

***LECTURE NOTES ON
INTERNET OF THINGS
SEM- 6TH***



Prepared By
Er. Sabyasachi Patra
Asst.Prof., Electronics & Tele-Communication Engineering

Department of Electronics & Tele-Communication Engineering
C V RAMAN POLYTECHNIC
BBSR

CONTENTS

Sl.No	Chapter Name	Page No
1.	Syllabus	3
2.	Introduction to IoT	4 - 15
3.	Elements of IoT	16 - 25
4.	IoT Application Development	26 - 31
5.	Smart Technology	32 - 44
6.	Smart TVs	45 - 49
7.	IoT case studies	50 - 59

Th.4(ii)- INTERNET Of THINGS (Elective) E & TC 6th Semester

Syllabus

1. Introduction to IoT

1.1 What is IoT..1.2 Architectural Overview, 1.3 Design principles and needed capabilities, 1.4 IoT Applications, Sensing, Actuation, 1.5 Basics of Networking, M2M and IoT Technology, 1.6 Fundamentals- Devices and gateways, 1.7 Data management, Business processes in IoT, 1.8 Everything as a Service(XaaS), 1.9 Role of Cloud in IoT, Security aspects in IoT.

2. Elements of IoT

2.1 Hardware Components- Computing (Arduino, Raspberry Pi), 2.2 Communication, Sensing, Actuation, I/O interfaces, 2.3 Software Components- Programming API (using Python/Node.js/Arduino) for Communication, 2.4 Protocols-MQTT, ZigBee, Bluetooth, CoAP, UDP, TCP.

3. IoT Application Development

3.1 Solution framework for IoT applications, 3.2 Implementation of Device integration, 3.3 Data acquisition and integration, 3.4 Device data storage- Unstructured data storage on cloud/local server, 3.5 Authentication, authorization of devices.

4. Smart Technology

4.1 Understanding the IoT Big Picture, 4.2 Building the Internet of Things, 4.3 Understanding Smart Devices, Building Blocks, 4.4 Understanding Network Connections, 4.5 Understanding IP Addresses

4.6 Understanding cellular Network & Mesh Network

5. Smart TVs: Viewing in a Connected World

5.1 What is Smart TV & its use, 5.2 What is inside Smart TV, 5.3 What a Smart TV does, 5.4 Smart TV Operating Systems, 5.5 What is Smart TV Set-TopDevices, 5.6 Intergrating Smart TV in to IOT

6. IoT Case Studies

6.1 IoT case studies (any one)

a. Smar t Home, b. Smart car c. Smar t Cities d. Smart Drones 6.2 Industrial automation,

UNIT 1. Introduction to IoT

1.1 What is IoT

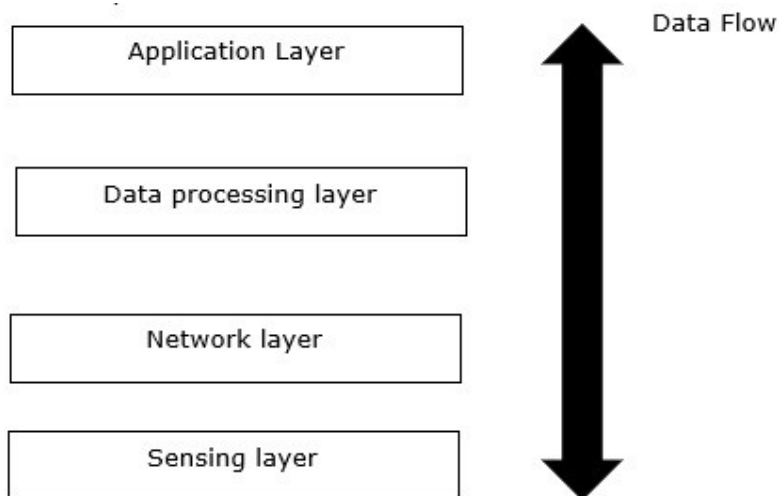
Internet of Things (IoT) is a system of interrelated, internet-connected objects/Things which are able to collect and transfer data over a wireless network without human intervention.

For example, smart fitness bands or watches, driverless cars or drones, smart homes that can be unlocked through smartphones and smart cars, etc.

1.2 Architectural Overview

Architecture of IoT

The basic fundamental architecture of IoT which consists of four stages as shown in the diagram given below –



- **Sensing Layer** – The first stage of IoT includes sensors, devices, actuators etc. which collect data from the physical environment, processes it and then sends it over the network.
- **Network Layer** – The second stage of the IoT consists of Network Gateways and Data Acquisition Systems. DAS converts the analogue data (collected from Sensors) into Digital Data. It also performs malware detection and data management.
- **Data Processing Layer** – The third stage of IoT is the most important stage. Here, data is pre-processed on its variety and separated accordingly. After this, it is sent to Data Centres. Here Edge IT comes into use.
- **Application Layer** – The fourth stage of IoT consists of Cloud/Data Centres where data is managed and used by applications like agriculture, defence, health care etc.

Advantages

The advantages of IoT are as follows –

- **Cost Reduction** – IOT devices catch any problem very fast as compared to traditional troubleshooting. It not only saves time but also saves costs of large repairs.

- **Efficiency and Productivity** – An automated PDF conversion and creation tool will remove the hustle of PDF editing and archiving. Hence, increase in Efficiency and Productivity.
- **Business Opportunities** – IOT provides advanced analytics, smart utility grids which help Small Management Businesses to provide more valuable content and things to their customers.
- **Customer Experience** – Nowadays customer's experience is the most valuable thing in running a business. IoT has drastically increased the customer's experience. An example of customer experience is Home Automation. Since everything is connected, customers need not have to worry about appliances. One can turn off the appliance through mobile.
- **Mobility and Agility** – With the help of IoT, employees can do their work from any geographical location, anytime without any restrictions.

Disadvantages

The disadvantages of IoT are as follows –

- **Security** – The data is travelling all over the Internet. So maintaining its privacy is still a Big Challenge. End-to-end Encryption is a must in IoT.
- **Compatibility** – There is no International Standard for the monitoring of the equipment.
- **Complexity** – Most of the devices still contain some software bugs. Each device must be able to seamlessly interact with other devices in the network.
- **Safety** – Suppose a patient is left unattended by a doctor. And some notorious guy changes the prescription or Health monitoring devices malfunctioned. Then it can result in the death of the patient.
- **Policies** – Government authorities must take some steps to make policies and standards related to IoT to stop the Black marketing of IoT devices.

The technologies supported by IoT are as follows –

- Big Data Analytics
- Cloud
- Wireless Sensor Networks
- Embedded Systems

1.3 Design principles and needed capabilities

7 design principles for IoT

Designers focused on designing software services and screen based interfaces or physical products, designing IoT solutions creates totally new design challenges. IoT solutions consist of multiple elements: physical devices like sensors, actuators and interactive devices, the network connecting these devices, the data gathered from these devices and analyzed to create a meaningful experience and last but definitely not least, the physical context in which user interacts with the solution..

1. Focus on value

In the world of IoT, user research and service design are more crucial than ever. While early adopters are eager to try out new technology, many others are reluctant to take new technology into use and cautious about using it, due to not feeling confident with it. For your IoT solution

to become widely adopted, you need to dig deep into users' needs in order to find out where lies a problem truly worth solving and what is the real end user value of the solution. You also need to understand what might be the barriers of adopting the new technology in general and your solution specifically. For deciding on your feature set, you need research too.

2. Take a holistic view

IoT solutions typically consist of multiple devices with different capabilities and both physical and digital touch points. The solution may also be provided in co-operation with multiple different service providers. It is not enough to design one of the touch points well, instead you need to take a holistic look across the whole system, the role of each device and service, and the conceptual model of how user understands and perceives the system. The whole system needs to work seamlessly together in order to create a meaningful experience.

3. Put safety first

As the IoT solutions are placed in the real world context, the consequences can be serious, when something goes wrong. At the same time the users of the IoT solutions may be vary of using new technology, so building trust should be one of your main design drivers. Trust is built slowly and lost easily, so you really need to make sure that every interaction with the product/service builds the trust rather than breaks it. What it means in practise? First of all, it means understanding possible error situations related to context of use, HW, SW and network as well as to user interactions and trying to prevent them. Secondly, if the error situations still occur, it means appropriately informing the user about them and helping them to recover. Secondly, it means considering data security & privacy as key elements of your design. It is really important for users to feel, that their private data is safe, their home, working environment and everyday objects cannot be hacked and their loved ones are not put at risk. Thirdly, quality assurance is critical and it should not only focus on testing the SW, but on testing the end to end system, in a real-world context.

4. Consider the context

IoT solutions exist at the crossroads of the physical and digital worlds. Commands given through digital interfaces may produce real world effects, but unlike digital commands, the actions happening in the real-world cannot necessarily be undone. In the real world context lots of unexpected things can happen and at the same time user should be able to feel safe and in control. The context places also other kind of requirements to the design. Depending on the physical context, the goal might be to minimize distraction of the user .

5. Build a strong brand

Due to the real world context of the IoT solutions, regardless of how carefully you design things and aim to build trust, something unexpected will happen at some point and your solution is somehow going to fail. In this kind of situations, it is of utmost importance, that you have built a strong brand that truly resonates with the end users. When they feel connected to your brand, they will be more forgiving about the system failures and will still keep on using your solution. While designing your brand, you must keep in mind, that trust should be a key element of the

brand, one of the core brand values. This core value should also be reflected in the rest of the brand elements, like the choice of color, tone of voice, imagery etc.

6. Prototype early and often

Typically HW and SW have quite different lifespans, but as successful IoT solution needs both the HW and SW elements, the lifespans should be aligned. At the same time, IoT solutions are hard to upgrade, because once the connected object is placed somewhere, it is not so easy to replace it with a newer version, especially if the user would need to pay for the upgrade and even the software within the connected object may be hard to update due to security and privacy reasons. Due to these factors and to avoid costly hardware iterations, it's crucial to get the solution right, from the beginning of implementation. What this means from the design perspective is that prototyping and rapid iteration of both the HW and the whole solution are essential in the early stages of the project. New, more creative ways of prototyping and faking the solution are needed.

7. Use data responsibly

IoT solutions can easily generate tons of data. However, the idea is not to hoard as much data as possible, but instead to identify the data points that are needed to make the solution functional and useful. Still, the amount of data may be vast, so it's necessary for the designer to understand the possibilities of data science and how to make sense of the data. Data science provides a lot of opportunities to reduce user friction, i.e. reducing use of time, energy and attention or diminishing stress. It can be used to automate repeated context dependent decisions, to interpret intent from incomplete/inadequate input or to filter meaningful signals from noise. Understanding what data is available and how it can be used to help the user is a key element in designing successful IoT services.

Summarizing questions

What is iot design?

Iot design is the practice of gathering data of various iot systems and their interactions with a goal of creating a meaningful user experiences. Iot design takes a holistic look across the whole system, the role of each device and service, and the creates conceptual model of how user understands and perceives the entire iot system.

Why is iot design important?

For Iot solutions to become widely adopted, businesses need to dig deep into users' needs in order to find out where lies a problem truly worth solving and what is the real end user value of the iot solution. Businesses also need to understand what the barriers might be in adopting the new iot technologies in general and their iot solutions specifically.

What are some of the best practices of iot design?

Considering user safety, data security & privacy aspects and incorporating quality assurance in the iot design process are important for user trust. Prototyping early and rapid iteration of the hardware and the whole iot solution are essential. Understanding what data is available of how it can be used responsibly to help the user are key elements in successful iot design

Requirements of IoT

➤ Connectivity

IoT devices must be able to connect to the internet through a wired or wireless connection. Connectivity is essential for the Internet of Things (IoT), as it allows IoT devices to communicate with each other and with a central server or cloud platform.

Various technologies and protocols can enable internet connectivity in IoT devices, including Wi-Fi, Bluetooth, and cellular networks. The choice of technology will depend on factors such as the device's location, power requirements, and data transfer needs. For example, an intelligent irrigation system located in a remote field may use a cellular network to connect to the internet, as it may not have access to a Wi-Fi network.

➤ Sensors

IoT devices must have sensors that can gather data from their environment, such as temperature, humidity, motion, or sound. This data is then used to trigger actions or provide user information. Sensors can be integrated into the device or connected through external means, such as a sensor module or a smart hub.

Some common types of sensors used in the IoT include temperature sensors, humidity sensors, motion sensors, and sound sensors. They can be integrated into the device or connected through external means, such as a sensor module or a smart hub. The data collected by sensors are often used to trigger actions or provide user information. For example, an intelligent irrigation system may use data from weather sensors to determine the optimal amount of water to apply to a lawn, or a smart security camera may use facial recognition to determine whether a person entering a home is an authorized user.

➤ Intelligence

IoT devices must be able to process and analyze the data they collect and make decisions based on that data. This can be done through onboard software or by sending the data to a central server for processing. For example, a smart irrigation system may use data from weather sensors to determine the optimal amount of water to apply to a lawn, or a smart security camera may use facial recognition to determine whether a person entering a home is an authorized user.

Intelligence is an essential requirement for the Internet of Things (IoT). It allows IoT devices to process and analyze the data they collect and make decisions based on that data. Without intelligence, an IoT device would be unable to perform its intended function or provide value to the user.

How are Connectivity, Sensors, and Intelligence Used in IoT?

Connectivity – This point is self-explanatory. Establishing a decent connection is like the preliminary stage in implementing IoT devices. This is usually implemented using Wi-Fi and Bluetooth as they are the most easily accessible among other ways and get the job done.

Sensors – There are a lot of ways how sensors can be implemented in IoT. Here's a brief information about everything you need to know. They allow the device to gather information about its surroundings and transmit it to other devices or systems. For example, Temperature sensors measure the temperature of a particular environment or object. Motion sensors detect movement or change in position. They are often used in security systems. Proximity sensors

detect the presence or absence of things close to the sensor. Pressure sensors measure the pressure or force applied to a particular object. They are often used in tire pressure monitoring systems. All in all, the type of sensor to be used depends on the objective that has to be accomplished.

Intelligence – As discussed before, implementing intelligence in an IoT device is like making its brain, as it is responsible for making every decision. Mainly there are 2 ways of getting this done: Edge Computing & Cloud Computing. The edge computing approach uses local processing power within the IoT device to analyze and act on data in real-time. One can achieve this through microprocessors or microcontrollers within the device. On the other hand, the Cloud computing approach involves sending data from the IoT device to the cloud for processing and analysis. It can be done through a cloud computing platform or by connecting the device to a local gateway that communicates with the cloud.

1.4 IoT Applications, Sensing, Actuation

Application areas of IoT: Smart Home: The smart home is one of the most popular applications of IoT. The cost of owning a house is the biggest expense in a homeowner's life. Smart homes are promised to save the time, money and energy.

Smart cities: The smart city is another powerful application of IoT. It includes smart surveillance, environment monitoring, automated transformation, urban security, smart traffic management, water distribution, smart healthcare etc.

Wearables: Wearables are devices that have sensors and software installed which can collect data about the user which can be later used to get the insights about the user. They must be energy efficient and small sized.

Connected cars: A connected car is able to optimize its own operation, maintenance as well as passenger's comfort using sensors and internet connectivity.

Smart retail: Retailers can enhance the in-store experience of the customers using IoT. The shopkeeper can also know which items are frequently bought together using IoT devices.

Smart healthcare: People can wear the IoT devices which will collect data about user's health. This will help users to analyze themselves and follow tailor-made techniques to combat illness. The doctor also doesn't have to visit the patients in order to treat them

Sensor is a device used for the conversion of physical events or characteristics into the electrical signals. This is a hardware device that takes the input from environment and gives to the system by converting it. For example, a thermometer takes the temperature as physical characteristic and then converts it into electrical signals for the system.

Actuators in IoT

An IoT device is made up of a Physical object ("thing") + Controller ("brain") + Sensors + Actuators + Networks (Internet). An actuator is a machine component or system that moves or controls the mechanism or the system. Sensors in the device sense the environment, then control signals are generated for the actuators according to the actions needed to perform. A servo motor is an example of an actuator. They are linear or rotatory actuators, can move to a given specified angular or linear position. We can use servo motors for IoT applications and make the motor rotate to 90 degrees, 180 degrees, etc., as per our need.

IoT Categories IOT can be classified into two categories:

1. Consumer IoT(CIOT): The Consumer IoT refers to the billions of physical personal devices, such as smartphones, wearables, fashion items and the growing number of smart home appliances, that are now connected to the internet, collecting and sharing data. A Consumer IoT network typically entails few consumer devices, each of which has a limited lifetime of several years. The common connectivity used in this kind of solutions are Bluetooth, WiFi, and ZigBee. These technologies offer short-range communication, suitable for applications deployed in limited spaces such as houses, or small offices.

2. industrial internet of things (IIoT): It 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.

1.5 Basics of Networking, M2M and IoT Technology

There are various baseline technologies that are very closely related to IOT, They include: Machinet-to-Machine (M2M), Cyber-Physical Systems (CPS), Web Of Things(WOT)

Machine-to-Machine (M2M):

M2M is a subset of IoT that deals explicitly with connections between machines M2M communication allows devices to communicate without direct human input. M2M is all about connecting machines to the Internet to communicate with each other, manage their tasks and data, and generally “do business” with one another

- Machine-to-Machine (M2M) refers to networking of machines (or devices) for the purpose of remote monitoring and control and data exchange.
- An M2M area network comprises of machines (or M2M nodes) which have embedded network modules for sensing, actuation and communicating various communication protocols can be used for M2M LAN such as ZigBee, Bluetooth. These protocols provide connectivity between M2M nodes within an M2M area network.
- The communication network provides connectivity to remote M2M area networks.
- The communication network can use either wired or wireless network (IP based). While the M2M are networks use either proprietary or non-IP based communication protocols, the communication network uses IP-based network. Since non-IP based protocols are used within M2M area network, the M2M nodes within one network cannot communicate with nodes in an external network. • To enable the communication between remote M2M are network, M2M gateways are used.

What are the technologies of IoT?

IoT primarily exploits standard protocols and networking technologies. However, the major enabling technologies and protocols of IoT are RFID, NFC, low-energy Bluetooth, low-energy wireless, low-energy radio protocols, LTE-A, and WiFi-Direct.

1.6 Fundamentals- Devices and gateways

A device is a hardware unit that can sense aspects of it's environment and/or actuate, i.e. perform tasks in its environment. A device can be characterized as having several properties, including:

- Microcontroller: 8-, 16-, or 32-bit working memory and storage.
- Power Source: Fixed, battery, energy harvesting, or hybrid.
- Sensors and Actuators: Onboard sensors and actuators, or circuitry that allows them to be connected, sampled, conditioned, and controlled.
- Communication: Cellular, wireless, or wired for LAN and WAN communication.
- Operating System (OS): Main-loop, event-based, real-time, or full featured OS.
- Applications: Simple sensor sampling or more advanced applications.
- User Interface: Display, buttons, or other functions for user interaction.
- Device Management (DM): Provisioning, firmware, bootstrapping, and monitoring.

- Execution Environment (EE): Application lifecycle management and Application Programming Interface (API).

Device types

Group devices into two categories

- **Basic Devices:** Devices that only provide the basic services of sensor readings and/or actuation tasks, and in some cases limited support for user interaction. These devices are often intended for a single purpose, such as measuring air pressure or closing a valve. The main focus is on keeping the bill of materials (BOM) as low as possible by using inexpensive microcontrollers with built-in memory and storage, often on an integrated circuit with all main components on one single chip. Another common goal is to enable battery as a power source, with a lifespan of a year and upwards by using ultra-low energy microcontrollers. The microcontroller typically hosts a number of ports that allow integration with sensors and actuators, such as General Purpose I/O (GPIO) and an analog-to-digital converter (ADC) for supporting analog input. LAN communication is supported via wired or wireless technology, thus a gateway is needed to provide the WAN connection.
- **Advanced Devices:** In this case the devices also host the application logic and a WAN connection. They may also feature device management and an execution environment for hosting multiple applications. Gateway devices are most likely to fall into this category.

Gateways

an IoT gateway is a physical device or virtual platform that connects sensors, IoT modules, and smart devices to the cloud. Gateways serve as a wireless access portal to give IoT devices access to the Internet.

However, there is a fundamental difference between router and gateway. A router is a networking layer system that we can use for managing and forwarding the data packets into various computer networks. A gateway is very different. It is basically hardware or a device that acts as a gate among various available networks. It also acts as a node that serves as an entry point for various other networks. Let us discuss the individual functionalities of both of these.

A gateway serves as a translator between different protocols, e.g. between IEEE 802.15.4 or IEEE 802.11, to Ethernet or cellular.

There are many different types of gateways, which can work on different levels in the protocol layers. A gateway refers to a device that performs translation of the physical and link layer, but application layer gateways (ALGs) are also common. Some examples of ALGs include the ZigBee Gateway Device which translates from ZigBee to SOAP and IP, or gateways that translate from Constrained Application Protocol (CoAP) to HyperText Transfer Protocol/Representational State Transfer (HTTP/REST).

The gateway device is also used for many other tasks, such as data management, device management, and local applications.

Data management Typical functions for data management include performing sensor readings and caching this data, as well as filtering, concentrating, and aggregating the data before transmitting it to back-end servers.

Local applications — Examples of local applications that can be hosted on a gateway include closed loops, home alarm logic, and ventilation control, or the data management function above example.

Device management

Device management (DM) is an essential part of the IoT and provides efficient means to perform many of the management tasks for devices:

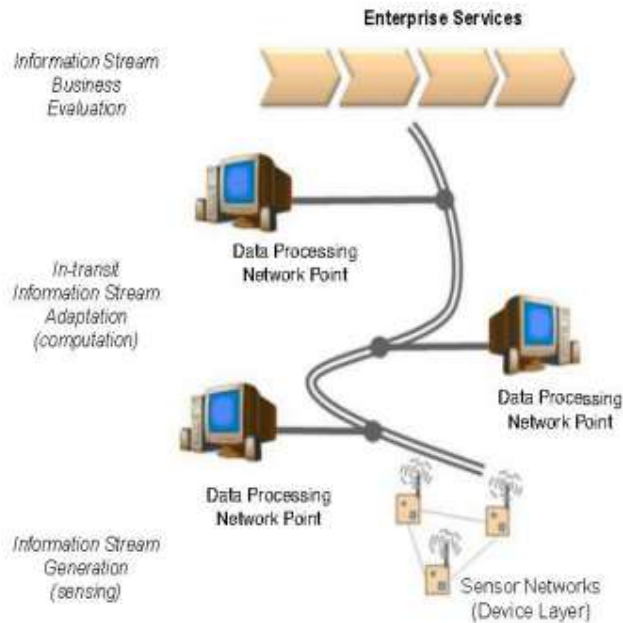
- **Provisioning:** Initialization (or activation) of devices in regards to configuration and features to be enabled.
- **Device Configuration:** Management of device settings and parameters.
- **Software Upgrades:** Installation of firmware, system software, and applications on the device.

The gateway functions as mediator between the server and the devices, and can operate in three different ways:

- If the devices are visible to the DM server, the gateway can simply forward the messages between the device and the server and is not a visible participant in the session.
- In case the devices are not visible but understand the DM protocol in use, the gateway can act as a proxy, essentially acting as a DM server towards the device and a DM client towards the server.
- For deployments where the devices use a different DM protocol from the server, the gateway can represent the devices and translate

between the different protocols (e.g. TR-069, OMA-DM, or CoAP). — The devices can be represented either as virtual devices or as part of the gateway

1.7 Data management, Business processes in IoT



In the era of M2M, where billions of devices interact and generate data at exponential growth rates, data management is of critical importance as it sets the basis upon which any other processes can rely and operate .

Some of the key characteristics of M2M data include: • **Big Data:** Huge amounts of data are generated, capturing detailed aspects of the processes where devices are involved.

• **Heterogeneous Data:** The data is produced by a huge variety of devices and is itself highly heterogeneous, differing on sampling rate, quality of captured values, etc.

• **Real-World Data:** The overwhelming majority of the M2M data relates to realworld processes and is dependent on the environment they interact with.

• **Real-Time Data:** M2M data is generated in real-time and overwhelmingly can be communicated also in a very timely manner.

• **Temporal Data:** The overwhelming majority of M2M data is of temporal nature, measuring the environment over time.

• **Spatial Data:** Increasingly, the data generated by M2M interactions are not only captured by mobile devices, but also coupled to interactions in specific locations, and their assessment may dynamically vary depending on the location.

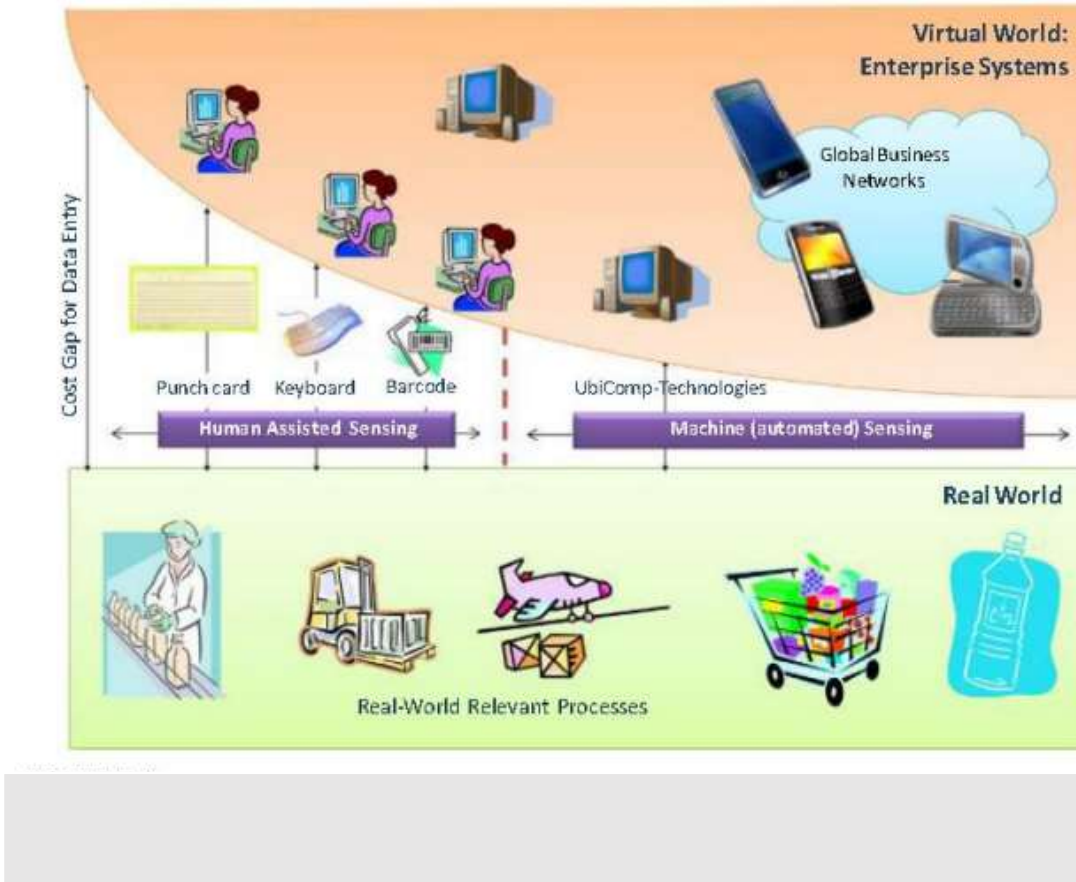
• **Polymorphic Data:** The data acquired and used by M2M processes may be complex and involve various data, which can also obtain different meanings depending on the semantics applied and the process they participate in.

• **Proprietary Data:** Up to now, due to monolithic application development, a significant amount of M2M data is stored and captured in proprietary formats. However, increasingly due to the interactions with heterogeneous devices and stakeholders, open approaches for data storage and exchange are used.

• **Security and Privacy Data Aspects:** Due to the detailed capturing of interactions by M2M, analysis of the obtained data has a high risk of leaking private information and usage patterns, as well as compromising security.

Managing M2M data

The data flow from the moment it is sensed (e.g. by a wireless sensor node) up to the moment it reaches the backend system has been processed manifold (and often redundantly), either to adjust its representation in order to be easily integrated by the diverse applications, or to compute on it in order to extract and associate it with respective business intelligence (e.g. business process affected, etc.). In Figure 5.5, we see a number of data processing network points between the machine and the enterprise that act on the datastream (or simply forwarding it) based on their endapplication needs and existing context



A business process refers to a series of activities, often a collection of interrelated processes in a logical sequence, within an enterprise, leading to a specific result. There are several types of business processes such as management, operational, and supporting, all of which aim at achieving a specific mission objective. — As business processes usually span several systems and may get very complex, several methods and techniques have been developed for their modeling, such as the Business Process Model and Notation (BPMN), which graphically represents business processes in a business process model. Several key business processes in modern enterprise systems heavily rely on interaction with real-world processes, largely for monitoring, but also for some control (management), in order to take business-critical decisions and optimize actions across the enterprise.

1.8 Everything as a Service(XaaS) & 1.9 Role of Cloud in IoT, Security aspects in IoT

Anything as a service” (XaaS) describes a **general category of services related to cloud computing and remote access**. It recognizes the vast number of products, tools, and technologies that are now delivered to users as a service over the internet. XAAS is a software architectural concept that enables deployment and development of applications, and offers services using web and SOA.

A computing paradigm is to integrate complex applications and services and use XAAS concept for deploying a cloud platform

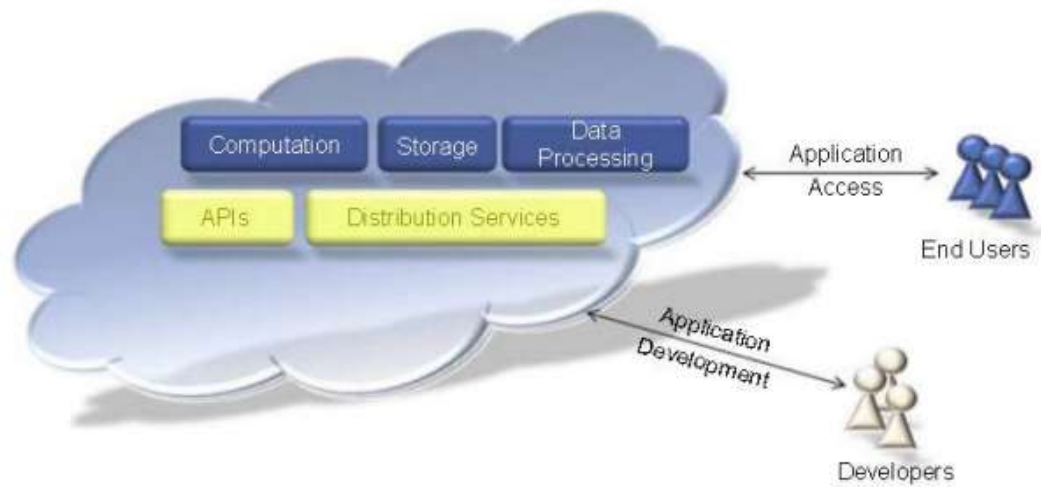


FIGURE 5.11

Conceptual Overview of Cloud Computing.

Cloud computing is a model for enabling ubiquitous, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications, and services) that can be provisioned, configured, and made available with minimal management effort or service provider interaction.

All applications need access to three things: compute, storage, and data processing capacities.

Cloud computing, a fourth element is added _ distribution services _ i.e. the manner in which the data and computational capacity are linked together and coordinated.

Characteristics of cloud computing

On-Demand Self-Service. A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed, or automatically, without requiring human interaction with each service provider.

Broad Network Access. Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g. mobile phones, tablets, laptops, and workstations).

Resource Pooling. The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. Examples of resources include storage, processing, memory, and network bandwidth.

Rapid Elasticity. Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand.

Measured Service. Cloud systems automatically control and optimize resource use by leveraging a metering capability, at some level of abstraction, appropriate to the type of service (e.g. storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

For M2M and IoT, these infrastructures provide the following:

1. Storage of the massive amounts of data that sensors, tags, and other “things” will produce.
2. Computational capacity in order to analyze data rapidly and cheaply.
3. Over time, cloud infrastructure will allow enterprises and developers to share datasets, allowing for rapid creation of information value chains.

IoT security refers to the methods of protection used to secure internet-connected or network-based devices.

IoT security is the family of techniques, strategies and tools used to protect these devices from becoming compromised. Ironically, it is the connectivity inherent to IoT that makes these devices increasingly vulnerable to cyberattacks.

Below are a few of the IoT security challenges that continue to threaten the financial safety of both individuals and organizations.

1. Remote exposure

Unlike other technologies, IoT devices have a particularly large attack surface due to their internet-supported connectivity. While this accessibility is extremely valuable, it also grants hackers the opportunity to interact with devices remotely. This is why hacking campaigns like phishing are particularly effective. IoT security, like cloud security, has to account for a large number of entry points in order to protect assets.

2. Lack of industry foresight

As firms continue with digital transformations of their business, so, too, have certain industries and their products. A reliance on technology can amplify the consequences of a successful data breach. What makes this concerning is that these industries are now relying on a piece of technology that is inherently more vulnerable: IoT devices. Not only that, but many healthcare and automotive companies were not prepared to invest the amount of money and resources required to secure these devices. This lack of industry foresight has unnecessarily exposed many organizations and manufacturers to increased cybersecurity threats.

3. Resource constraints

Lack of foresight isn't the only IoT security issue faced by newly digitized industries. Another major concern with the IoT security is the resource constraints of many of these devices. Not all IoT devices have the computing power to integrate sophisticated firewalls or antivirus software. Some barely have the ability to connect to other devices. IoT devices that have adopted Bluetooth technology, for example, have suffered from a recent wave of data breaches. The automotive industry, once again, has been one of the markets hurt the most

UNIT-2

Arduino

Arduino is an open-source electronics platform based on easy-to-use hardware and software. It consists of **microcontroller boards** that can be programmed to interact with sensors, motors, LEDs, and other electronic components. It uses the Arduino programming language and the Arduino Software (IDE), based on Processing.

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..

Arduino was born at the Ivrea Interaction Design Institute.

[An integrated development environment (IDE) is a software application that helps programmers develop software code efficiently.]

Advantage/Feature

Inexpensive - Arduino boards are relatively inexpensive compared to other microcontroller platforms.

Cross-platform - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems

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.

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.

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.

Type of arduino boards

Arduino boards are available with many different types of built-in modules in it. These modules are known as Shield.

Arduino Ethernet shield: It that allows an Arduino board to connect to the internet using the Ethernet library and to read and write an SD card using the SD library .

□ **Arduino Wireless shield:** It allows your Arduino board to communicate wirelessly using Zigbee .

□ **Arduino Motor Driver Shield:** It allows your Arduino boards to interface with driver of a motor etc. .

Arduino Type	Microcontroller	Clock Speed
Arduino Uno	ATmega328	16 MHz with auto reset

Elements of an Arduino Board can be done into two categories:

- Hardware
- Software

Hardware

The Arduino Development Board consists of many components that together makes it work. Here are some of those main component blocks that help in its functioning:

- **Microcontroller:** This is the heart of the development board, which works as a mini computer and can receive as well as send information or command to the peripheral devices connected to it. The microcontroller used differs from board to board; it also has its own various specifications.
- **External Power Supply:** This power supply is used to power the Arduino development board with a regulated voltage ranging from 9 – 12 volts.
- **USB plug:** This plug is a very important port in this board. It is used to upload (burn) a program to the microcontroller using a USB cable. It also has a regulated power of 5V which also powers the Arduino board in cases when the External Power Supply is absent.
- **Internal Programmer:** The developed software code can be uploaded to the microcontroller via USB port, without an external programmer.
- **Reset button:** This button is present on the board and can be used to resets the Arduino microcontroller.
- **Analog Pins:** There are some analog input pins ranging from A0 – A7 (*typical*). These pins are used for the analog input / output. The no. of analog pins also varies from board to board.
- **Digital I/O Pins:** There are some digital input pins also ranging from 2 to 16 (*typical*). These pins are used for the digital input / output. The no. of these digital pins also varies from board to board.
- **Power and GND Pins:** There are pins on the development board that provide 3.3, 5 volts and ground through them

Software

The program code written for Arduino is known as a sketch. The software used for developing such sketches for an Arduino is commonly known as the Arduino IDE. This IDE contains the following parts in it:

- **Text editor:** This is where the simplified code can be written using a simplified version of C++ programming language.
- **Message area:** It displays error and also gives a feedback on saving and exporting the code.
- **Text:** The console displays text output by the Arduino environment including complete error messages and other information
- **Console Toolbar:** This toolbar contains various buttons like Verify, Upload, New, Open, Save and Serial Monitor. On the bottom right hand corner of the window there displays the Development Board and the Serial Port in use.

Arduino IDE

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.

Programs written using Arduino Software (IDE) are called **sketches**. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information.

Arduino UNO Applications

The Arduino boards can work as a stand-alone project and can be interfaced with other Arduino boards or Raspberry Pi boards. Arduino UNO board is used in the following applications.

- ☐ Weighing Machines
- ☐ Traffic Light Count Down Timer
- ☐ Parking Lot Counter
- ☐ Embedded systems
- ☐ Home Automation
- ☐ Industrial Automation
- ☐ Medical Instrument
- ☐ Emergency Light for Railways

Raspberry Pi:

Raspberry Pi is the name of a series of single-board computers made by the Raspberry Pi Foundation, a UK charity, launched in 2012. The original Pi had a single-core 700MHz CPU and just 256MB RAM, and the latest model has a quad-core CPU clocking in at over 1.5GHz, and 4GB RAM.

The Raspberry Pi can open opportunities for create home automation projects, rather than using a proprietary closed system.

The Raspberry Pi is a very cheap computer that runs Linux, but it also provides a set of GPIO (general purpose input/output) pins, allowing you to control electronic components for physical computing and explore the Internet of Things (IoT).

Difference Between Arduino and Raspberry Pi

Parameters	Arduino	Raspberry Pie
Control Unit	The Control Unit of the Arduino is from the ATmega family.	The Control Unit of the Raspberry Pi is from the ARM family.
Basis	Arduino works on the basis of a microcontroller.	Raspberry Pi, on the other hand, works on the basis of a microprocessor.
Use	The Arduino basically helps in controlling all the electrical components that connect to a system's circuit board.	The Raspberry Pi primarily computes data and info for producing valuable outputs. It also controls the various components in any given system on the basis of the outcome (of the computation).

Structure of Hardware and Software	The Arduino boards have a very simple structure of software and hardware.	The Raspberry Pi boards consist of comparatively complex software and hardware architecture.
Type of CPU Architecture	Arduino has an 8-bit architecture.	Raspberry Pi has a 64-bit architecture.
RAM Usage	Arduino makes use of very little RAM of about 2 kB (Kilobytes).	Raspberry Pi always requires more RAM than Arduino of about 1 GB (Gigabytes).
Processing Speed	Arduino clocks 16 MHz (Megahertz) of processing speed in a system.	The Raspberry Pi clocks 1.4 GHz (Gigahertz) of processing speed in a system.
Cost Efficiency	It has a higher cost-efficiency because it is comparatively cheaper.	It has a lower cost-efficiency because it is comparatively more expensive.
I/O Drive Strength	The I/O current drive strength in the case of Arduino is higher.	The I/O current drive strength in the case of Raspberry Pi is lower.
Power Consumption	Arduino consumes power of about 200 MW (Milliwatts).	Raspberry Pi consumes about 700 MW.

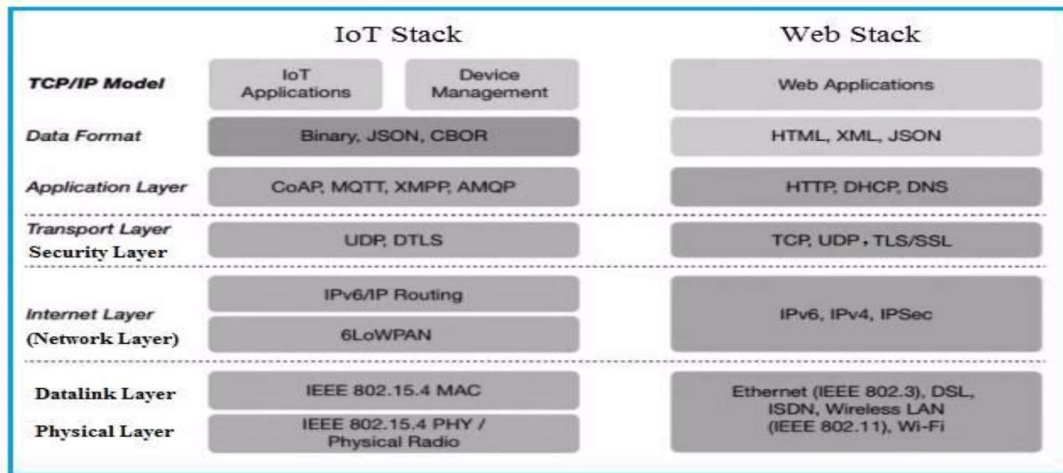
Protocols-MQTT, ZigBee, Bluetooth, CoAP, UDP, TCP

Connectivity Terminologies

The number of IoT connected devices will increase drastically in near future. The reason is the integration of existing devices, smart devices as well as constrained nodes in a singular Framework. Following are the connectivity terminologies used in IoT.

- **IoT Node-** "These are machines, things or computers connected to other nodes inside a LAN via the IoT LAN.
- **IoT LAN-** Local Area Network or LAN is short to medium range, where distances can be up to hundreds of meters, such as home automation or sensors that are installed within a factory production line that communicate over Wi-Fi with a gateway device that is installed within the same building.
- **IoT WAN-** Wide Area Networks or WAN is the connection of various network segments organizationally and geographically wide which connects to the Internet. It may consist of several LANs. Each node in a LAN will have a unique address but another node in a different LAN may have the same address. Different LANs are connected to WAN via gateways.

- **IoT WAN Gateway**- This is a router connecting the IoT LAN to a WAN or to the Internet. The main responsibility of IoT gateways is to forward packets between LAN and WAN on the IP layer.
- **IoT Proxy**-This performs active application layer functions between IoT nodes and other entities. the architecture of IoT Proxy, where LU represents Locally Unique address.



The 3 main network technologies for IoT Connectivity are:

1. Standard Wireless Access – WiFi, 2G, 3G, 4G,5G and standard LTE.
 2. Private Long Range – LoRA based platform, Zigbee, and SigFox.
 3. Mobile IoT Technologies – LTE-M, NB-IoT, and EC-GSM-IoT
- Some of the most common connectivity options available for IoT-enabled devices.

Ethernet

Ethernet is *the traditional technology for connecting devices in a wired local area network (LAN) or wide area network*. Easy to implement, cheap, and high bandwidth; most common IoT connectivity technologies. Ethernet is great when you need a hard-line uninterrupted link to your device.

Another major consideration is that data transmitted over ethernet isn't secure. In using ethernet, your data and device management actions are subject to attack from any other equipment on the same network unless you take specific steps to encrypt your messages. If deploying equipment into a new construction site, then Ethernet can work well.

WiFi

A WiFi connection transmits data via wireless signals, while an Ethernet connection transmits data over cable. No cables are needed to access a WiFi connection, providing greater mobility for users who can connect to a network or the Internet while moving freely around a space. Data is generally encrypted from your device to the WiFi access point but will be sent in the clear over the access point's internet connection unless you add your own IoT system encryption.

Bluetooth

A common option for low-power connected devices – typically battery-powered wearables with a limited connection duration only when the device is being used is Bluetooth. Bluetooth range is typically low in the 3-5m area, and a secondary device is required to bridge the data to an internet connection. Often Bluetooth is bridged through mobile phones, but dedicated gateways are also popular in fixed applications like domestic health monitoring.

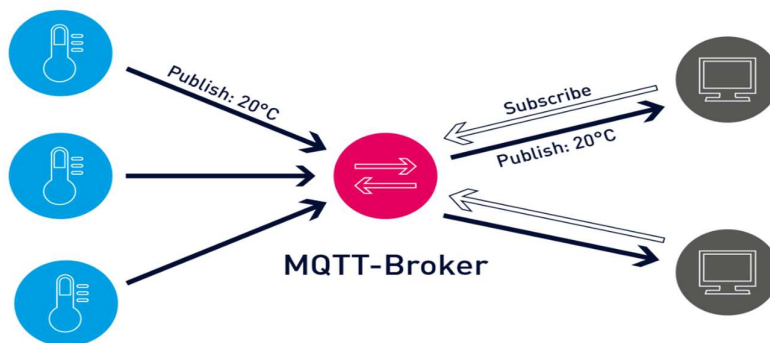
Zigbee

It is a packet-based protocol intended for low-cost, battery-operated devices and products. The Zigbee IoT protocol allows devices to communicate data seamlessly in a variety of network topologies using extremely limited power. Zigbee devices can transmit data over long distances by passing data through a mesh network of intermediate devices to reach more distant ones. Zigbee is typically used in low data rate applications that require long battery life and secure networking. Zigbee uses a mesh network topology, which allows for devices to communicate with each other without the need for a central hub or router. Zigbee is an IEEE 802.15.4-based technology built on the physical layer and media access control defined in that standard. In the industrial and commercial automation space, ZigBee-based devices can handle various functions, from measuring temperature and humidity to tracking assets. For home automation, ZigBee can control lighting, thermostats, and security functions.

MQTT (Message Queuing Telemetry Transport)

MQTT is a messaging protocol for restricted low-bandwidth networks and extremely high-latency IoT devices. It is an ideal **protocol** for machine-to-machine (M2M) communication. In MQTT sender and receiver have no direct connection. MQTT runs on top of TCP/IP using a PUSH/SUBSCRIBE topology. In MQTT architecture, there are two types of systems: clients and brokers. A broker is the server that the clients communicate with. The broker receives communications from clients and sends those communications on to other clients. Clients do not communicate directly with each other, but rather connect to the broker. Each client may be either a publisher, a subscriber, or both.

MQTT is an event-driven protocol. There is no periodic or ongoing data transmission. This keeps transmission to a minimum. A client only publishes when there is information to be sent, and a broker only sends out information to subscribers when new data arrives. In IoT and IIoT, MQTT is used all the way to connecting cloud environments.



CoAP

Constrained Application Protocol (CoAP) is a specialized web transfer protocol for use with constrained nodes and constrained networks with low bandwidth and low availability in the Internet of Things. It is generally used for machine-to-machine (M2M) applications such as smart energy and building automation.

CoAP functions as a sort of [HTTP](#) for restricted devices, enabling equipment such as sensors or actuators to communicate on the IoT. These sensors and actuators are controlled and contribute by passing along their data as part of a system. The protocol is designed for

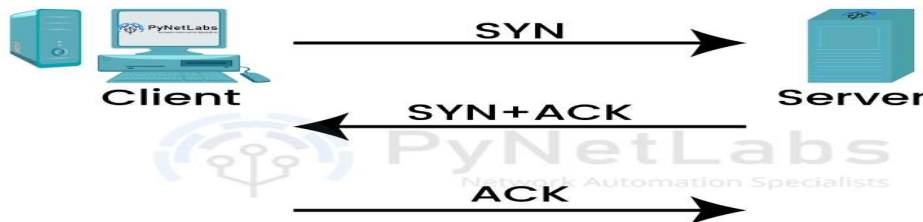
reliability in low bandwidth and high congestion through its low power consumption and low network overhead.

COAP uses [UDP](#) as the underlying network protocol. COAP is basically a client-server IoT protocol where the client makes a request and the server sends back a response as it happens in HTTP. The methods used by COAP are the same used by HTTP.

TCP

TCP, short for Transmission Control Protocol, is a connection-oriented protocol that ensures reliable and ordered delivery of data packets between devices. It provides error detection, flow control, and congestion control mechanisms to guarantee the successful transmission of data.

TCP establishes a connection between a sender and a receiver before transferring data, ensuring that packets arrive in the correct order and without errors. It retransmits lost or corrupted packets and acknowledges the receipt of data, making it highly reliable.



Features of TCP

Here are some features of TCP –

- TCP establishes a reliable connection between the sender and receiver before data exchange.
- TCP detects and corrects errors in transmitted data using checksums and retransmission.
- TCP regulates the flow of data to prevent overwhelming the receiver or congesting the network.
- TCP assigns sequence numbers to packets for proper ordering.
- TCP uses acknowledgments to verify successful receipt of data.

Advantages of TCP

- Reliable data delivery
- Error detection and correction
- Ordered data delivery
- Flow control
- Congestion control
- Connection-oriented communication
- Compatibility
- Versatility
- Error recovery
- Widely adopted

Disadvantages of TCP

- More Overhead
- Limited scalability
- TCP is a more complex protocol compared to UDP
- Inefficient for low-bandwidth networks

Where TCP is used?

TCP (Transmission Control Protocol) is used in various applications and scenarios where reliable and ordered data transmission is essential. Here are some common use cases for TCP:

- **Web browsing:** TCP is used to establish a connection between web browsers and servers.
- **File transfer:** TCP is employed in protocols like FTP (File Transfer Protocol) and SFTP (Secure File Transfer Protocol).
- **Remote access:** TCP is utilized in protocols like SSH (Secure Shell) and Telnet to establish secure and reliable remote connections.
- **VPN (Virtual Private Network):** TCP is often used in VPN implementations to provide secure and reliable communication.
- **IoT (Internet of Things) applications:** TCP is used in various IoT applications where reliable communication and data delivery are crucial.

UDP

User Datagram Protocol is a transport layer protocol used to transmit data. UDP, has less overhead for establishing, maintaining, or terminating a connection; hence, it is faster than TCP. In UDP, the data is continuously sent to the recipient irrespective of whether it was received or not.

UDP can be considered a lightweight protocol because of its lesser responsibilities while delivering data. It is unreliable because there is no acknowledgment after receiving the data successfully, which means the sender won't know if the data was lost during communication or received by the recipient.

Features of UDP

Some of the features of UDP are –

- UDP is a connectionless protocol, meaning it does not establish a dedicated connection before transmitting data.
- UDP has minimal header overhead compared to TCP, making it more lightweight.
- UDP prioritizes speed over reliability, making it suitable for real-time applications that require quick data delivery.
- UDP supports broadcast and [multicast communication](#), allowing efficient data distribution to multiple recipients.

Advantages of UDP

- Low overhead
- High-speed transmission
- UDP has a very simple design
- Reduced network congestion
- Suitable for loss-tolerant applications

Disadvantages of UDP

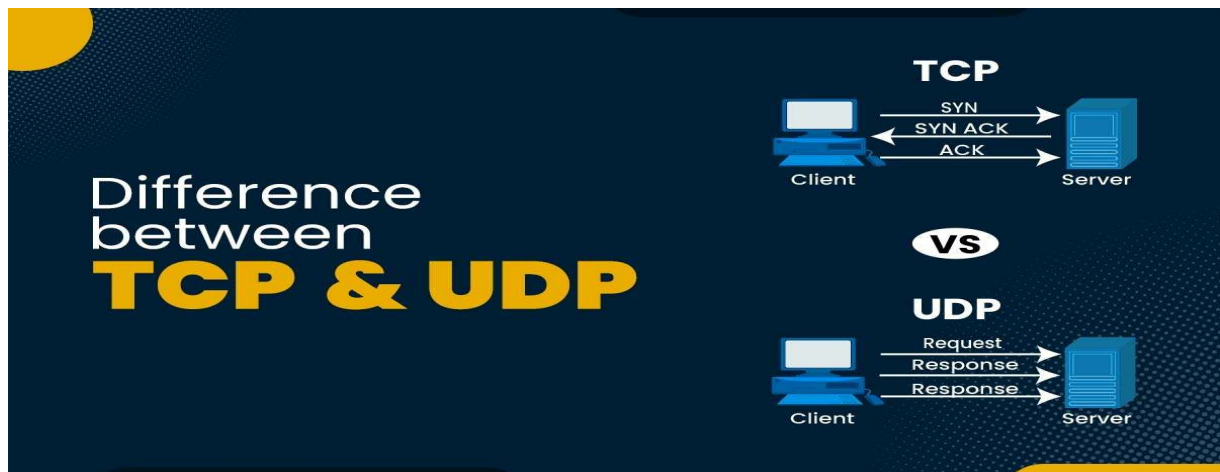
Here are a few disadvantages of using UDP (User Datagram Protocol):

- UDP does not provide built-in mechanisms for reliable data delivery, error correction, or retransmission.
- It does not ensure the ordered delivery of packets.
- UDP does not perform congestion control.
- Vulnerability to packet loss.
- Inefficient for large data transfers.
- Unsuitable for applications requiring guaranteed delivery.

Where UDP is used?

- UDP is widely used in real-time applications, such as video streaming, live broadcasting, online gaming, and voice over IP (VoIP).
- UDP is the primary protocol used for DNS resolution.
- UDP is often used in IoT applications for transmitting data from connected devices to servers or gateways.
- UDP is the underlying protocol for TFTP, a simple file transfer protocol used for bootstrapping network devices or transferring configuration files.
- UDP is used in [DHCP](#) for dynamic IP address assignment.

Difference between TCP and UDP



Here are the top difference between TCP and UDP Protocol –

Parameters	TCP (Transmission Control Protocol)	UDP (User Datagram Protocol)
Connection	Connection-oriented protocol	Connectionless protocol
Reliability	Reliable delivery of data with error detection and correction	Unreliable delivery of data without error detection or correction

Ordering	Guarantees in-order delivery of data	No guarantee of in-order delivery of data
Congestion	Includes congestion control mechanisms	No built-in congestion control mechanisms
Packet Structure	Contains header and payload information	Contains minimal header information
Use Cases	Suitable for applications requiring data integrity and accuracy	Suitable for applications that prioritize speed and efficiency

UNIT 3 : IoT Application Development

3.1 Solution framework for IoT applications-

An IoT framework ensures smooth communication among a network of IoT devices, facilitating seamless data exchange. A typical open-source IoT framework can be divided into four major components: hardware devices, software applications, communication and cloud platforms, and cloud applications.

IoT involves connecting various devices and sensors to the internet, enabling them to communicate, collect data, and perform intelligent actions. Here are some key aspects of an effective IoT solution framework:

1. Device Integration:

- **Definition:** Device integration refers to making disparate programs, data, and devices work seamlessly together.
- **Importance:** The end-to-end IoT solutions, integrating new IoT devices, data, platforms, and applications with existing IT assets (such as business applications, legacy data, mobile apps, and SaaS) is crucial.
- **Advantages:**
 - **Improved Data Accuracy:** Automated data entry reduces the risk of errors during manual data transfer.
 - **Efficient Procedures:** Integration streamlines business processes, especially for field personnel who can directly input data using devices, which then sync automatically to a centralized database.

2. IoT Frameworks:

- **Definition:** An IoT framework provides a structured approach for developing and deploying IoT applications and services.
- **Components:**
 - **Protocols:** Standard communication protocols (e.g., MQTT, CoAP) for device-to-device and device-to-cloud communication.
 - **Tools:** Development tools, libraries, and APIs for building IoT applications.
 - **Standards:** Guidelines and best practices for security, scalability, and interoperability.
- **Benefits:**
 - **Consistency:** Frameworks ensure consistency in design, security, and deployment.
 - **Efficiency:** Developers can focus on application logic rather than low-level details.
 - **Interoperability:** Frameworks promote compatibility across devices and platforms.

3. IoT Development Frameworks:

- **Premium Options:**
 - **Amazon Web Services (AWS) IoT:** A cloud-based service offering scalable infrastructure, device management, and analytics.
 - **Microsoft .NET:** A versatile framework for building Windows-based IoT applications.
- **Open-Source Options:**
 - **Zetta:** An open-source framework for building reactive, real-time IoT applications.

- **Thing Speak:** A cloud-based platform for collecting, analyzing, and visualizing IoT data.
- **Choosing the Right Framework:**
 - Consider factors like scalability, security, ease of development, and community support.

3.2 Implementation of Device integration,

Device integration refers to making disparately designed programs and data operate seamlessly together. In the realm of end-to-end IoT solutions, effective device integration ensures that new IoT devices, IoT data, IoT platforms, and IoT applications work harmoniously alongside existing IT assets (such as business applications, legacy data, mobile apps, and SaaS).

Here are some key aspects of implementing device integration for successful IoT applications:

1. **Improved Data Accuracy:**
 - **Automated Approach:** Device integration automates data entry, reducing the risk of errors during manual data transfer.
 - **Data Validation:** Integrated systems can validate data automatically, ensuring higher-quality information.
 - **[Example:** Imagine field personnel using iPads to collect data. The data syncs automatically to a centralized database, eliminating manual data handling and enhancing accuracy. *Green portion is Not required , for your knowledge only]*
2. **Efficient Procedures:**
 - **Streamlined Processes:** Integration simplifies business processes.
 - For instance, sales representatives or inspectors can directly input data using devices, which then syncs seamlessly to the central system.
 - **Advantages:**
 - **Faster Data Flow:** Real-time data entry speeds up decision-making.
 - **Reduced Administrative Overhead:** Manual data transfer is eliminated.
 - **Enhanced Productivity:** Field personnel can focus on their core tasks rather than data handling.
3. **Hierarchical Address Allocation:**
 - **IPv6:** In the context of IoT, IPv6's hierarchical address allocation facilitates route aggregation across the Internet. This limits the expansion of routing tables.
 - **Benefits:** Efficient routing improves network performance and scalability.
4. **Security Considerations:**
 - **Built-in Security:** IPv6 offers features like data authentication and encryption.
 - **Secure Internet Connections:** Enhanced security ensures safer communication.
 - **Example:** IoT devices transmitting sensitive data can benefit from IPv6's security features.
5. **Simplified Header Format:**
 - **IPv6:** Compared to IPv4, IPv6 has a simpler and more efficient header structure.
 - **Speed and Efficiency:** Streamlined headers contribute to faster internet connections.
 - **Example:** IoT applications with real-time requirements benefit from reduced overhead.

6. Improved Support for Mobile Devices:

- **IPv6:** Designed to handle mobile devices effectively.
- **Quick Connections:** Mobile devices can connect seamlessly using IPv6.
- **Safety:** IPv6 ensures secure communication even on mobile networks.

In summary, implementing device integration involves seamless data flow, efficient processes, and leveraging the benefits of IPv6. By integrating IoT devices effectively, organizations can enhance accuracy, streamline operations, and pave the way for successful IoT applications.

3.3 Data acquisition and integration,

Data acquisition refers to collecting information from various sensors, devices, or sources, while data integration ensures that this data flows seamlessly across different systems and platforms. Here are some key aspects:

1. Device Integration:

- **Definition:** Device integration involves making disparate devices, sensors, and data sources work together seamlessly.
- **Importance:** In IoT solutions, integrating new IoT devices, data streams, and platforms with existing IT systems is crucial.
- **Advantages:**
 - **Improved Data Accuracy:** Automated data entry reduces errors during manual transfer.
 - **Efficient Procedures:** Integration streamlines business processes (e.g., field personnel using iPads to collect data).
 - **Hierarchical Address Allocation:** IPv6 facilitates efficient route aggregation across the Internet, limiting routing table expansion.
 - **Security Considerations:** IPv6 offers built-in security features like data authentication and encryption.
 - **Simplified Header Format:** IPv6 headers are more efficient than IPv4's, enhancing internet speed.
 - **Improved Support for Mobile Devices:** IPv6 handles mobile connections effectively.

2. IoT Frameworks:

- **Choose Wisely:** Select IoT development frameworks based on your project needs.
- **Premium Options:**
 - **Amazon Web Services (AWS) IoT:** Cloud-based service for scalable infrastructure, device management, and analytics.
 - **Microsoft .NET:** Versatile framework for Windows-based IoT applications.
- **Open-Source Options:**
 - **Zetta:** Open-source framework for building real-time IoT applications.
 - **ThingSpeak:** Cloud platform for collecting, analyzing, and visualizing IoT data.
- **Consider Factors:** Evaluate scalability, security, ease of development, and community support.

3. Data Acquisition:

- **Sensor Data:** Collect data from sensors (temperature, humidity, motion, etc.).
- **Protocols:** Use standard communication protocols (e.g., MQTT, CoAP) for data exchange.

- **Edge Computing:** Process data at the edge (near the source) for real-time insights.
- **Data Validation:** Ensure data quality and accuracy during acquisition.
- 4. **Data Integration:**
 - **Data Pipelines:** Set up pipelines to move data from sensors to storage or analytics platforms.
 - **APIs and Connectors:** Use APIs (Application Programming Interface) and connectors to integrate data across systems.
 - **Data Transformation:** Convert data formats (e.g., JSON to CSV) for compatibility.
 - **Real-Time vs. Batch Integration:** Choose based on latency requirements.

In summary, effective data acquisition and integration are essential for successful IoT applications. By seamlessly connecting devices, handling data, and leveraging IPv6, organizations can build robust and efficient IoT solutions

3.4 Device data storage- Unstructured data storage on cloud/local server,

Unstructured Data

Unstructured data doesn't adhere to a fixed data model, yet it holds immense value. Here are some examples:

1. **Social Media Data:**
 - Social media text (comments, feedback) is unstructured.
 - Social media data like friends, followers, and likes are structured.
2. **Email:**
 - The body copy of an email is unstructured.
 - Fields like "to," "cc," and "subject" are structured.
3. **Multimedia:**
 - Multimedia content (vectors, bitmaps, GIFs, frames) can take various forms, making it unstructured.

Challenges in Unstructured Data Storage:

1. **No Fixed Format:**
 - Storing details about employees becomes complex.
 - Each employee may have unique characteristics, and predefined fields might be unused.
 - Schema changes and downtime are problematic.
2. **Scaling Issues:**
 - Traditional storage systems struggle to scale out.
 - Adding more disks increases costs, but data keeps growing.
 - Relational databases face performance issues due to complex table joins across nodes.
3. **Complex Data Retrieval and Querying:**
 - Dumping unstructured data without a plan leads to inefficiency.
 - Finding, updating, or deleting multimedia data becomes challenging.
 - Efficient indexing is crucial.

Storage Requirements for Unstructured Data:

1. **NoSQL Databases:**
 - Provide scalability and flexible schema.
 - Efficiently manage unstructured data.
2. **Data Lakes:**

- Store large volumes of unstructured data.
 - Options include Amazon S3, Google Cloud Storage, and Microsoft Azure Blob Storage.
3. **Data Warehouses:**
- Enable efficient data management.
 - Consider cloud-based solutions for scalability and cost-effectiveness.

Whether choose on-premise or cloud storage, the key lies in seamless data access and efficient management. Unstructured data holds immense potential for innovation and business value.

3.5 Authentication, authorization of devices.

Authentication ensures that devices and machines in the IoT ecosystem can be trusted. It establishes the identity of these entities, protecting data and controlling access over unsecured networks like the Internet. Strong IoT authentication is crucial for the following reasons:

1. **Preventing Unauthorized Control Commands:**
 - Connected IoT devices must be trusted to prevent unauthorized control commands from malicious users or devices.
 - Imagine an attacker claiming to be an IoT device and attempting to access sensitive data like recorded conversations or images.
2. **Methods of IoT Authentication:**
 - There are several methods to achieve strong authentication for secure IoT device communications:
 - **One-way authentication:** Only one party authenticates itself to the other.
 - **Two-way authentication (mutual authentication):** Both entities authenticate each other.
 - **Three-way authentication:** A central authority assists the two parties in authenticating each other.
 - **Distributed authentication:** Direct authentication between communication parties.
 - **Centralized authentication:** A trusted third party manages authentication certificates.
3. **Unique Digital Identity:**
 - Each IoT machine needs a **unique digital identity** when connecting to a gateway or central server.
 - This identity is bound to a cryptographic key unique to each IoT device.
 - For example, trusted platform modules (TPMs) issue registration IDs for secure identification.

Authorization validates the identity of each endpoint in the IoT system. It ensures that only authorized actions and operations occur. Key points about IoT authorization:

1. **Machine Identity Management:**
 - **Machine Identity Management** builds and manages confidence in a machine's identity.
 - It includes both **authentication** (establishing identity) and **authorization** (validating permissions).
2. **Devices Requiring Authorization:**
 - Various devices fall under this umbrella:

- Industrial Control Systems
- Connected medical devices
- Vehicle Engine Control Units (ECUs)
- Security cameras
- Home security systems
- Mobile devices
- Smart speakers, lights, and outlets

3. Unique Digital Identity for Each Device:

- To prevent malicious actors from gaining control, each IoT machine needs a unique digital identity.
- This identity is tied to a cryptographic key specific to that device.

In summary, robust authentication and authorization mechanisms are essential for securing the diverse and interconnected world of IoT

UNIT 4 : Smart Technology

4.1 Understanding the IoT Big Picture

1.

- The **Internet of Things** refers to the interconnected network of physical devices, vehicles, buildings, and other objects that collect and exchange data over the internet.
- These devices are embedded with sensors, software, and connectivity, enabling them to communicate and interact with each other and with centralized systems.

2. Key Components of IoT:

- **Devices and Sensors:**
 - These are the building blocks of IoT. Devices can be anything from smart thermostats and wearables to industrial machinery and agricultural sensors.
 - Sensors capture data such as temperature, humidity, motion, light, and more.
- **Connectivity:**
 - IoT devices rely on various communication protocols to transmit data. These include Wi-Fi, cellular networks, Bluetooth, Zigbee, LoRaWAN, and more.
 - Choosing the right connectivity technology depends on factors like range, power consumption, and data rate.
- **Cloud Services:**
 - IoT generates massive amounts of data. Cloud platforms provide storage, processing, and analytics capabilities.
 - Cloud services allow real-time data processing, predictive analytics, and machine learning.
- **Edge Computing:**
 - Edge devices process data closer to the source (at the edge of the network) rather than sending it all to the cloud.
 - Edge computing reduces latency, improves response time, and minimizes bandwidth usage.
- **Security and Privacy:**
 - IoT security is critical due to the potential impact of breaches.
 - Measures include encryption, secure boot, access control, and regular updates.
- **Applications and Use Cases:**
 - IoT applications span various domains:
 - **Smart Homes:** Connected thermostats, lights, and appliances.
 - **Healthcare:** Wearable health monitors, remote patient monitoring.
 - **Agriculture:** Soil moisture sensors, precision farming.

- **Industrial IoT (IIoT):** Factory automation, predictive maintenance.
 - **Smart Cities:** Traffic management, waste management, energy efficiency.
 - **Transportation:** Fleet management, connected vehicles.
- **Standards and Protocols:**
 - Standardization ensures interoperability among devices and systems.
 - Examples include MQTT, CoAP, and OCF.
- **Data Analytics and Insights:**
 - Extracting meaningful insights from IoT data drives decision-making.
 - Machine learning models predict failures, optimize processes, and enhance efficiency.

In summary, IoT is a vast ecosystem that encompasses hardware, software, connectivity, and applications. It transforms industries, enhances convenience, and shapes our connected future.

4.2 Building the Internet of Things

Internet of Things (IoT)—a realm where everyday objects transform into intelligent, interconnected entities.

1. **Conceptual Foundation:**
 - The IoT is a network of physical devices (often embedded with sensors, actuators, and communication modules) that collect and exchange data over the internet.
 - These devices can be anything: from smart thermostats and wearable fitness trackers to industrial machinery and agricultural sensors.
2. **Key Components:**
 - **Devices (Things):** These are the heart of the IoT. They include sensors (for data collection) and actuators (for control).
 - **Connectivity:** Devices communicate via Wi-Fi, cellular networks, Bluetooth, Zigbee, or other protocols.
 - **Data Processing:** Cloud platforms process and analyze the data generated by devices.
 - **User Interfaces:** Web or mobile apps allow users to interact with IoT systems.
3. **Building Blocks:**
 - **Hardware:** Choose microcontrollers (like Arduino or Raspberry Pi) or specialized IoT boards.
 - **Sensors:** Measure temperature, humidity, light, motion, etc.
 - **Actuators:** Control lights, motors, valves, etc.
 - **Communication Modules:** Wi-Fi, Bluetooth, LoRa, etc.
 - **Power Supply:** Battery or external power source.
 - **Protocols:** MQTT, CoAP, HTTP, etc.
4. **Development Steps:**
 - **1. Define Case:**
 - What problem will IoT solution solve? Identify the purpose.
 - **2. Select Hardware and Sensors:**
 - Based on use case, choose the right components.
 - **3. Write Firmware/Software:**
 - Code the logic for data collection, processing, and communication.
 - **4. Set Up Cloud Services:**

- Use platforms like AWS, Google Cloud, or Azure.
 - **5. Security Measures:**
 - Encrypt data, secure communication, and manage access.
 - **6. User Interface:**
 - Create a dashboard or app to visualize data.
 - **7. Test and Deploy:**
 - Ensure everything works as expected.
 - **8. Monitor and Maintain:**
 - Regularly update firmware and monitor performance.
- 5. Challenges and Considerations:**
- **Scalability:** Handling millions of devices efficiently.
 - **Interoperability:** Ensuring devices from different manufacturers can work together.
 - **Privacy:** Balancing data collection with user privacy.
 - **Energy Efficiency:** Prolonging device battery life.
 - **Reliability:** IoT systems must be robust and resilient.

4.3 Understanding Smart Devices, Building Blocks

Four things form basic building blocks of the IoT system –sensors, processors, gateways, applications. Each of these nodes has to have its own characteristics in order to form an useful IoT system.

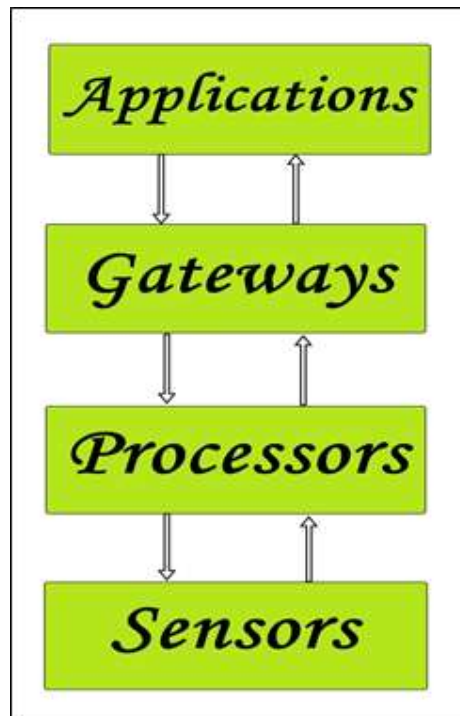


Figure 1: Simplified block diagram of the basic building blocks of the IoT

Sensors:

- These form the front end of the IoT devices. These are the so-called “Things” of the system. Their main purpose is to collect data from its surroundings (sensors) or give out data to its surrounding (actuators).
- These have to be uniquely identifiable devices with a unique IP address so that they can be easily identifiable over a large network.
- These have to be active in nature which means that they should be able to collect real-time data.
- Examples of sensors are gas sensor, water quality sensor, moisture sensor, etc.

Processors:

- Processors are the brain of the IoT system. Their main function is to process the data captured by the sensors and process them so as to extract the valuable data from the enormous amount of raw data collected.
- Processors mostly work on real-time basis and can be easily controlled by applications. These are also responsible for securing the data – that is performing encryption and decryption of data.
- Embedded hardware devices, microcontroller, etc are the ones that process the data because they have processors attached to it.

Gateways:

- Gateways are responsible for routing the processed data and send it to proper locations for its (data) proper utilization.
- In other words, we can say that gateway helps in to and fro communication of the data. It provides network connectivity to the data. Network connectivity is essential for any IoT system to communicate.
- LAN, WAN, PAN, etc are examples of network gateways.

Applications:

- Applications form another end of an IoT system. Applications are essential for proper utilization of all the data collected.
- These cloud-based applications which are responsible for rendering the effective meaning to the data collected. Applications are controlled by users and are a delivery point of particular services.
- Examples of applications are home automation apps, security systems, industrial control hub, etc.

In Figure 2, the extreme right block forms the application end of the IoT system.

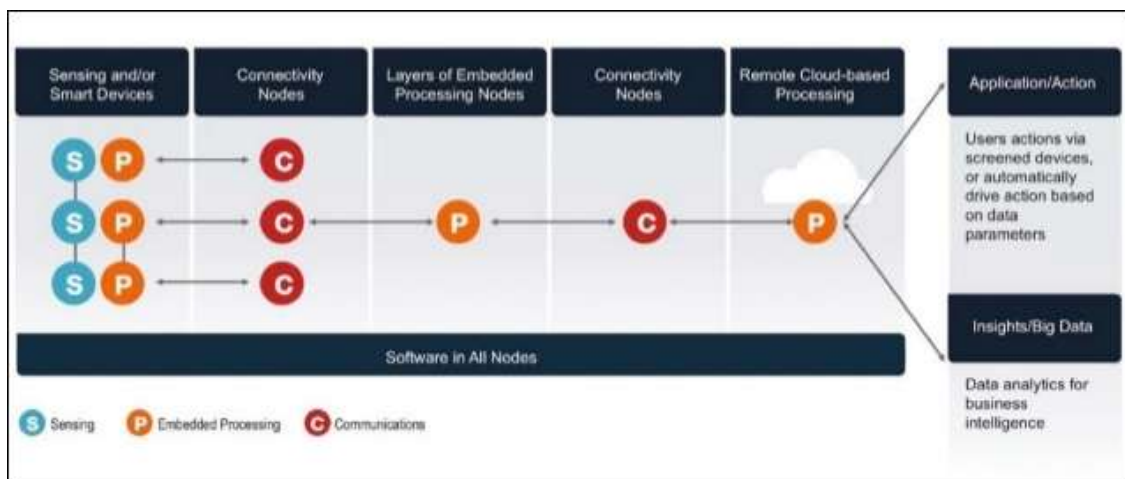


Figure 2: Basic building blocks of IoT

From the figure it is evident that the information gathered by the sensing node (end node) is processed first then via connectivity it reaches the embedded processing nodes that can be any embedded hardware devices and are processed there as well. It then passes through the connectivity nodes again and reaches the remote cloud-based processing that can be any software and is sent to the application node for the proper applied usage of the data collected and also for data analysis via big data.

4.4 Understanding Network Connections

Sensor, Connectivity and Network Layer

- This layer consists of RFID tags, sensors (responsible for collecting raw data). These form the essential “things” of an IoT system.
- Sensors, RFID tags are wireless devices and form the Wireless Sensor Networks (WSN).
- Sensors are active in nature which means that real-time information is to be collected and processed.
- This layer also has the network connectivity (like WAN, PAN, etc.) which is responsible for communicating the raw data to the next layer which is the Gateway and Network Layer.
- The devices which are comprised of WSN have finite storage capacity, restricted communication bandwidth and have small processing speed.

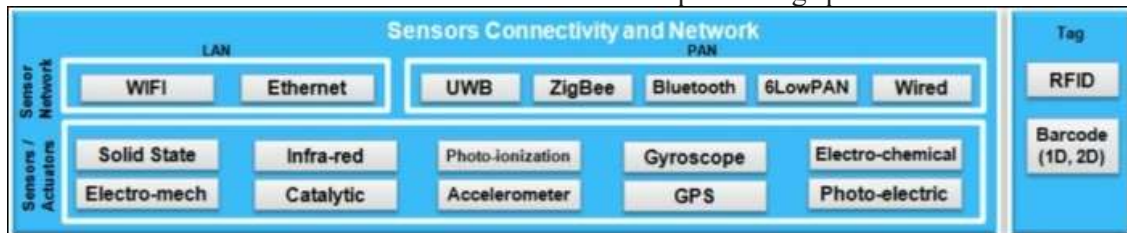


Figure 5: Sensor, Connectivity and Network Layer

Gateway and Network Layer

- Gateways are responsible for routing the data coming from the **Sensor, Connectivity and Network layer** and pass it to the next layer which is the **Management Service Layer**.
- This layer requires having a large storage capacity for storing the enormous amount of data collected by the sensors, RFID tags, etc. Also, this layer needs to have a consistently trusted performance in terms of public, private and hybrid networks.
- Different IoT device works on different kinds of network protocols. All these protocols are required to be assimilated into a single layer. This layer is responsible for integrating various network protocols.

From the figure below, at the bottom, the gateway which is comprised of the embedded OS, Signal Processors, and Modulators, Micro-Controllers etc. Above the gateway we have the Gateway Networks which are LAN(Local Area Network), WAN(Wide Area Network), etc.

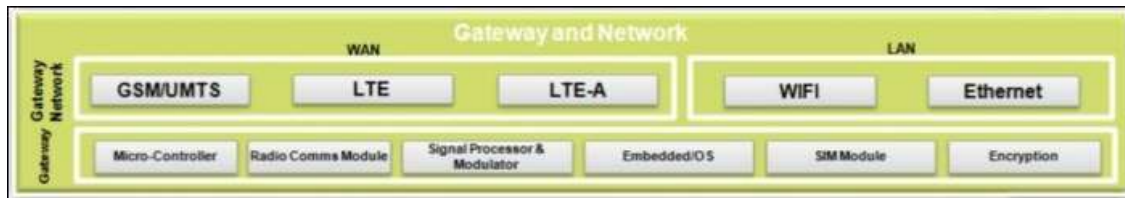


Figure 6: Gateway and Network Layer

Management Service Layer

- This layer is used for managing IoT services. The management Service layer is responsible for Securing Analysis of IoT devices, Analysis of Information (Stream Analytics, Data Analytics), Device Management.
- Data management is required to extract the necessary information from the enormous amount of raw data collected by the sensor devices to yield a valuable result of all the data collected. This action is performed in this layer.
- This layer is also responsible for data mining, text mining, service analytics, etc.

Also, from the figure, there are IoT/M2M Application Services which includes Analytics Platform; Data – which is the most important part; Security which includes Access Controls, Encryption, Identity Access Management, etc. ; and then we have the Business Rule Management (BRM) and Business Process Management (BPM).



Figure 7: Management Service Layer

Application Layer

- Application layer forms the topmost layer of IoT architecture which is responsible for effective utilization of the data collected.
- Various IoT applications include Home Automation, E-health, E-Government, etc.
- From the figure below, there are two types of applications which are Horizontal Market which includes Fleet Management, Supply Chain, etc. and on the Sector-wise application of IoT we have energy, healthcare, transportation, etc.



Figure 8: Application Layer

4.5 Understanding IP Addresses

What Is an IP Address?

An Internet Protocol address is a unique identifier for devices connected to a network. The unique identifier makes it possible for devices to find and communicate with each other. Initially, the main types of devices that required an IP address included network devices, such as computers, servers, routers, and printers. However, with the Internet of Things, the list includes cell phones, televisions, refrigerators, automobiles, light bulbs, or anything else capable of receiving or exchanging information over a network.

An **IP address** uniquely identifies a device on a network. [these addresses before; they look something like 192.168.1.34.] Here are some key points about IP addresses:

1. **Format:**
 - An IP address consists of **four numbers**, each ranging from **0 to 255**.
 - The full IP addressing range goes from 0.0.0.0 to 255.255.255.255.
 - Each number is actually an **eight-digit binary number** (also called an octet). For
2. **Network ID and Host ID:**
 - A device's IP address consists of two separate parts:
 - **Network ID:** The leftmost part of the IP address identifies the specific network on which the device is located.
 - **Host ID:** The rightmost part specifies the unique address of the device within that network.
 - For example, in the IP address 192.168.1.34, the 192.168.1 part is the network ID, and 34 is the host ID.
3. **Why Should You Care?:**
 - Understanding IP addresses is crucial for troubleshooting network issues.
 - It's essential if you want to set up more advanced tasks like hosting game servers or media servers accessible over the internet.

In summary, IP addresses play a fundamental role in networking, allowing devices to communicate and find each other on the vast digital

IPv4:

IP version four addresses are 32-bit integers which will be expressed in dotted decimal notation. Example- 192.0.2.126 could be an IPv4 address.

Characteristics of IPv4

IPv4 could be a 32-Bit IP Address.

IPv4 could be a numeric address, and its bits are separated by a dot.

The number of header fields is twelve and the length of the header field is twenty.

It has Unicast, broadcast, and multicast style of addresses.

IPv4 supports VLSM (Virtual Length Subnet Mask).

IPv4 uses the Post Address Resolution Protocol to map to the MAC address.

Packet fragmentation permits from routers and causing host.

IPv4 Datagram Header

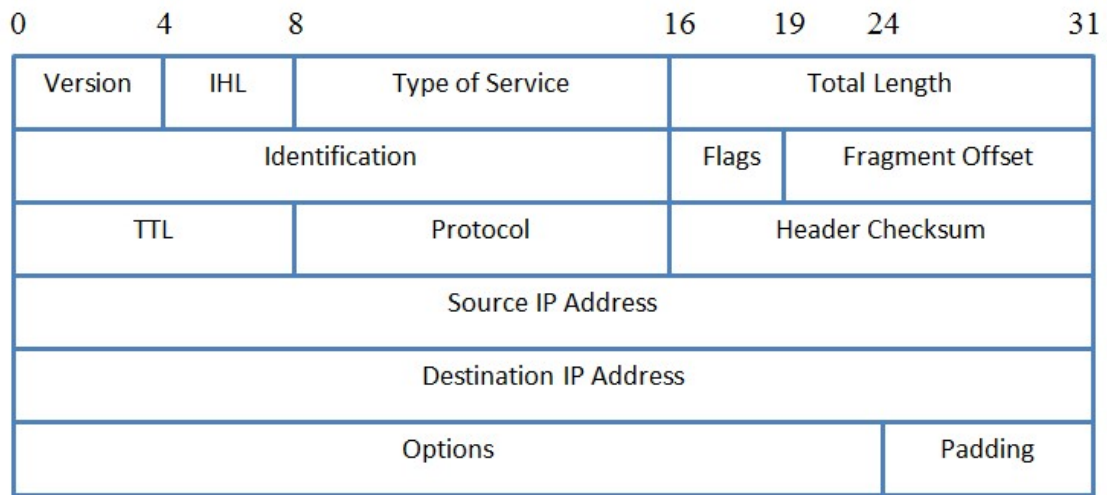


Fig: IPv4 Frame Format

Version:

This field indicates the version number of the IP packet so that the revised version can be distinguished from the previous version. The current IP version is 4.

Internet Header Length (IHL):

It specifies the length of the IP header in unit 32 bits.

Type of Service: This field specifies the priority of the packets based on delay, throughput, reliability and cost requirements.

Total Length:

This field specifies the number of bytes of the IP packet including header and data. As 16 bits are assigned to this field, the maximum length of the packet is 65535 bytes.

Identification:

The identification field is used to identify which packet a particular fragment belongs to so that fragments for different packets don't get mixed up.

Flags:

The flag field has three bits:

1. Unused bit
2. Don't fragment (DF) bit
3. More fragment (MF) bit

Fragment Offset:

The fragment offset field identifies the location of the fragment in a packet. The value measures the offset in a unit of 8 bytes, between the beginning of the packet to be fragmented and the beginning of the fragment.

Time to live (TTL):

This field is used to indicate the amount of time in seconds a packet is allowed to remain in the network.

Protocol:

This field specifies the protocol that is to receive the IP data at the destination host.

Header Checksum:

This field verifies the integrity of the header of the IP packet. The integrity of the data part is left to the upper layer protocols. The checksum is generated by the source and it is sent along with the frame header to the next router.

Source IP address & Destination IP address:

These two fields contain the IP addresses of the source and destination hosts respectively.

Options:

Options fields are rarely used to include special features such as security level, the route to be taken and time stamp at each router. It is used in RSVP.

Padding:

This field is used to make the header a multiple of 32-bit words.

- **Advantages of IPv4:**

- **Security:** Supports encryption for privacy.
- **Network allocation:** Over 85,000 practical routers.
- **Connectivity:** Easy to connect multiple devices across large networks.
- **Quality service:** Efficient data transfer.
- **Scalable routing:** Addresses are managed effectively.

- **Limitations of IPv4:**

- **Address exhaustion:** The world's supply of unique IP addresses is dwindling.
- **Inefficient internet routing.**
- **High system management costs.**
- **Security challenges.**
- **Difficult to support future needs.**

In summary, IPv4 has been the workhorse of internet communication, but its limitations led to the development of **IPv6** as a successor.

IPv6

Internet Protocol version 6 (IPv6) is also known as **Internet Protocol next generation (IPng)**. It also accommodates more feature to meet the global requirement of growing Internet. To allocate a sufficient number of network address, IPv6 allows 128 bits of IP address separated into 8 sections of 2 bytes each. Unlike IPv4 where the address is represented as dotted-decimal notation, IPv6 uses hexadecimal numbers and colon (":") is used as a delimiter between the sections.

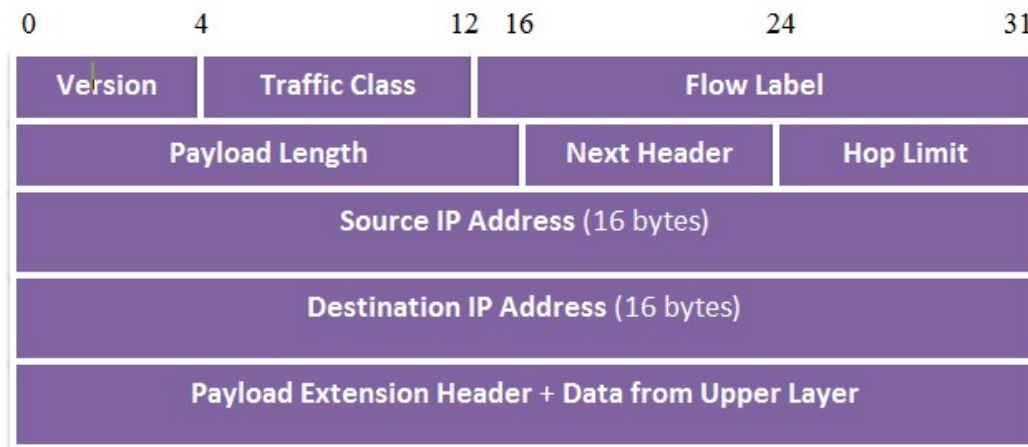


Fig: IPv6 Packet Format

Example: IPv6 address may be like this :

FA20:B120:6230:0000:0000:CE12:0006:ABDF

Version: This field is 4 bits long and it defines the version of the IP packet. The value of it for IPv6 is 6 and IPv4 its value is 4. During the transition period from IPv4 to IPv6, the routers will be able to distinguish the two versions of the IP packets.

Traffic Class: This field is 20 bits long and it is used to distinguish between the different requirements for real-time delivery services.

Flow Label: This field is 20 bits long and it is used to allow the source and destination nodes to set up a pseudo connection with particular properties and requirements. It is designed to provide special handling of a particular flow of data.

Payload Length: It is of 2 bytes length and signifies the number of bytes that follow the 40 bytes base header. It is the length of the IP datagram excluding the base header.

Next Header: This field is of 1 byte length and it defines one of the extension headers that follow the base header. The extension headers also contain this field to indicate the next header. if this is the last IP header then Next header field tells which of the transport protocols (TCP or UDP) the packet is to be passed.

Hop Limit: This field contains 1 byte and it signifies the maximum number of hops a packet can travel. The time to live field in the IPv4 header did the same task, except that in IPv4 it was counted in time and in IPv6 it is counted in terms of the number of routers.

Source Address: It is 16 bytes long and contains the IP address of the source machine to the network interface.

Destination Address: It is 16 bytes long and usually contains the IP address of the ultimate destination machine to the network interface. In case of specific routing, it may contain the IP address of the next router.

Extension Header: Some of the fields Ipv4 that are missing in IPv6 is necessary in some of the cases. To handle this problem, IPv6 has introduced the concept of the extension header. There are be one or more of the six possible extension headers. These headers appear directly after the base header.

1. **IPv4 vs. IPv6:**

- **IPv4** (Internet Protocol version 4) has been the dominant protocol for decades. However, it faces challenges due to the limited address space it provides. IPv4 uses 32-bit addresses, resulting in approximately 4.3 billion unique addresses.
- **IPv6**, on the other hand, was developed by the **Internet Engineering Task Force (IETF)** to overcome these limitations. It uses **128-bit addresses**, theoretically allowing for a staggering 2^{128} (about 3.4×10^{38}) total addresses. this means an abundance of addresses to accommodate the growing number of internet-connected devices.s¹².

2. **Key Features of IPv6:**

- **Larger Address Space:** With 128-bit addresses, IPv6 ensures that we won't run out of unique IP addresses anytime soon.
- **Hierarchical Address Allocation:** IPv6 facilitates efficient route aggregation across the Internet, limiting the expansion of routing tables.
- **Multicast Addressing:** IPv6 expands and simplifies multicast addressing, optimizing service delivery.
- **Improved Security and Mobility:** IPv6 considers device mobility, security, and configuration aspects in its design.
- **Hexadecimal Representation:** IPv6 addresses are represented as eight groups of four hexadecimal digits each, separated by colons (e.g., 2001:db8::8a2e:370:7334)

3. **Benefits of IPv6:**

- **Larger Address Space:** IPv6 provides a significantly larger address pool than IPv4, which is crucial for accommodating the growing number of internet-connected devices.
- **Improved Security:** IPv6 includes built-in security features such as data authentication and encryption, enhancing overall internet safety.
- **Simplified Header Format:** IPv6's header structure is simpler and more efficient than IPv4's, contributing to faster internet connections.
- **Prioritization (QoS):** IPv6 offers stronger support for Quality of Service (QoS) features, improving traffic management and audio/video quality on websites.
- **Better Mobile Device Support:** IPv6 is designed to handle mobile devices more effectively than IPv4

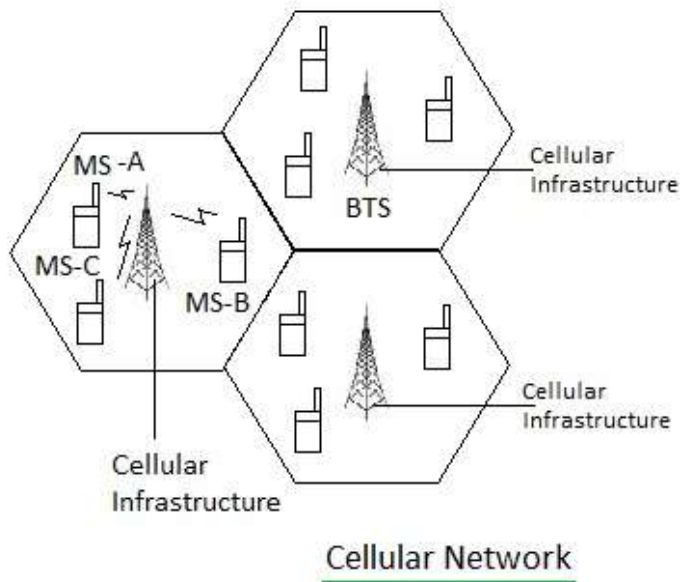
4. **Interoperability:**

- IPv4 and IPv6 are **not directly interoperable**. They cannot communicate directly due to their different address formats. However, transition mechanisms exist to facilitate communication between them.

4.6 Understanding cellular Network & Mesh Network

A **Cellular network or Mobile network** is a radio network distributed over land areas called cells. Each cell served by at least one fixed-location transceiver, known as a cell site or base station. In a cellular network, each cell uses a different set of frequencies from neighboring cells, to avoid interference and provide guaranteed bandwidth within each cell. These base stations provide the cell with the network coverage which can be

used for transmission of voice, data, and other types of content. A cell typically uses a different set of frequencies from neighboring cells, to avoid interference and provide guaranteed service quality within each cell.



The figure-1 depicts cellular network. BSs are interfaced as per various configurations such as star, mesh etc. They are interfaced with PSTN (Public Switched Telephone Network) , MSCs (Mobile Switching Centres) and PSDN (Public Switched Data Network) in the backhaul. The common cellular networks are GSM, CDMA, LTE, 5G etc.

Cellular Technology characteristics:

- **Cellular networks** are widely used for communication. In this technology:
 - Each device communicates through the cellular network infrastructure.
 - Devices do not directly communicate with each other.
 - Communication happens via base stations (cell towers) that relay data to and from the devices.
 - Cellular networks are suitable when there are relatively **few devices per location** .
 - These networks are great for scenarios where devices primarily talk to the cloud and have little need to communicate with each other.
 - **Cellular coverage** must be available, and communication tends to be short and focused.

Mesh Networking:

Mesh networks are made up of a series of interconnected nodes. These nodes are wireless radio devices that communicate with one another. A Mesh IoT network is a local network topology

where devices are connected directly in a non-hierarchical way to route data across the network. The devices in a mesh network communicate according to a predefined protocol that allows each device to participate in the data transmission on the network. Mesh networks are usually, used for IoT applications such as industrial automation and smart metering. All the devices in a mesh topology work together to distribute data across the different components in the network.

- **Mesh networking** characteristics:

There are three major components in an IoT Mesh Network: the gateway, the repeater, and the endpoints. Gateways are devices that connect the mesh network to the internet. Repeaters capture data in the network and reiterate it to the endpoints. Endpoints are devices that receive data from repeaters. They don't need to network with other elements in a mesh network, and so can remain passive if unused

- It allows to extend the reach of device network by enabling data to hop from device to device.
- Radios that are too far apart to hear each other directly can still communicate via intermediate devices.
- Devices form a self-healing mesh, where they relay data to their local neighbors. This means the network reconfigures if a path breaks. Hence, the entire network remains stable because the algorithms always use the only available nodes.
- Mesh networks are beneficial when you have many devices in the same location or when cellular coverage is not available.
- Communication between devices is frequent, and they interact locally.
- Automated joining and self-healing mechanisms help maintain network stability.
- For instance, mesh networking is commonly used in municipal street lighting control.

- **Mesh Networking:**

- **Advantages:**
 - Extends coverage where cellular signals are weak or absent.
 - Ideal for dense device deployments.
 - Frequent local communication.
- **Disadvantages:**
 - Complex setup and management.
 - Limited range compared to cellular.

- **Cellular Networking:**

- **Advantages:**
 - Wider coverage.
 - Simplicity and reliability.
 - Cloud-centric communication.
- **Disadvantages:**
 - Requires cellular coverage.
 - Limited local communication between devices.

UNIT 5. Smart TVs: Viewing in a Connected World

5.1 What is Smart TV & its use

A smart TV, also called a connected TV or CTV, is a television set with interactive, web-enabled features. It is equipped with an operating system (like Android TV, WebOS, Tizen, etc.) and can connect to the internet, allowing it to run apps and stream content without the need for external devices.

1. Streaming Content: Watch online content from apps like Netflix, YouTube, Amazon Prime, etc.
2. Internet Connectivity: Surf the internet directly from the TV using a browser.
3. Home Automation Hub: Acts as a control center for other smart devices like lights, thermostats, or security cameras.
4. Voice Control: Many Smart TVs support voice assistants like Google Assistant or Alexa.
5. Remote Access: Users can control the TV using a smartphone or tablet via mobile apps.
6. Personalization: Recommends content based on user behavior and viewing history.

5.2 What is inside Smart TV

A Smart TV combines traditional television features with advanced computing capabilities and internet connectivity. The internal components that make this possible include:

1. System-on-Chip (SoC):
 - o Acts as the "brain" of the TV.
 - o Integrates the CPU (processor), GPU (graphics processor), RAM, and storage.
 - o Runs the operating system and apps.
2. Operating System (OS):
 - o Examples: Android TV, WebOS, Tizen, Roku OS.
 - o Provides a user interface and manages apps and settings.
3. Wi-Fi / Ethernet Module:
 - o Enables the Smart TV to connect to the internet.
 - o Allows streaming, browsing, software updates, and communication with IoT devices.
4. Memory and Storage:
 - o RAM: Temporary memory to run applications.
 - o Storage: For saving apps, settings, and updates.
5. Display Panel (LED/OLED/QLED):
 - o The screen that shows video content.
 - o Comes with built-in backlighting and color processing technology.
6. Speakers and Audio Processing Unit:
 - o Provides built-in sound output.
 - o Supports audio enhancements like Dolby Digital or DTS.
7. Input/Output Ports:
 - o HDMI, USB, AV, Optical Audio, LAN, etc.
 - o Allows connection with external devices like gaming consoles, USB drives, soundbars.
8. Remote Control Receiver & Voice Recognition Module:
 - o Infrared or Bluetooth receivers.
 - o Some models include voice assistant support (Google Assistant, Alexa).
9. Sensors (optional):
 - o Light sensors for automatic brightness adjustment.
 - o Motion sensors for smart control and power saving.

5.3 What a Smart TV does

A Smart TV is a television set integrated with internet connectivity and interactive features that allow users to stream content, browse the web, and run apps.

Functions of a Smart TV:

1. Internet connectivity: Connects to the internet via Wi-Fi or Ethernet for online access.
2. Streaming Services: Access platforms like Netflix, YouTube, Amazon Prime, and Hotstar directly.
3. Web Browsing: Allows browsing websites using a built-in web browser.
4. Apps and Games: Supports downloading and running apps from app stores (e.g., Google Play).
5. Voice Control: Some Smart TVs have built-in voice assistants (Google Assistant, Alexa).
6. Casting and Screen Mirroring: Lets users cast or mirror content from smartphones or laptops.
7. Software Updates: Regular updates for performance, new features, and security.
8. Smart Home Integration: Can be part of an IoT-based smart home system (control lights, appliances, etc.).
9. Media Playback: Plays content from USB drives or network-shared media libraries.

5.4 Smart TV Operating Systems

Smart TV OSs have revolutionized home entertainment by merging computing with television. A **Smart TV Operating System (OS)** is the software layer that runs on smart televisions, allowing users to interact with the TV, run applications, browse the internet, stream videos, and access other smart features. It is similar to the operating system on a smartphone or computer.

Functions of a Smart TV Operating System

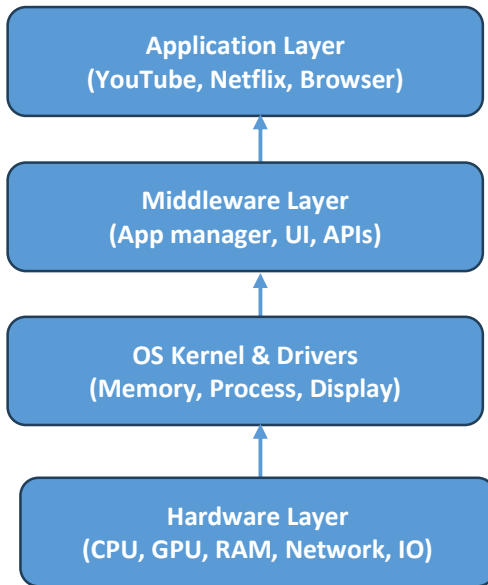
- Controls the hardware (e.g., CPU, RAM, display).
- Provides a user interface (UI) for easy interaction.
- Manages applications like YouTube, Netflix, or Amazon Prime Video.
- Enables internet connectivity (Wi-Fi/Ethernet).
- Supports voice control, screen mirroring, and smart home integration.
- Handles media playback, settings, and notifications.

Common Smart TV Operating Systems:

1. **Android TV**
 - Developed by Google.
 - Supports Google Play Store apps.
 - Built-in Google Assistant and Chromecast.
 - Used in Sony, Xiaomi, and TCL TVs.
2. **WebOS**
 - Developed by LG.
 - Known for its clean interface and fast navigation.
 - Supports multitasking and LG App Store.
3. **Tizen OS**
 - Developed by Samsung.
 - Lightweight and fast.
 - Good app ecosystem and smart home integration.
4. **Roku TV OS**
 - Found in TCL, Hisense, and Sharp TVs.
 - Simple interface, large app library.
 - Strong in the North American market.
5. **Fire TV OS**
 - Developed by Amazon, based on Android.
 - Integrates Amazon Prime content.
 - Alexa voice assistant built-in.
6. **VIDAA**
 - Used in Hisense Smart TVs.

- Simple interface, focused on core streaming apps.

Smart TV OS Architecture block diagram



Comparison Table

Feature	Android TV	WebOS	Tizen OS	Roku OS	Fire TV OS
App Store	Google Play	LG Content Store	Samsung App Hub	Roku Channel	Amazon Appstore
Voice Assistant	Google Assistant	LG ThinQ	Bixby	Roku Voice	Alexa
Casting Support	Chromecast	Miracast	Smart View	Screen Mirroring	Firecast
IoT Integration	Google Home	LG SmartThings	Samsung SmartThings	Limited	Alexa IoT
Customizability	High	Medium	Medium	Low	Medium

5.5 What is Smart TV Set-Top Devices

A Smart TV Set-Top Device is an external hardware unit that connects to a regular (non-smart) television and enables it to access internet-based services and apps, similar to a Smart TV. These devices allow users to stream online content (such as Netflix, YouTube, Amazon Prime Video), browse the internet, play games, and even control smart home devices. They typically connect to the TV via an HDMI port and to the internet via Wi-Fi or Ethernet.

Technical Components and Functionality:

Component	Description
Processor (CPU & GPU)	Executes application logic, video decoding (e.g., H.264/H.265), and UI rendering.
Operating System	Lightweight embedded OS such as Android TV , Fire OS , or tvOS , enabling app management and hardware abstraction.
RAM & Storage	Temporary and permanent memory to run and store streaming apps, system files, and user data.
Wi-Fi/Ethernet Module	Provides internet access to stream content and update firmware.
HDMI Output	Interface to transmit high-definition video and audio signals to the TV.
USB/Bluetooth	For connecting peripherals such as game controllers, keyboards, or USB drives.

Features:

- **Streaming support:** Apps like Netflix, Prime Video, YouTube.
- **Voice Assistant Integration:** Google Assistant, Alexa.
- **Cloud Connectivity:** Sync with cloud media libraries and services.
- **App Store Access:** Download and manage apps from Google Play or proprietary stores.
- **IoT Hub Capability (optional):** Some advanced set-top devices can function as IoT control hubs (e.g., Amazon Echo integration).
-

How it works:

- Connects to the TV using an HDMI or AV cable.
- Connects to the internet using Wi-Fi or Ethernet.
- Controlled using a remote, a smartphone app, or voice commands.

Common Smart TV Set-Top Devices:

- Amazon Fire TV Stick
- Google Chromecast
- Apple TV
- Roku

5.6 Integrating Smart TV in to IOT

Smart TV Integration into the Internet of Things (IoT) refers to connecting the Smart TV with other internet-enabled devices to create a unified, automated, and intelligent home or office environment. Smart TVs are not a standalone entertainment devices only, it acts as **IoT nodes** capable of communicating with smartphones, smart speakers, home automation systems, and cloud services over the internet.

How Integration Happens:

Element	Role in IoT Integration
Wi-Fi/Ethernet	Provides internet connectivity for cloud communication and local device interaction.
Smart Home Protocols (e.g., MQTT, HTTP, DLNA, Chromecast)	Enable device-to-device and app communication.
Voice Assistants (e.g., Alexa, Google Assistant)	Allow the Smart TV to act as a voice-controlled IoT hub.

Element	Role in IoT Integration
Mobile App Connectivity	Smartphone apps sync with the TV for remote control and streaming.
Cloud Services	Store preferences, content, and enable real-time interaction with other IoT devices.

Examples of IoT Use Cases with Smart TV:

- Home Automation Dashboard: Use the Smart TV to monitor and control smart lights, thermostats, door cameras, etc.
- Voice-Controlled Entertainment: "Play music on YouTube" using voice commands through Alexa.
- Security Alerts: Display real-time CCTV footage or notifications from doorbell cameras.
- Energy Monitoring: Smart TV receives updates from smart plugs or sensors about energy consumption.

Benefits of IoT Integration:

- Centralized control interface.
- Enhanced user experience through automation.
- Real-time information sharing.
- Increased convenience and accessibility.

UNIT 6: IoT Case Studies



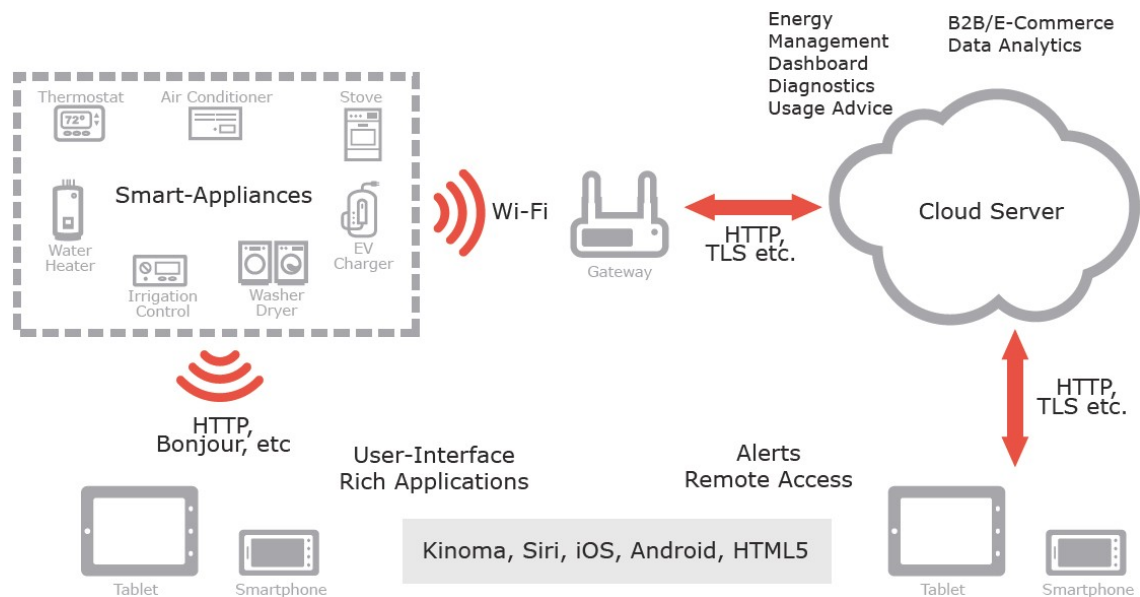
IoT Applications:

☐ Wearables

- Smart Home Applications
- Smart Buildings
- Smart Infrastructure
- Securities
- Health Care
- Smart Cities
- Agriculture
- Industrial Automation

IoT Applications: Smart Home, Smart Buildings and Infrastructure

- IoT home automation is the ability to control domestic appliances by electronically controlled, internet-connected systems. It may include setting complex heating and lighting systems in advance and setting alarms and home security controls, all connected by a central hub and remote-controlled by a mobile app.



6.1b : SMART CITIES

The “Smart City” concept has become extremely popular in scientific literature and international policies. This concept essentially harnesses a plethora of IT innovations hitting us at breathtaking speed to make cities smarter for the citizens. Cities and urban areas comprise about half of the total world’s population . The urban population inflation for the last few decades has been adversely affecting quantity and quality of services provided to the citizens. Smart cities aim at providing effective solutions. Various Smart City (SC) initiatives by both government and private sector organizations have resulted in deployment of Information and Communication Technologies (ICT) to find sustainable efficient and effective solutions to the growing list of challenges facing cities

The core components of a smart city

A modern city is a complex entity, comprising of a range of facets that support the human lifecycle. From transport to administrative services, a range of spheres transformed by smart city technologies includes the following:

Smart manufacturing

An umbrella term for digital production operations, smart manufacturing harnesses IoT and the digital workforce to boost operational and energy efficiency, enhance employee security, and decrease environmental pollution levels.

Smart transportation

Smart transportation means using digital tools and systems to improve and innovate the urban transportation experience, resolve traffic issues such as traffic jams, and reduce the number of accidents.

Smart energy systems

Some of the most enticing smart cities opportunities include digital systems for sustainable and renewable energy. For example, smart power plants using solar or wind energy may become an integral part of a smart city’s ecosystem.

Smart healthcare

Smart healthcare is a mix of technologies aimed at increasing longevity and improving the quality of life of its citizens. Smart healthcare systems leverage mobile, IoT wearables, and computer technology to obtain accurate diagnoses and improve healthcare services.

Smart buildings

Smart buildings use a complex combination of technologies and services that ensure energy efficiency, enhance security, and deliver better communal services. From wireless technologies to IoT devices, intelligent building systems help manage and control lighting, ventilation, air conditioning, basically the entire infrastructure of a modern building.

Implementing smart building technologies results in improved security and health levels, as well as enhanced convenience of its residents.

Digital citizenship

Smart city technologies are interactive and primarily targeted at citizens who are active users of digital devices and services. Digital citizenship implies using digital technologies to partake in local, social, political, and government activities.

Digital government

Also known as e-government, digital government aims to facilitate access to public services by delivering them directly to a citizen's laptop or a smartphone screen. A digital government promotes communication and engagement between citizens and states, reduces the cost of public services and delivers them in a faster and more secure way.

Smart farming

Smart farming is a set of digital technologies used in agriculture to increase the quality and quantity of crops and agricultural products. Smart technologies like IoT, soil scanning, GPS, and data management are also helping reduce the negative impact of farming on the environment.

Open data

To fully embrace infrastructural innovations, smart cities need a transparent culture that will grant open access to data related to just about every facet of a smart city's infrastructure: from info on new children's playgrounds to details on government tenders.

The open data concept also implies that all data should be available in convenient readable formats and open for processing and analysis. By accessing open data, businesses can introduce new services and solutions for the benefit of the city.

Many characteristics define a smart city; they can vary according to the social context in which the city is located or other variables related to culture. The three most important ones are:

Infrastructure Development

A smart city prioritises the optimal development of infrastructure in order to **enhance economy, and social, cultural and urban development**. This is the reason why it improves communication channels so that services like housing, entertainment, telecommunications, business, among others, can be connected using advanced technologies that allow a city to grow and develop

Strategies to Create a Competitive Environment

Through **Information and Communication Technologies (ICT)** and planning, smart cities seek to create a competitive environment in the sector so as to expand urban sectors, thereby enhancing the development of new businesses and improving the city's socio-economic performance.

Inclusive and Sustainable Cities

A smart city's main strategic element will be sustainability so as to look for participation drivers, create better consumption habits and better energy management, and use renewable energies for the preservation of natural resources and the environmental care.

Other Characteristics of smart cities

Effective utilization of data

A smart city should ideally gather huge volumes of data to be analyzed rapidly to offer useful information to residents. You can install open data portals to publish online city data and this data can be accessed and used for predictive analysis to identify future models.

Solid waste disposal system

As population multiplies, the volume of waste also goes up. Effective waste management is another characteristic of smart city projects that is bound to benefit not only citizen but the environment as well. In some smart cities in India plans are afoot to convert certain waste materials to energy as well as fuel. It is necessary to dispose of waste properly in any city and this objective is at the heart of most smart cities.

Better utilization of water

Water is an essential commodity and urban areas face acute shortage during peak summers. So, preserving the water and its proper utilization are largely emphasized in the smart city planning. In this endeavor, the cities are stressing upon the installation of water meters to ensure residents are charged as per usage. These meters are also adept at detecting leakage as well. So, several smart city projects, especially in India, are being planned with sustainable water management systems, to avoid any kind of wastage. This is another hallmark of a smart city.

Smart transport management

Better traffic management is another characteristic of a smart city. An efficient public transport network that brings down energy consumption and traffic congestion are needed for the wellbeing of urban population. So, in smart cities the transport is managed in an efficient manner, so that the infamous traffic jams and the ensuing pollution become a thing of the past. In addition, a smart city can also reduce pollution levels and promote a healthier lifestyle.

Heightened safety and security

The protection and security of its citizens is vital for any city. In a smart city, the networks of CCTV cameras, adequate street lighting, intensive surveillance and patrolling, and a fast response system for emergency calls ensure utmost safety for its residents. Most of the smart city projects have advanced safety and security infrastructure as one of the top priorities while devising the urban initiative.

CHALLENGES:

1. Infrastructure Must Be a Foundational Element

The basic elements of a smart city today are stitched together from various stakeholders, vendors and technologies, which creates a fragmented ecosystem. As the initiative scales, this environment will not be able to meet its demands, support new technologies or effectively align with planned municipal services or construction efforts.

The reality of most cities capable of launching large-scale smart city initiatives is that their physical infrastructure is not suited to support them without significant changes to the existing components, which not only drives up costs but also disrupts the lives of residents.

In this sense, IT infrastructure is equally important — along with a common network delivered through adherence to industry-proven open standards — for a smart city initiative to support the demands of multiple solutions from technology and application providers, systems integrators and infrastructure service providers and operators.

1. A smart city's infrastructure platform should enable seamless integration of sensors, applications and services to not only improve returns on capital investments over time, but also provide key stakeholders with a strong foundation for their digital transformation journey.

2. Smart City IT Infrastructure Must Be Agile and Flexible to Scale

Infrastructure that is not scalable will be useless as smart city capabilities continue to evolve. While modular components are indeed necessary building blocks for smart cities, the amount of data used to power these modular components must be able to scale up as the amount of data produced increases.

For instance, as cities continue weaving together bus routes, ride-sharing apps and gridlock patterns with transportation infrastructure like traffic lights, data usage will soar. Without the ability to scale and connect the data pulled from each of these devices, the full benefits of a connected, smart city cannot be fully manifest themselves.

3. Cities Need Effective and Efficient Data Processing and Analytics

The ability to effectively and efficiently capture, store and analyze ever-growing amounts of IoT data closer to the edge is what really accelerates the benefits of smart cities. Smart cities are only as good as their ability to process data, which requires an intelligent and automated infrastructure that can handle exponential data creation and deliver the capabilities required to support long-term storage, processing and analysis.

4. Cities Must Protect Residents' Data to Assuage Privacy Concerns

While infrastructure provides the common foundation and offers advanced capabilities, open data and public trust weighs heavily on the success of a smart city project. In today's climate, government entities and private companies face rising scrutiny over data collection, with increasing public demand for transparency and oversight.

5. Political Differences Can Be a Roadblock to Smart City Deployments

The intricate dynamics and continuous cycle of politics is another ongoing challenge that could impede smart city initiatives. Large-scale smart city projects are often challenging to fund, as they require buy-in from multiple stakeholders involved in a public-private funding mechanism which blends interests from national, state and local levels with private enterprises.

6. Public and Private Sector Organizations Need to Coordinate

Collaboration and cooperation between key stakeholders in municipalities and the private sector can be another hurdle for smart cities. Government agencies and private sector organizations are often reluctant to share sensitive data or standardize on common networks, tools and infrastructure. This "need-to-know" data-sharing policy can prevent the kind of cross-collaboration that can help cities prevent terrorist attacks, improve drinking water and garbage collection and reduce noise and light pollution.

6.2 Industrial automation

Industrial Automation refers to the use of control systems, such as computers or robots and information technologies for handling different processes and machinery in an industry to replace human intervention. When integrated with Industrial IoT (IIoT), it becomes smarter, allowing machines to communicate, analyze, and act based on real-time data.

Key Components of Industrial Automation:

1. Sensors and Input Devices: Detect real-world parameters like temperature, pressure, speed, etc.
2. Controller (PLC, DCS, or Microcontroller): Processes data from sensors and makes decisions.
3. Actuators and Output Devices: Carry out actions like switching, moving, heating, etc., based on controller commands.
4. Human Machine Interface (HMI): Allows operators to monitor and interact with the system.
5. Communication Network: Facilitates data exchange between machines, controllers, and systems.

Industrial IoT (IIoT)

IIoT extends traditional automation by connecting machines to the internet, enabling real-time data analytics, remote monitoring, predictive maintenance, and smarter decision-making.

Key components

- IoT Sensors
- Cloud Computing
- Edge Devices
- Data Analytics Platforms

- Secure Communication Protocols

Benefits of Industrial Automation with IIoT:

- Improved productivity and efficiency
- Predictive maintenance and reduced downtime
- Real-time monitoring and control
- Enhanced safety and quality
- Energy efficiency

Industrial IoT

What is the industrial internet of things (IIoT)?

The industrial internet of things (IIoT) refers to the extension and use of the internet of things (IoT) in industrial sectors and applications with a strong focus on machine-to-machine (M2M) communication, big data, and machine learning, the IIoT enables industries and enterprises to have better efficiency and reliability in their operations.

□ The IIoT encompasses industrial applications, including robotics, medical devices, and software-defined production processes.

The IIoT goes beyond the normal consumer devices and internetworking of physical devices usually associated with the IoT.

What makes it distinct is the intersection of information technology (IT) and operational technology (OT).

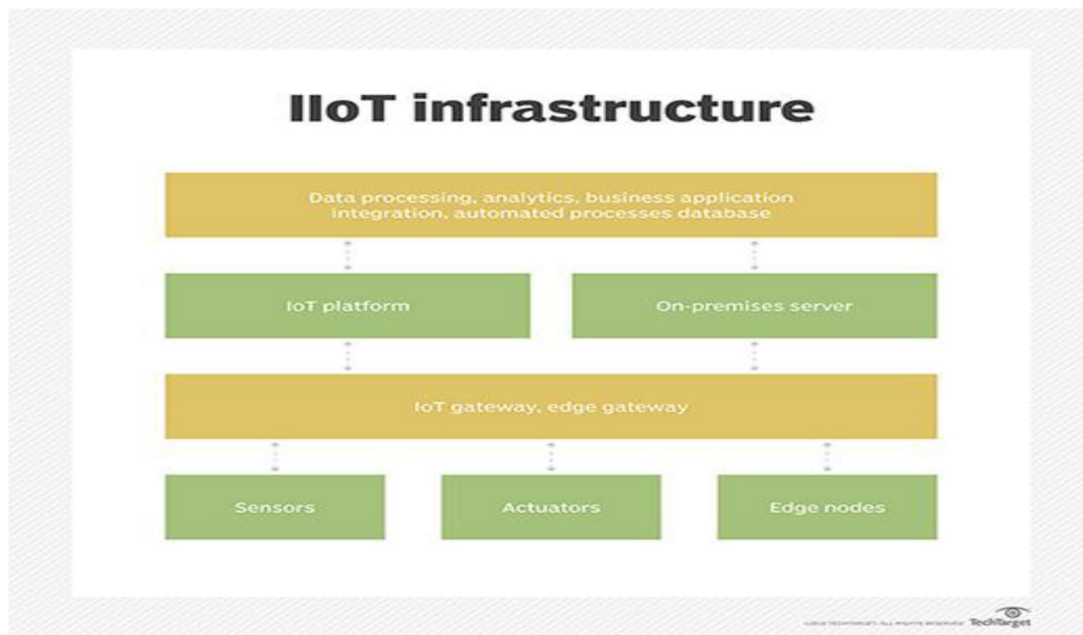
OT refers to the networking of operational processes and industrial control systems (ICSs), including human machine interfaces (HMIs), supervisory control and data acquisition (SCADA) systems, distributed control systems (DCSs), and programmable logic controllers (PLCs).

How does IIoT work?

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

connected devices 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;
storage for the data that is generated by the IIoT devices; and
people.

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



Which industries are using IIoT?

There are countless industries that make use of IIoT. One example is the automotive industry, which uses IIoT devices in the manufacturing process. The automotive industry extensively uses industrial robots, and IIoT can help proactively maintain these systems and spot potential problems before they can disrupt production.

The agriculture industry makes extensive use of IIoT devices, too. Industrial sensors collect data about soil nutrients, moisture and more, enabling farmers to produce an optimal crop.

The oil and gas industry also uses industrial IoT devices. Some oil companies maintain a fleet of autonomous aircraft that can use visual and thermal imaging to detect potential problems in pipelines. This information is combined with data from other types of sensors to ensure safe operations.

What are the benefits of IIoT?

Benefits of IIoT in Manufacturing and Beyond

According to Microsoft's IoT Signals report, 56% of companies adopt IIoT solutions for operations optimization, 47% cite workforce productivity as their primary use case, and 44% said that safety and security were the driving force behind adoption.

A report from Dell found that 49% of manufacturers have achieved improved process performance, while 33% say that they've made improvements to their asset utilization. Thirty-six percent of respondents report reductions in downtime.

While manufacturing is clearly leading the charge when it comes to adoption, other industries are embracing the benefits of IIoT, too. Here are a few examples:

- o **Pharmaceuticals.** The pharmaceutical automation company Parata Systems uses several IIoT technologies to identify the potential uses of its products and their impact on end-users and even makes predictions about how those products will perform.

- o **Agriculture.** IIoT is changing the game for agriculture as well. Interconnected sensors can be used to reduce water waste, monitor crops and livestock, track weather patterns to plan for the best possible yields, manage equipment, and more.

- o **Retail.** According to Microsoft's Spotlight on Retail report, retailers are embracing a wide range of IIoT applications. Among adopters, 57% use IoT for store analytics, 48% for supply chain optimization, 46% say security, and 45% say loss prevention.

- o **Mining.** IIoT is even shaking up mining. This case study looks at how Dundee Precious Metals used IIoT technologies to increase production by 400%, while creating a safer, knowledge-based workflow.

o **Oil & Gas.** Like the manufacturing industry, oil & gas companies can use IoT sensors to manage equipment and predict breakdowns. Additionally, sensors can detect hazards like gas leaks to prevent injury or death among workers and civilians.

One of the top touted benefits of IIoT devices used in the manufacturing industry is that they enable predictive maintenance. Organizations can use real-time data generated from IIoT systems to predict when a machine will need to be serviced. That way, the necessary maintenance can be performed before a failure occurs. This can be especially beneficial on a production line, where the failure of a machine might result in a work stoppage and huge costs. By proactively addressing maintenance issues, an organization can achieve better operational efficiency.

Another benefit is more efficient field service. IIoT technologies help field service technicians identify potential issues in customer equipment before they become major issues, enabling techs to fix the problems before they inconvenience customers. These technologies might also provide field service technicians with information about which parts they need to make a repair. That way, the technician has the necessary parts with them when making a service call.

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

IIoT also allows for enhanced customer satisfaction. When products are connected to the internet of things, the manufacturer can capture and analyze data about how customers use their products, enabling manufacturers and product designers to build more customer-centric product roadmaps.

IIoT also improves facility management. Manufacturing equipment is susceptible to wear and tear, which can be exacerbated by certain conditions in a factory. Sensors can monitor vibrations, temperature and other factors that might lead to suboptimal operating conditions.

Is IIoT secure?

Early on, manufacturers created IoT devices with little regard for security, resulting in a perception that IoT devices are inherently insecure. Given the similarities between IoT and IIoT devices, it's worth considering whether it's safe to use IIoT devices.

As with any other connected device, IIoT devices must be evaluated on a device-by-device basis. It's entirely possible that one manufacturer's device is secure while another isn't. Even so, security is a bigger priority among device manufacturers than ever before.

In 2014, several technology companies including AT&T, Cisco, General Electric, IBM and Intel came together to form the Industrial Internet Consortium (IIC). Although this group's primary objective is to accelerate the adoption of IIoT and related technologies, it's making security a priority, even going so far as to form a security working group. The IIC's other working groups include Technology, Liaison, Marketing, Industry and Digital Transformation.

What is the difference between IoT and IIoT?

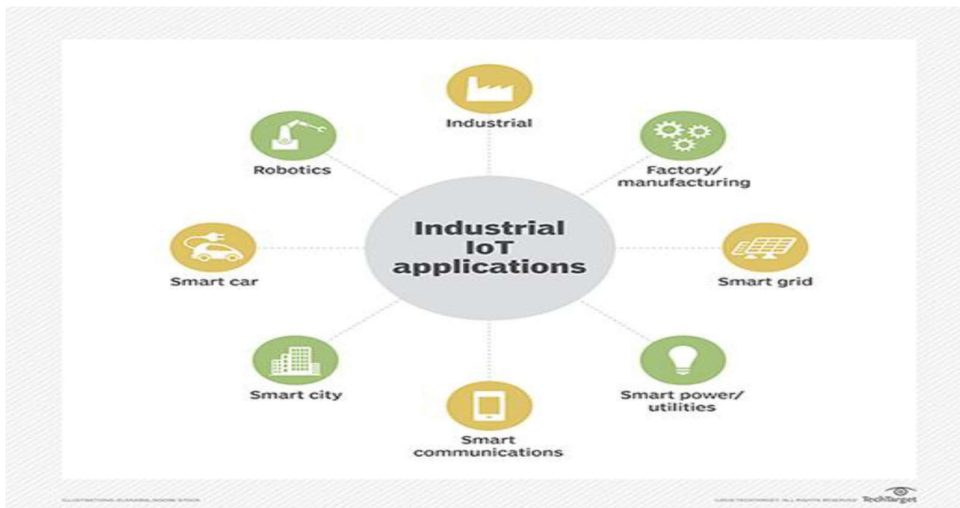
Although IoT and IIoT have many technologies in common, including cloud platforms, sensors, connectivity, machine-to-machine communications and data analytics, they are used for different purposes.

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.

What are IIoT applications and examples?

In a real-world IIoT deployment of smart robotics, ABB, a power and robotics firm, uses connected sensors to monitor the maintenance needs of its robots to prompt repairs before parts break.

Likewise, commercial jetliner maker Airbus has launched what it calls the factory of the future, a digital manufacturing initiative to streamline operations and boost production. Airbus has integrated sensors into machines and tools on the shop floor and outfitted employees with wearable tech -- e.g., industrial smart glasses -- aimed at cutting down on errors and enhancing workplace Safety.



IIoT is used in many industries and sectors, including robotics, manufacturing and smart cities. Another robotics manufacturer, Fanuc, is using sensors in its robotics, along with cloud-based data analytics, to predict the imminent failure of components in its robots. Doing so enables the plant manager to schedule maintenance at convenient times, reducing costs and averting potential downtime.

The key benefits of IIoT in an industry context

- Improved and intelligent connectivity between devices or machines
- Increased efficiency
- Cost savings and
- Time savings
- Enhanced industrial safety

What are the risks to IIoT systems?

Many security problems associated with the IIoT stem from a lack of basic security measures in place. Security gaps like exposed ports, inadequate authentication practices, and obsolete applications contribute to the emergence of risks. Combine these with having the network directly connected to the internet and more potential risks are invited.

Unsecure IIoT systems can lead to operational disruption and monetary loss, among other considerable consequences. More connected environments mean more security risks, such as:

- Software vulnerabilities that can be exploited to attack systems.

- Publicly searchable internet-connected devices and systems.

- Malicious activities like hacking, targeted attacks, and data breaches.

- System manipulation that can cause operational disruption (e.g., product recalls) or sabotage processes (e.g., production line stoppage).

- System malfunction that can result in damage of devices and physical facilities or injury to operators or people nearby.

- OT systems held for extortion, as compromised through the IT environment.

Who are IIoT vendors?

There are several vendors with IIoT platforms, including:

ABB Ability. An IIoT company specializing in connectivity, software and machine intelligence.

Aveva Wonderware. A company that develops human-machine interface (HMI) and IoT edge platforms for OEMs (original equipment manufacturers) and end users.

Axzon. An IIoT company focusing on smart automotive manufacturing, predictive maintenance and cold chain.

Cisco IoT. A networking company offering platforms for network connectivity, connectivity management, data control and exchange, and edge computing.

Fanuc Field System. A company that has developed a platform for connecting various generations, makes and models of industrial IoT equipment.

Linx Global Manufacturing. A product development and manufacturing company offering custom IIoT, application and data management platforms.

MindSphere by Siemens. An industrial IoT solution based around artificial intelligence (AI) and advanced analytics.

Plataine. An IIoT company specializing in using AI to generate actionable insights in manufacturing.

Predix by GE. A platform for connecting, optimizing and scaling digital industrial applications.

IIoT and 5G

5G is the emerging standard for mobile networks. It has been specifically designed to deliver fast data throughput speeds with low latency. 5G will support download speeds of up to 20 Gbps (gigabits per second) with sub-millisecond latency.

The emergence of 5G will likely affect the use of IIoT devices in two main ways. First, 5G's high throughput and low latency will make it possible for devices to share data in real time. Previously, this was only possible when the devices were located on private networks with high-speed connectivity. This real-time connectivity will support use cases such as driverless cars and smart cities. The other way 5G will affect IIoT adoption is that it will likely result in device proliferation. Industrial operations might use thousands of 5G connected devices. 5G's high speed and low latency also means we'll likely see IIoT devices used in remote sites whose lack of high-speed connectivity previously made IIoT use impractical.