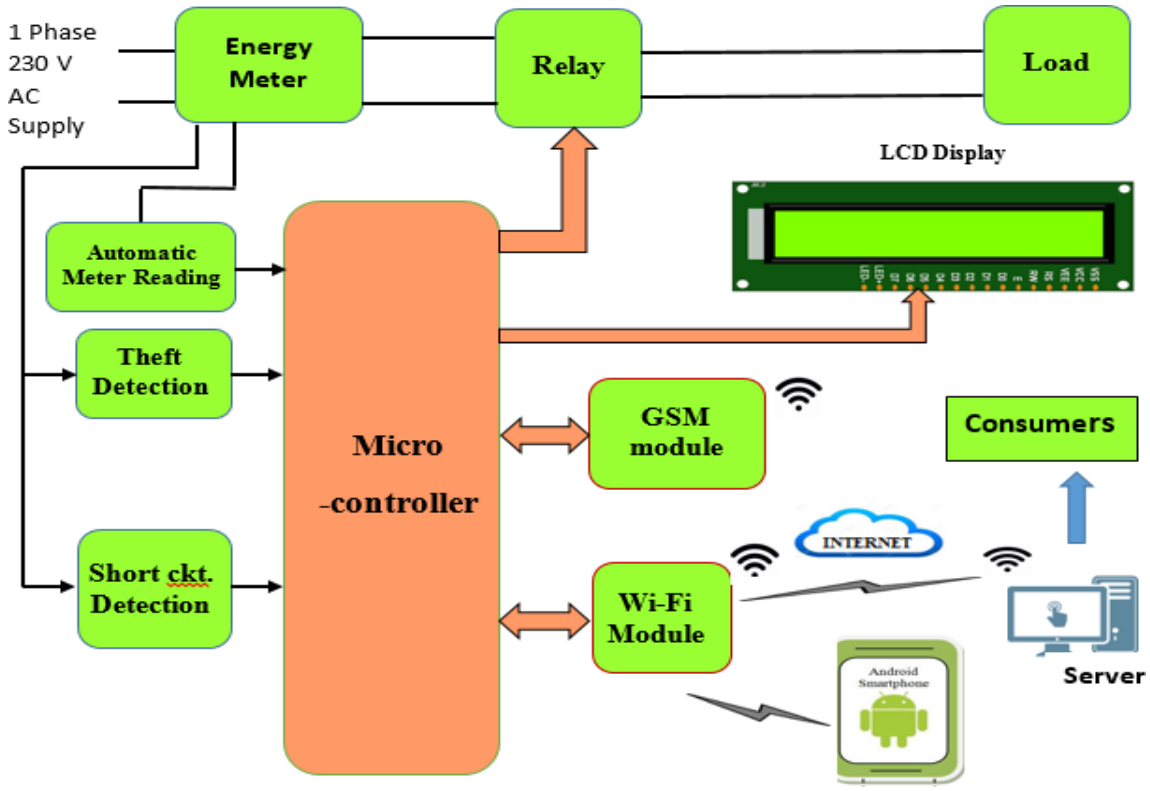


Fig. : Block Diagram of Smart Energy Meter

IOT (Internet of things) is the network of physical things with electronics software, sensors, and connectivity to enable objects to collect and exchange data. IOT based automatic meter reading is the technology of automatic collecting data energy meter and transferring data to the server for billing process and if there is any tempering then also detectable. The internet connected to meter collect the data and display data on the LCD by which we can read and understand the things that are going on the system. Current drawn also calculated by the current transformer that connected in series with the load will be shown on the LCD. This data transfer to sever unit at MSEB. MSEB is not a technical word but it is electricity distribution board –Maharashtra State Electricity Board‖. The data is received by the internet and whenever a key is pressed microcontroller send SMS through the internet to the transmitter to get the reading of the meter. It is difficult to manual reading and calculating bill of individually. This will help for the proper and accurate reading of billing process. By taking all these features that can be done by IOT based smart energy meter easily.

The concept of Internet of Things (IoT) plays an important role in changing the current Internet into well featured internet. The IoT based smart energy meter contains mainly five modules.

1. Microcontroller module
2. Theft detection module
3. Energy Meter module
4. Automated Meter Reading (AMR) module
5. Wi-Fi module

In the smart energy meter, the microcontroller is interfaced with AMR module, Theft detection module and Wi-Fi module. The microcontroller is a core component of the smart energy meter system which is placed at the consumer end for the purpose of measuring the meter reading, theft detection and storing the data. This data is transferred between consumer end and energy supplier end using IoT Wi-Fi module. The AMR module continuously monitors the meter and collects the reading and sends to the microcontroller. In the current scenario, there is a need to uniquely identify the smart meter device remotely in a reliable manner. To achieve the characteristic of device remotely we have provided IP address for each connection. The system have concentrated on the theft detection, optimum utilization of power and convey the energy consumption information to the user end. The block diagram given in above figure illustrates the **Smart energy meter**.

A. Process at Consumer End

At the consumer end, the power supply module provides the entire power needed by the system to function. Also this power supply charges the DC Backup so that when there is no power from the utility company, the DC Backup can energize the system. Microcontroller is used to collect and store the meter reading information from the electricity meter and also performs the control process and sends the required information to energy provider such as number of units consumed using Wi-Fi module. The purpose of LCD module is to get visual information about the number of units consumed, alert messages and connection status. This is a backup power supply unit (DC backup) for the system. The purpose of the DC backup is to makes the system active even there is

no energy supply from the utility company. A small 8.4V, 5600mAh rechargeable battery is used here.

B. Process at Supplier End

At the energy provider end, there is a server computer to receive the meter reading and generate the bill. If any theft is detected the system sends an alert message and disconnect the energy supply. If consumer fails to pay the electricity bill amount within the due time by the supplier, the system can disconnect and reconnect the by sending the appropriate command to the controller.

Advanced Metering Infrastructure

What is Advanced Metering Infrastructure?

AMI (Advanced Metering Infrastructure) is the collective term to describe the whole infrastructure from Smart Meter to two way-communication network to control center equipment and all the applications that enable the gathering and transfer of energy usage information in near real-time. AMI makes two-way communications with customers possible and is the backbone of smart grid. The objectives of AMI can be remote meter reading for error free data, network problem identification, load profiling, energy audit and partial load curtailment in place of load shedding.

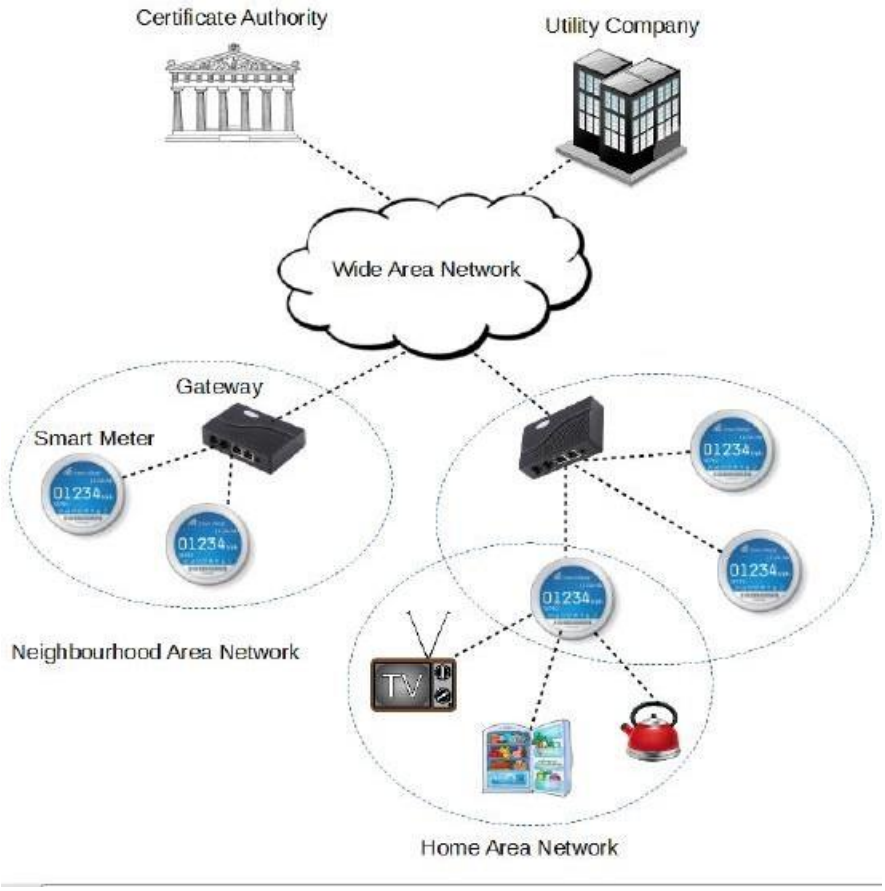


Fig. 1. Advanced metering infrastructure (AMI) network architecture.

Building Blocks of AMI

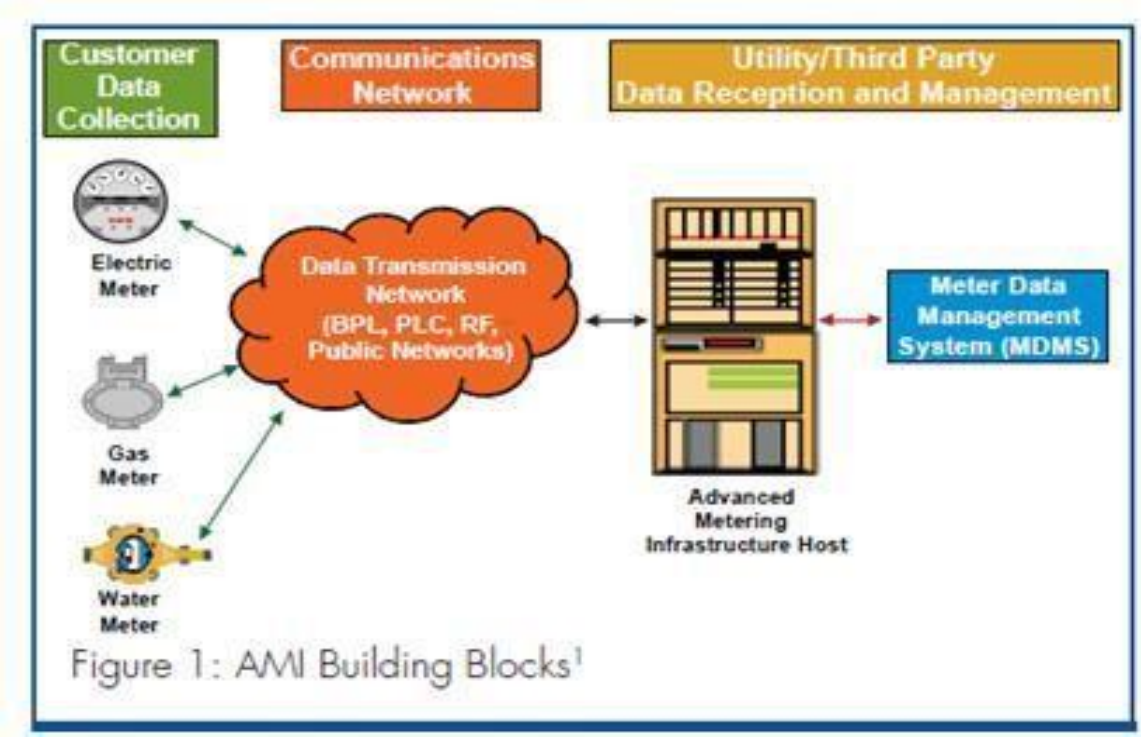


Figure 1: AMI Building Blocks[1]

AMI is comprised of various hardware and software components, all of which play a role in measuring energy consumption and transmitting information about energy, water and gas usage to utility companies and customers. The overarching technological components of AMI include:

- ***Smart Meters-*** Advanced meter devices having the capacity to collect information about energy, water, and gas usage at various intervals and transmitting the data through fixed communication networks to utility, as well as receiving information like pricing signals from utility and conveying it to consumer.
- ***Communication Network:*** Advanced communication networks which supports two way communication enables information from smart meters to utility companies and vice-versa. Networks such as Broadband over Power Line (BPL), Power Line Communications, Fiber Optic Communication, Fixed Radio Frequency or public networks (e.g., landline, cellular, paging) are used for such purposes.
- ***Meter Data Acquisition System-*** Software applications on the Control Centre hardware and the DCUs (Data Concentrator Units) used to acquire data from meters via communication network and send it to the MDMS
- ***Meter Data Management System (MDMS):*** Host system which receives, stores and analyzes the metering information.

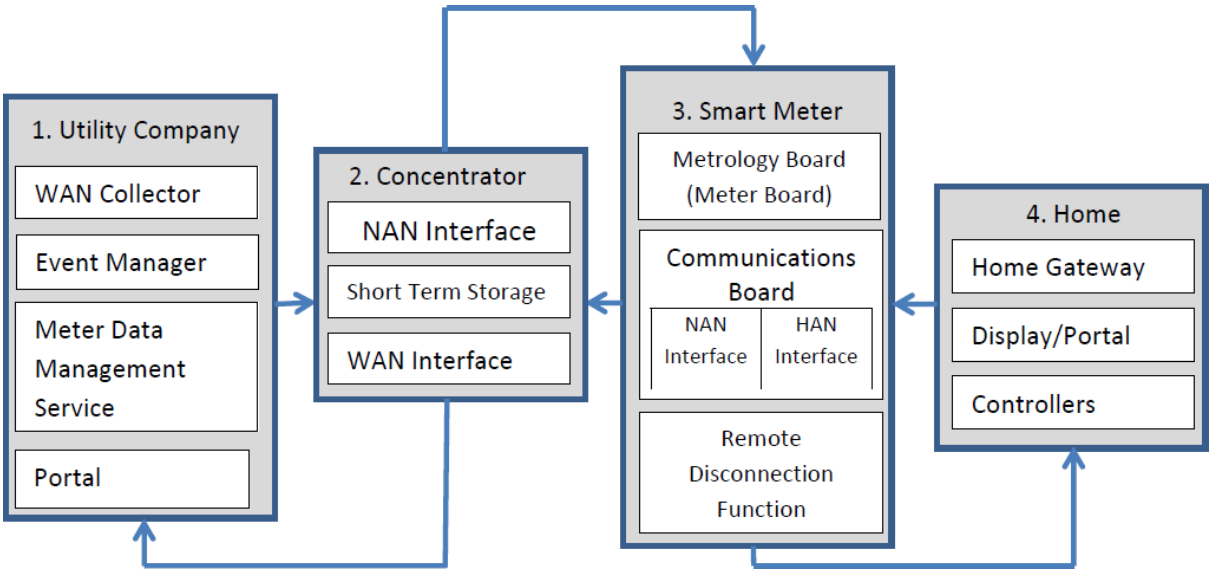


Figure: The Advanced Metering Infrastructure (AMI) consisting of four main logical components: 1 Utility Company, 2. Concentrator, 3. Smart Meter, and 4. Home

An abstraction of an Advanced Metering Infrastructure (AMI) could enumerate four main components: the utility company (utility, for short), the data collector or concentrator often located in the neighborhood, the smart meter, and the home or office as shown in above Figure. Other AMI topologies are possible, resulting from merging or duplicating these logical components. The AMI or the intelligent AMI –smart meter‖ is a combination of equipment, communications and processes for utilities to enhance their internal operations and customer service. Beginning with the smart meter, above Figure box 3 includes the smart meter board used to measure energy consumption and interface cards to communicate with the home area network (HAN) and the concentrator.

The meter may also contain a disconnection function that allows one, if enabled, to remotely connect or disconnect the service. Smart meters form a collection of communicating devices within the same approximate geographic location that support the transportation of data from the individual meters to the WAN collector. This two-way wireless or wired communications network is labeled the Neighborhood Area Network (NAN).The home, box 4, provides the consumer access points to control and monitor consumption through remote control interfaces. It contains the home gateway to communicate with the meter which also supplies a path for the home display and controllers to the energy consuming devices. The first two boxes (1 and 2) in Figure 1 introduce additional capabilities over previous metering systems and are the most important to secure. The concentrator normally has two card interfaces, one to the meter (NAN interface) and one to the central office (WAN Interface). The concentrator may possess additional message processing capabilities and, depending on its purpose, may have a small amount of buffer memory to use between the two communication cards or larger amounts of memory to serve as a storage location.

The central office connects to the concentrator through the WAN interface, interfaces with the event manager to process alarms and alerts. The Meter Data Management Service module in the utility is the processing unit for billing and system operations. Other modules, such as portals allowing customers to view meter data and bills also can be present.

WAN Collector Event Manager Meter Data Management Service Portal 1. Utility Company 2. Concentrator 3. Smart Meter 4. Home Short Term Storage WAN Interface Metrology Board (Meter Board) Communications Board NAN Interface HAN Interface Remote Disconnection Function Home Gateway Display/Portal Controllers NAN Interface

Smart Metering, or an AMI, relies on a complex communications infrastructure to exchange information with and control digital meters and other devices that make up the AMI. This communications infrastructure can consist of various communications networks and requires frequent transmittal of measurements and command and control data. Typically, there are three or more networks, namely the HAN, the NAN, and the WAN. As it can be seen by the arrows in above Figure, there are six communication paths in the AMI. These paths can vary from simple low speed power line carriers to very high bandwidth wireless systems. No one communication technology provides the entire coverage.

Smart Metering, or an AMI, relies on a complex communications infrastructure to exchange information with and control digital meters and other devices that make up the AMI. This communications infrastructure can consist of various communications networks and requires frequent transmittal of measurements and command and control data. Typically, there are three or more networks, namely the HAN, the NAN, and the WAN. As it can be seen by the arrows in above Figure, there are six communication paths in the AMI. These paths can vary from simple low speed power line carriers to very high bandwidth wireless systems. No one communication technology provides the entire coverage.

AMI will improve three key features of grid system including:

1. *System Reliability:* AMI technology improves the distribution and overall reliability of electricity by enabling electricity distributors to identify and automatically respond to electric demand, which in turn minimizes power outages.

1. *Energy Costs:* Increased reliability and functionality and reduced power outages and streamlined billing operations will dramatically cut costs associated with providing and maintaining the grid, thereby significantly lowering electricity rates.

1. *Electricity Theft:* Power theft is a common problem in India. AMI systems that track energy usage will help monitor power almost in real time thus leading to increased system transparency.

SMART Governance

–SMART Governance‖ is about the future of the public services, it‘s about greater efficiency, community leadership, mobile working and continuous improvement through innovation.

Additional Information

SMART Governance is about using technology to facilitate and support better planning and decision making. It is about improving democratic processes and transforming the ways that public

services are delivered. It includes e-government, the efficiency agenda and mobile working. Eg: National Institute of Smart Governance

What is Smart Governance?

Concerns the efficiency of public services of a **smart** city and their improvement through innovations without forgetting the democratic inclusiveness of its residents. Learn more in: Scenarios for a Smart Tourism Destination Transformation: The Case of Cordoba, Spain

- The process of **governance** based on using ICT tools and the Internet to provide information and public services, on communication and collaboration between government and citizens and on the principles of good **governance**.
- **Smart governance** is about the use of technology and innovation for facilitating and supporting enhanced decision making and planning. It is associated with improving the democratic processes and transforming the ways that public services are delivered.
- The process of **governance** based on using ICT tools and the Internet to provide information and public services, on communication and collaboration between government and citizens and on the principles of good **governance**.

The last two centuries have seen many revolutionary changes in government structures. The feudal structure of earlier administration was mainly concentrated on defense and revenue with small involvement in health and education sector.

The context of the administration changed during the times of industrial revolution with the emergence of new political philosophies. The post-second world war period saw the emergence of the welfare state leading to a new outlook for administration. The era of globalization led to "New Public Management" which led to a complete reconfiguration of administrative outlook.

Good Governance through "SMART" Governance

In the 21st Century, the democratic states have become a synonym of a welfare state. The democratic government acts as a service provider. The government uses government machinery to reach to its citizen and provide necessary services in an efficient and equitable manner.

SMART captures the important attributes of Good Governance i.e. **Simple, Moral, Accountable,**

Responsive and Transparent government.

- **Simple** --Citizen expects a user-friendly government with the simplicity of laws, rules, regulations, processes. To achieve simplicity, in the delivery of services. The government is implementing Single Window one-stop services through CSC to facilitate delivery of services to a common citizen under its e-governance framework.
- **Moral** - The word Moral in SMART governance denotes emergence of a new system of governance based on moral values. The cleaning up process is very slow but it is very essential for the survival of values cherished by Good Governance. Some of the processes like systematic changes in electoral reforms, downsizing of government, enhancing literacy, increasing awareness and participation can take a longer duration to show its effects. However, such programmes should not be overlooked.

- **Accountability** - It is the soul of a democratic government and the very cornerstone of public administration.
- **Responsiveness** It refers to the quality of being attentive to the needs of common man. A responsive administration shows urgency in responding to different problems faced by the common man. Citizen Charter developed in 1990's in the UK can be used to improve the responsiveness of the government. Citizen's Charter is a set of assurances given by the government agency on the quality of service and time limit for delivery.
- **Transparency** - Transparency brings some of the essential virtues into public life such as equity, level playing field and the rule of law. These virtues confer social benefits to the people with no discretion and no scope of corruption. Transparency arises out of the citizen's Right To Information (RTI)**, the right to know why certain decisions were taken.**

Smart Governance To e-Governance

- The government of India has set a vision to develop in India a Simple, Moral, Accountable, Responsive, and Transparent, i.e. SMART Government. The new approach is to evolve a Citizen-Centric Design (CCD), especially for e-Governance applications.
- e-Governance can act as a trigger or a catalyst for progress in different spheres & will reduce the digital divide.
- Citizen Participation
- Citizen's charter to e-Governance
- Implementation of the Right to Information
- E-democracy

Resistance Factors To e-Governance

E-governance brings a significant change in the way the government functions. These changes in the function are bound to create resistance due to the cultural legacy of the government departments.

Reasons for resistance to e-governance

Government culture of secrecy

- The culture of secrecy is strengthened by the Official Secrets Act. But of late there has been an attempt to bring greater transparency in government functioning and empowering the citizen. Right to Information Act (RTI) is the outcome.

Corruption

- E-Governance tries to remove these basic factors that promote corruption in the government system.
- India ranks 85th among 133 countries in the Corruption Perception Index of Transparency International. The World Bank Report 2004 argued that red tape and stifling bureaucracy juxtaposed with abject poverty is no coincidence. The report shows how governments in poor countries often tie their poor people in the thicket of useless regulation. e-Governance is supposed to be an antidote for red tape, and hence it is being promoted under the framework of smart governance.

Culture of seniority

- Seniors show neglect and indifference in the whole effort of introducing e-Governance.

Lack of imagination

- Government Department focuses on rules and regulation rather than on the delivery of the services. There is lack of innovation and initiatives to tackle new challenges. The bureaucratic culture of Red Tapism even discourages innovation.

So to implement Smart/e-governance these cultural resistance needs to be tackled first with proper behavioural training and making the government officials understand the changed philosophy of governance.

e-Government functions :

- Citizen access to government information,
- Facilitating general compliance with a set of rules or regulations,
- Citizen access to personal benefits,
- Procurement including bidding, purchasing and payment,
- Government to Government information and service integration and
- Citizen Participation

State Of Smart Governance in India

Over the years various steps have been taken by governments at the state level and at central departments/Ministries to implement e-governance projects to promote Smart Governance. A national e-governance plan has been created which provides the path to achieve smart governance. Massive countrywide information technology infrastructure is being created to provide government services to the remotest of villages in an efficient, reliable and user-friendly manner. The ultimate aim is to bring government to the doorstep of a citizen as articulated in the vision of National e-governance plan.

Some recent examples of the work done to promote SMART Governance

Prajavani (Andhra Pradesh): It's a unique public-private initiative which allows citizens to connect to the government through Information Technology. It helps in redressal of grievances and access to information from the government.

Jankari (Bihar): A phone-based RTI support system. A citizen can register his RTI through a phone call and get information through the phone.

Online Land Records (Chhattisgarh): Digitization of land records in the state. Simplifying the process of registry of land, during the buying and selling process.

e-city (Gujarat): A one-stop civic shop. It provides information about birth and Death registration, building plan, utility services, primary health care etc.

Conclusion

Though India has covered a long journey, which is visible through its improved ranking in ease of doing business index and initiatives like e-biz portals, single window clearance in multiple sectors. However, India is still plagued by multiple issues as far as the dynamic implementation of SMART governance framework is concerned.

Smart Transportation (Intelligent Transportation)

What Is An Intelligent Transport System?

Every city needs an effective transportation and mobility system, and in smart cities, this takes the shape of an Intelligent Transportation System (ITS). The scope of ITS includes traffic and mobility management, managing movement of vehicles and assisting drivers, enhancing transport infrastructure, and providing improved interfaces for transport systems. ITS is a growing market, valued at $20 billion worldwide in 2015 and expected to more than double itself by 2024.

ITS is used in car parks, traffic lights, toll booths, bridges, and roads; and is used to create interconnected transportation systems with open communication between devices and vehicles. ITS can actively manage traffic so it flows smoothly and public transport arrives as scheduled.

ITS ensures citizens have access to real time information about traffic and public transportation conditions. This reduces travel time for commuters and makes traveling throughout the city easier, safer and more comfortable.

Benefits of IoT in Smart Transportation

1. Finding new revenue
2. Increasing safety
3. Getting important, actionable information
4. Higher efficiencies
5. Delivering new experiences

Applications of IoT in Transportation

IOT Services for Smart Transportation

a. Car

Most of the companies in the automotive sector have started envisioning a future for motors in which IoT era makes vehicles ‒smart,‖ appealing alternatives identical to MRT, IoT gives few substantial improvements to private cars. The maximum benefits come from better control over related infrastructure and the inherent flaws in vehicle transport. However, IoT does enhance private motors as non-public spaces. IoT brings the identical enhancements and customization to a vehicle as those inside the home.

b. Avenue

The number one concerns of traffic is handling congestion, decreasing accidents, and parking. IoT allows us to better take a look at and analyze the go with the flow of visitors through gadgets in any respect traffic commentary factors. It aids in parking with the aid of making storage glide obvious whilst present-day techniques offer little if any statistics.

Accidents usually end result from more than a few of factors, but, visitors‘ management impacts their frequency. Production sites, poor rerouting, and a lack of facts about traffic reputation are all issues that result in incidents. IoT provides solutions within the shape of higher facts sharing with the general public, and among diverse events directly affecting road visitors.

c. Rails & Mass Transit

Contemporary structures deliver sophisticated integration and overall performance, but, they appoint older era and techniques to MRT. This has outcomes that are better in the management of ordinary overall performance, maintenance issues, maintenance, and improvements.
Mass transit options past preferred MRT can be afflicted by a lack of the combination important to convert them from a choice to a committed carrier. IoT offers an inexpensive and advanced manner to optimize overall performance and convey traits of MRT to other transportation alternatives like buses. This improves offerings and service transport in the regions of scheduling, optimizing shipping instances, reliability, handling system troubles, and responding to purchaser wishes.

d. Industrial Transportation

Transportation advantages expand to commercial enterprise and manufacturing by means of optimizing the transport arm of businesses. It reduces and removes troubles related to bad fleet control through better analytics and control together with monitoring idling, gasoline consumption, tour situations, and travel time among factors. This results in product transportation operating greater like an aligned provider and less like a group of contracted offerings.
So, this was all about Applications of IoT in Transportation. Hope you like our explanation.

Smart Transportation System

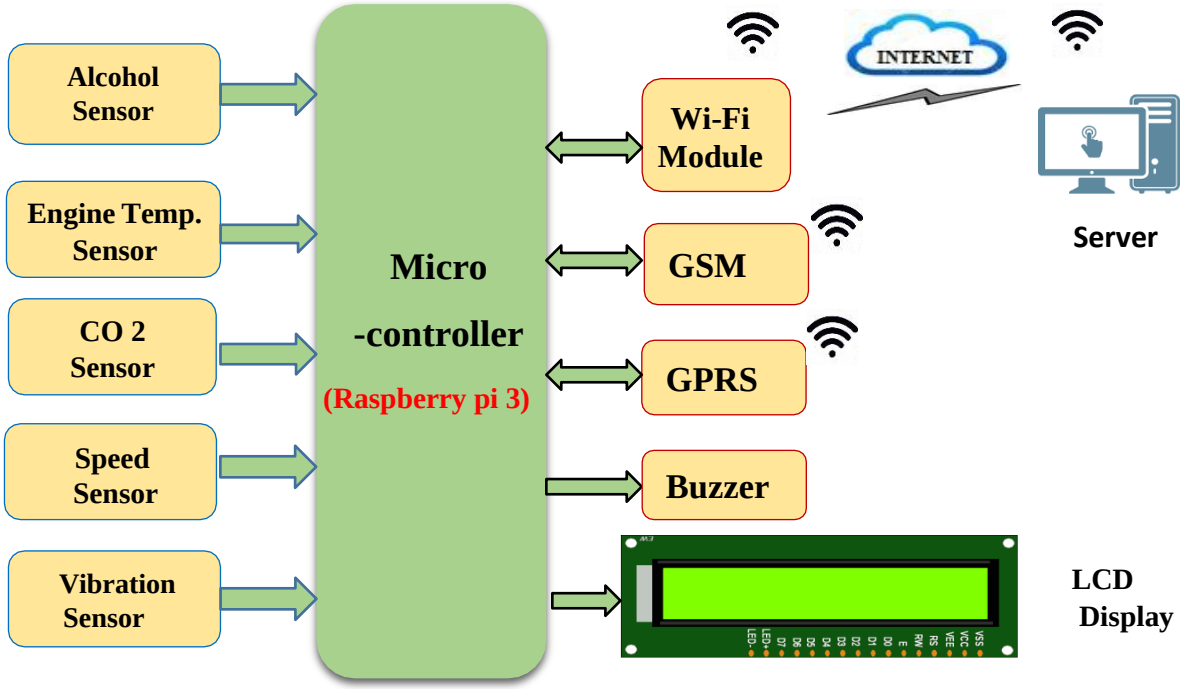


Fig. Block diagram of Smart Transportation System

Internet of Things (IoT) is a world-wide network connecting all the smart objects together. It is the medium by which all things are enable to talk with each other. Whenever those smart things are restricted to connected vehicles only, then it is called as Internet of Vehicles. (IoV) In recent years popularity of private cars is getting urban traffic more and more crowded. As a result traffic is

becoming one of important problems in big cities in all over the world. Some of traffic concerns are the congestions and collision which have caused a huge waste of time, property damage & environmental pollution. This research paper provides Internet of Vehicle (IoV) based on intelligent traffic management system, which is featured by high compatibility, easy to upgrade to replace traditional traffic management system, low cost and the proposed system can improve road traffic tremendously. The Internet of Vehicles is based on the internet, detection technologies and network wireless sensors to recognize traffic object, monitoring, managing, and tracking & processed automatically. The basic functionalities of the proposed system include monitoring of speed limits, pollution checks, emergency response to road accidents and providing security etc. should also be taken care to make life easier.

Hardware Description

IR Proximity sensor:

This Medium Range Infrared (IR) sensor offers simple, fast and user friendly. Obstacle detection using infrared; is non-contact detection. Sensor uses Small LED as indicator for detection status. It can detect Obstacle up to 15cm. It has adjustable sensing range (2cm – 15cm).

Co2 sensor:

MQ-9 gas sensor make detection by method of cycle high and low temperature, and detect CO when low temperature (heated by 1.5V). The sensor‘s conductivity is higher along with the gas concentration rising. The sensor could be used to detect different gases contains CO & combustible gases, it‘s with low cost and suitable for different applications.

MQ-9 gas sensor is used as co2 sensor it has high sensitivity to Carbon Monoxide, Methane and LPG.

Vibration sensor:

Vibration sensor buffers a piezoelectric transducer that are responds to strain Changes by generating a measurable output voltage which is Proportional to the strength of vibration.

Speed sensor:

The LM-193 sensor have sensing elements consisting of two independent precision voltage comparators with an offset voltage specification as low 2.0 mV max for two comparators which were designed specifically to operate on a single power supply over a wide range of voltages.

Tyre pressure sensor:

The NPC- 1220 series is intended packaged in a dual-in-line configuration for printed circuit board mounting. The NPC-1220 sensor series of solid state pressure sensors are used to provide a cost effective solution for such applications that require calibrated performance over a wide temperature range.

Alcohol sensor:

Sensor provides an analog output based on alcohol concentration. This alcohol gas sensor is suitable for detection of alcohol concentration on breath, just like your common Breathalyzer. It has a high sensitivity and fast response time.

Raspberry pi 3:

The Raspberry Pi 3 Model B is the latest version of the Raspberry Pi, it is a tiny credit card size computer developed in the United Kingdom by the Raspberry Pi Foundation. Raspberry Pi Model B was released in February 2016 and has onboard Bluetooth, WiFi and USB boot capabilities.

Smart Parking using IoT

Now a days the number of personal vehicles usage is increasing on a large scale. People prefer personal vehicles than public transportation. It is very difficult and frustrating as well to find parking space in most metropolitan areas, especially during the rush hours. It is often costly in almost every major city in the world to find proper and secure parking space. Due to this there is a need to provide sufficient parking places providing plenty of slots to help the user park his vehicle safely. The aim of this paper is to propose a design of IOT based smart Parking System that regulates the number of cars to be parked on designated parking areas.

This is done by automating the Parking and non-parking of the vehicles with the help of IOT and Android Application. Our system will reduce the human work and help user to park their vehicle with ease and more security. It will use the vehicle number plate to authenticate the vehicle and also it will be used to reserve the space for vehicle in parking lot. This system will be robust and can be used for all types of parking lots.

Android application will be used to get the location of nearest parking lots, real time space availability also the user will be able to pay and book their parking slots through the application. The parking lots will be operated by Raspberry pi for data transfer and it will collect data from the sensors and camera for retrieving the vehicles details and allocated parking space for them.

This system is very much secure and it will work fast from other system as KNN (K Nearest Neighbor) algorithm is used with OpenCV, which will help to recognize number plate very accurately and fast. This system contains hardware as Ultrasonic Sensor, USB camera, IR sensor and raspberry pi3. This is done by using Python programming language for codes of Deep learning, Java for Android Application and Google firebase for database.

Daily Parking Issues

Not enough parking spaces in a densely populated area.
Poor use of available parking spaces.
Time and gas used to find open parking spaces.
Difficulty finding vehicles in large parking lots.
Traffic congestion centered around poorly executed parking structures.
Business parking spaces are taken by commuter parking.
Inconvenient parking spaces.
Poor parking price models.
Proper handling of handicapped spaces.
Unused private parking spaces.
Environmental impact of excessive fuel use in seeking parking.
Unclear parking policies.

What Is Smart Parking?

Combining human innovation and advanced technology, smart parking is a strategy that smart cities can use to increase parking efficiency and combat numerous issues. Smart parking will reduce fuel use because the driver won‘t be circling streets looking for a place to park because they‘ll have an identified space. Drivers will also save money and time with denser, easier parking. Smart parking utilizes smartphones and other sensing devices to ascertain the occupancy of a parking structure or level. It‘s accomplished through the use of cameras, counting utilities at the entrances or gates of parking structures, sensors embedded in the pavement of individual parking spaces, etc.

System Architecture of the Smart Parking System

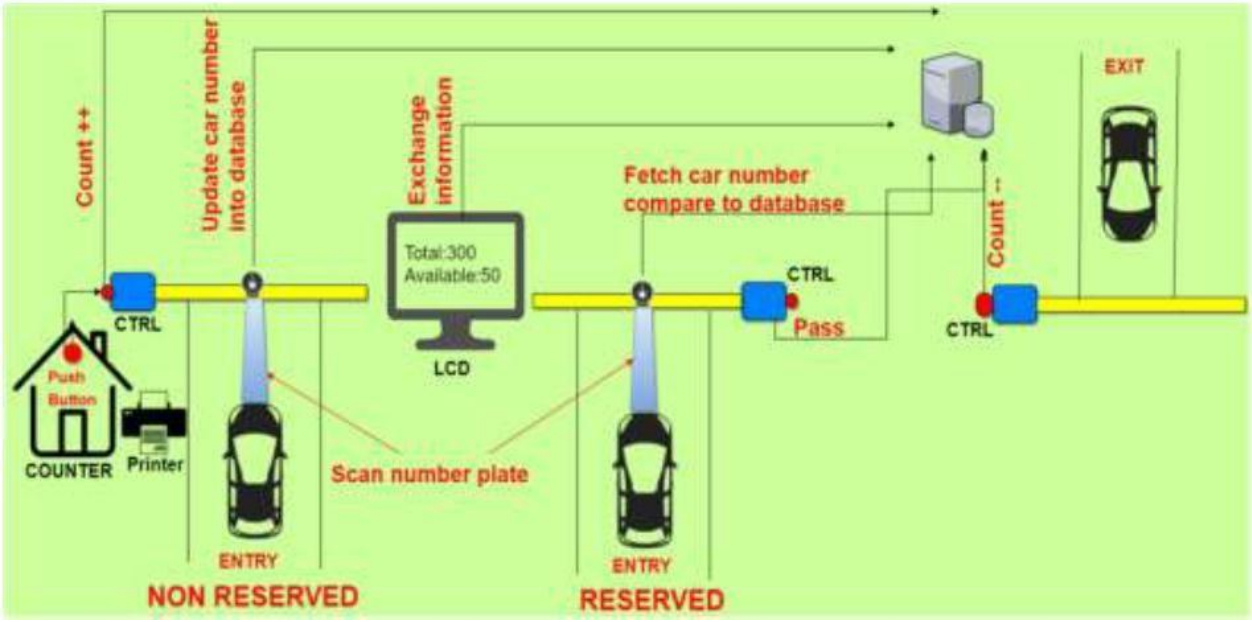


Figure: System Architecture of the Smart Parking System

The System Architecture shows the working of the smart parking system, In this there will be different gate for the reserved parking and non-reserve parking for reserving the parking space the user should have the app of the smart parking ,form there the user will reserve its place in the parking station for parking by paying the charge through online.

Once the user reserve its parking in the parking station than he/she will get the route to reached to the parking station with the help of google map. And when the user reached to the nearest parking station the number of his car will be fetch by the camera and sensor which is fit on the gate and after fetching the number, the number will be matched through the database of the parking station if the number get matched than the barrier on the gate will be open and the user can access the parking lot.

It has been used the concept of the balloon, there will be a balloon at every parking space and if the parking space is vacant the balloon will be up in the air and if it is occupied than the balloon will down towards the ground. The balloon can also be used for the advertisement for the different purpose. If the user didn‘t want to reserve its parking space or if they are unable to do the online transaction by any problem than the user could go to the unreserved gate and pay the charge for the parking and get the receipt of the payment and can move inside the parking station and park the car.

Smart Parking using IoT

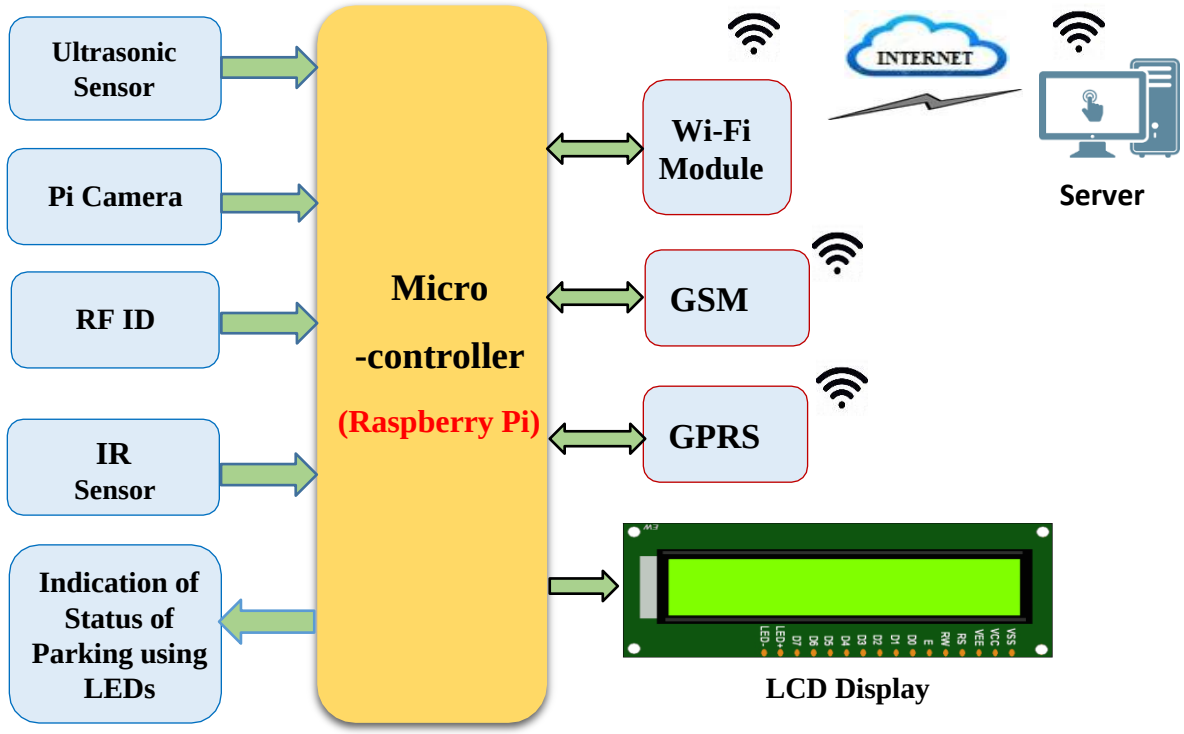


Fig. : Block diagram of Smart Car Parking using IoT

Ultrasonic sensor and Pi Camera: Act as an input to collect the data about the status of parking.
Raspberry Pi: The input from parking lot sensor is given to microcontroller system. Upon any changes in input, microcontroller will update the information collected from sensor to Firebase server.

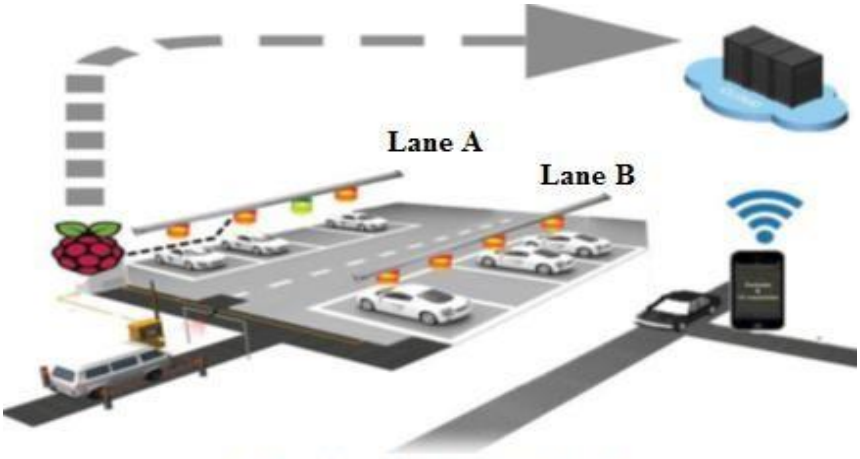


Fig.: Smart Parking System

IR Sensor
The IR sensor used to detect the car in parking. If car is present then it shows on cloud as that parking slot is allowed if not allowed then it will shows that parking slot empty.

Reservation
If you booked parking then from booking time then it will booked for next 5 minutes and countdown start at timer.
Buzzer
If any other person parked their car in booked slot then alarm will get buzzed periodically for some time as per set.
LED
On LED we will show the indication of booked slot.
LCD
LCD part is used in security area for check out the parking is allotted or free.

WORKING PRINCIPLE

Algorithm

- Start.
- Turn on the power supply.
- IR sensor will get activated.
- Search online for empty parking slot from android application.
- Space detection will start.
- If space is detected data sends and stored on cloud by sending system status by GSM.
- LED will start showing the number of parking slots.
- Display on front LCD that the slot is booked.
- Else go to step 5.
- Shows space on cloud.
- We can now book empty parking slot online.
- LED indication will get off.
- IR sensor will open a gate.
- End.

What are some of your favorite examples how brands are partnering with cities to address social issues

16 Brands Doing Corporate Social Responsibility Successfully

Corporate Social Responsibility (CSR) allows businesses large and small to enact positive change. When companies choose to do what is right not only for their bottom line but also benefit financially while building trust with consumers.

Consumers feel that when they use a product or service of a socially responsible company, they are doing their part. The more socially responsible the company, the more supportive the community and consumers become.

Corporate social responsibility helps build trust, raise awareness, and encourage social change. Although there are tens of thousands of companies doing their part, large global corporations‘

efforts have far-reaching results that can impact major world issues from hunger and health to global warming. Here are examples of how some major brands are doing CSR successfully.

Examples of Corporate Social Responsibility in Action

Corporate social responsibility comes in many forms. Even the smallest company impacts social change by making a simple donation to a local food bank. Some of the most common examples of CSR include:

- Reducing Carbon Footprints
- Improving Labor Policies
- Participating In Fairtrade
- Charitable Giving
- Volunteering In The Community
- Corporate Policies That Benefit The Environment
- Socially And Environmentally Conscious Investments

1. Innovation: Johnson & Johnson

An excellent example of CSR is global giant Johnson & Johnson. They have focused on reducing their impact on the planet for three decades. Their initiatives range from leveraging the power of the wind to providing safe water to communities around the world. Its purchase of a privately-owned energy supplier in the Texas Panhandle allowed the company to reduce pollution while providing a renewable, economical alternative to electricity. The company continues to seek out renewable energy options with the goal to procure 35% of their energy needs from renewable sources.

2. Google

Google is trusted not only for its environmentally friendly initiatives but also due to their outspoken CEO Sundar Pichai. He stands up against social issues including President Donald Trump‘s anti-Muslim comments. Google also earned RI‘s highest CSR score much in part due to their data center using 50% less energy than others in the world. They also have committed over $1 billion to renewable energy projects and enable other businesses to reduce their environmental impact through services such as Gmail

3. Coca-Cola

Coca-Cola‘s massive fleet of delivery trucks contributed 3.7 million metric tons of greenhouse gases to the world. They have made major changes to their supply chain practices including investing in new alternatively fueled trucks. Their initiatives are intended to create a 25% reduction in their carbon footprint by 2020

4. Ford Motor Company

Ford plans to reduce their greenhouse gas emissions using their Eco Boost engine to increase fuel efficiency. It also plans to introduce 40 electrified vehicles (electric and hybrid) by 2022, in an investment of $11 billion. According to Ford: ―We‘re all in on this and we‘re taking our mainstream vehicles, our most iconic vehicles, and we‘re electrifying them. If we want to be successful with electrification, we have to do it with vehicles that are already popular.‖

In addition, American Ford dealerships rely on wind sail and solar PV systems to power their locations greatly reducing their use of electricity.

5 & 6. Netflix & Spotify

From a social perspective, companies such as Netflix and Spotify offer benefits to support their employees and families. Netflix offers 52 weeks of paid parental leave, which can be taken at any time whether it is the first year of the child's life or another time that suits their needs. This compares to 18 weeks at other tech companies.

Spotify offers a similar program, although for a shorter duration of 24 weeks of paid leave.

7. Pfizer

Pfizer uses the term corporate citizenship to coin their CSR initiatives and believes it is a core part of their company and 'simply how they do business'. Across the globe, the company drives initiatives that raise awareness for non-infectious diseases as well as providing healthcare for women and children who otherwise would not have the care they need. One example of this is the reduction in the price of their Pevenar 13 vaccines (for pneumonia, ear and blood infections) for those in need and in situations such as refugees and emergency settings.

8. Wells Fargo

Wells Fargo donates up to 1.5% of its revenue to charitable causes each year. which has raised $286.5 million in 2017 alone to more than 14,500 nonprofits through philanthropy such as food banks and incubators to hasten the speed to market for start-ups. They also provide employees two paid days off per year to volunteer and give back to the charity of their choice.

9. TOMS

TOMS mission is to donate a pair of shoes for every pair they sell and has resulted in the donation of over 60 million pairs of shoes to children in need. Profits are used to assist the visually-impaired by providing prescription glasses and medical treatments, provide' safe' drinking water and build businesses in developing countries to create jobs. They are also strong anti-bullying advocates and work with several non-governmental organizations and nonprofits to set examples of ethical behavior.

10. Bosch

Half of Bosch‘s research and development budget is invested in creating environmental protection technology. By 2021 the company will have invested €50 million to support universities and research programs that are focused on the environment, energy and mobility in Germany, India, the U.S and China through Bosch Energy Research Network, otherwise known as BERN.

11. GE

It's been more than a decade since General Electric launched Ecomagination, its renewable business strategy with a mission to double down on clean technology and generate $20 billion in revenue from green products. In those ten years, it has manufactured its Evolution Series Tier 4 Locomotive, which will reduce emissions by more than 70% and launched the Digital Wind Farm which can boost a wind farm‘s energy production by 20%.

12. Starbucks

With an eye to hiring, Starbucks is looking to diversify their workforce and provide opportunities for certain cohorts. By 2025 it has pledged to hire 25,000 veterans by 2025 as part of their socially responsible efforts. This hiring initiative will also look to hire younger people with the aim of "helping jump-start careers by giving them their first job'. While globally the company has joined with the UN Refugee Agency to scale up the company‘s support and efforts to reach refugee candidates to hire 10,000 refugees by 2022.

13. New Belgium Brewing Company

This brewing company owned entirely by its employees through a stock ownership plan is focused on sustainability. Its Fort Collins brewery produces 18% of its own electricity through solar panels and wastewater. It also contributes to bicycle and eco-focused organizations. According to their Director of CSR, Katie Wallace: –We consider social and environmental well-being to be intricately intertwined.‖

14. The Walt Disney Company

Disney is committed to reducing their carbon footprint with goals for zero net greenhouse gas emissions, zero-waste, and a commitment to conserve water. They are actively ensuring that they set strict international labor policies to protect the safety and rights of their employees.

They are also active in the community and encourage employees to do the same. They also have healthy living initiatives to promote healthy eating habits amongst employees.

15. Lego

Lego will invest $150 million over the next 15 years with a focus on addressing climate change and reducing waste. It has reduced their packaging as well as investing in an alternative energy source and plans to source 100% renewable energy by 2020. To accomplish this the company will hire a team to support its commitment to using sustainable materials and plans to reach a 90% recycling rate.

/