

Lecture Notes

Internet Of Things



Prepared By:
MANAS RANJAN MISHRA
ASST.PROF., (CSE)CVRP

UNIT-1

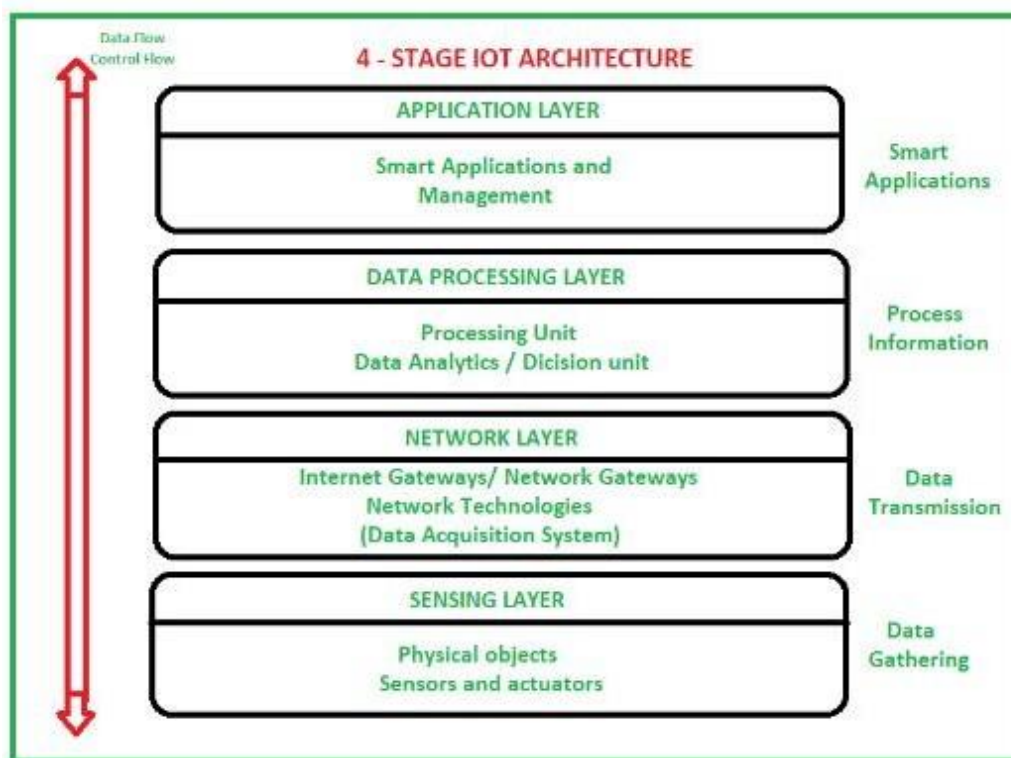
Introduction to IoT

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

Architectural Overview

IOT technology has a wide variety of applications and use of Internet of Things is growing so faster. Depending upon different application areas of Internet of Things, it works accordingly as per it has been designed/developed. But it has not a standard defined architecture of working which is strictly followed universally. The architecture of IoT depends upon its functionality and implementation in different sectors. Still, there is a basic process flow based on which IoT is built.

So, here in this article we will discuss basic fundamental architecture of IoT i.e., 4 Stage IoT architecture.



So, from the above image it is clear that there are 4 layers present that can be divided as follows: Sensing Layer, Network Layer, Data processing Layer, and Application Layer.

These are explained as follows below.

1. **Sensing Layer –**

Sensors, actuators, devices are present in this Sensing layer. These

Sensors or Actuators accepts data(physical/environmental parameters), processes data and emits data over network.

2. **Network Layer –**

Internet/Network gateways, Data Acquisition System (DAS) are present in this layer. DAS performs data aggregation and conversion function (Collecting data and aggregating data then converting analog data of sensors to digital data etc). Advanced gateways which mainly opens up connection between Sensor networks and Internet also performs many basic gateway functionalities like malware protection, and filtering also some times decision making based on inputted data and data management services, etc.

3. **Data processing Layer –**

This is processing unit of IoT ecosystem. Here data is analyzed and pre-processed before sending it to data center from where data is accessed by software applications often termed as business applications where data is monitored and managed and further actions are also prepared. So here Edge IT or edge analytics comes into picture.

4. **Application Layer –**

This is last layer of 4 stages of IoT architecture. Data centers or cloud is management stage of data where data is managed and is used by end-user applications like agriculture, health care, aerospace, farming, defense, etc.

Design principles and needed capabilities

6 principles of IoT design

1. Do your research

When designing IoT-enabled products, designers might make the mistake of forgetting why customers value these products in the first place. That's why it's a good idea to think about the value an IoT offering should deliver at the initial phase of your design.

When getting into IoT design, you're not building products anymore. You're building services and experiences that improve people's lives. That's why in-depth qualitative research is the key to figuring out how you can do that.

Assume the perspective of your customers to understand what they need and how your IoT implementation can solve their pain points. Research your target audience deeply to see what their existing experiences are and what they wish was different about them.

2. Concentrate on value

Early adopters are eager to try out new technologies. But the rest of your customer base might be reluctant to put a new solution to use. They may not feel confident with it and are likely to be cautious about using it.

If you want your IoT solution to become widely adopted, you need to focus on the actual tangible value it's going to deliver to your target audience.

What is the real end-user value of your solution? What might be the barriers to adopting new technology? How can your solution address them specifically?

Note that the features the early tech adopters might find valuable might turn out to be completely uninteresting for the majority of users. That's why you need to carefully plan which features to include and in what order, always concentrating on the actual value they provide.

3. Don't forget about the bigger picture

One characteristic trait of IoT solutions is that they typically include multiple devices that come with different capabilities and consist of both digital and physical touchpoints. Your solution might also be delivered to users in cooperation with service providers.

That's why it's not enough to design a single touchpoint well. Instead, you need to take the bigger picture into account and treat your IoT system holistically.

Delineate the role of every device and service. Develop a conceptual model of how users will perceive and understand the system. All the parts of your system need to work seamlessly together. Only then you'll be able to create a meaningful experience for your end-users.

4. Remember about the security

Don't forget that IoT solutions aren't purely digital. They're located in the real-world context, and the consequences of their actions might be serious if something goes wrong. At the same time, building trust in IoT solutions should be one of your main design drivers.

Make sure that every interaction with your product builds consumer trust rather than breaking it. In practice, it means that you should understand all the possible error situations that may be related to the context of its use. Then try to design your product in a way to prevent them. If error situations occur, make sure that the user is informed appropriately and provided with help.

Also, consider data security and privacy as a key aspect of your implementation. Users need to feel that their data is safe, and objects located in their workspaces or home can't be hacked. That's why quality assurance and testing the system in the real-world context are so important.

5. Build with the context in mind

And speaking of context, it pays to remember that IoT solutions are located at the intersection of the physical and digital world. The commands you give through digital interfaces produce real-world effects. Unlike digital commands, these actions may not be easily undone.

In a real-world context, many unexpected things may happen. That's why you need to make sure that the design of your solution enables users to feel safe and in control at all times.

The context itself is a crucial consideration during IoT design. Depending on the physical context of your solution, you might have different goals in mind.

For example, you might want to minimize user distraction or design devices that will be resistant to the changing weather conditions.

The social context is an important factor, as well. Don't forget that the devices you design for workspaces or homes will be used by multiple users.

6. Make good use of prototypes

IoT solutions are often difficult to upgrade. Once the user places the connected object somewhere, it might be hard to replace it with a new version – especially if the user would have to pay for the upgrade.

Even the software within the object might be hard to update because of security and privacy reasons. Make sure that your design practices help to avoid costly hardware iterations. Get your solution right from the start. From the design perspective, it means that prototyping and rapid iteration will become critical in the early stages of the project.

Characteristics of IoT

There are 7 crucial IoT characteristics:

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

DEFINING IOT: 7 CHARACTERISTICS



IoT Applications

Smart Homes

One of the best and the most practical applications of IoT, smart homes really take both, convenience and home security, to the next level. Though there are different levels at which IoT is applied for smart homes, the best is the one that blends intelligent utility systems and entertainment together. For instance, your electricity meter with an IoT device giving you insights into your everyday water usage, your set-top box that allows you to record shows from remote, Automatic Illumination Systems, Advanced Locking Systems, Connected Surveillance Systems all fit into this concept of smart homes. As IoT evolves, we can be sure that most of the devices will become smarter, enabling enhanced home security.

2. Smart City

Not just internet access to people in a city but to the devices in it as well – that's what smart cities are supposed to be made of. And we can proudly say that we're going towards realizing this dream. Efforts are being made to incorporate connected technology into infrastructural requirements and some vital concerns like Traffic Management, Waste Management, Water Distribution, Electricity Management, and more. All these work towards eliminating some day-to-day challenges faced by people and bring in added convenience.

3. Self-driven Cars

We've seen a lot about self-driven cars. Google tried it out, Tesla tested it, and even Uber came up with a version of self-driven cars that it later shelved. Since it's human lives on the roads that we're dealing with, we need to ensure the technology has all that it takes to ensure better safety for the passenger and those on the roads.

The cars use several sensors and embedded systems connected to the Cloud and the internet to keep generating data and sending them to the Cloud for informed decision-making through Machine Learning. Though it will take a few more years for the technology to evolve completely and for countries to amend laws and policies, what we're witnessing right now is one of the best applications of IoT.

4. IoT Retail Shops

If you haven't already seen the video of Amazon Go – the concept store from the eCommerce giant, you should check it out right away. Perhaps this is the best use of the technology in bridging the gap between an online store and a retail store. The retail store allows you to go cashless by deducting money from your Amazon wallet. It also adds items to your cart in real-time when you pick products from the shelves.

If you change your mind and pick up another article, the previous one gets deleted and replaces your cart with the new item. The best part of the concept store is that there is no cashier to bill your products. You don't have to stand in line but just step out after you pick up your products from shelves. If this technology is effective enough to fetch more patronage, this is sure to become a norm in the coming years.

5. Farming

Farming is one sector that will benefit the most from the Internet of Things. With so many developments happening on tools farmers can use for agriculture, the future is sure promising. Tools are being developed for Drip Irrigation, understanding crop patterns, Water Distribution, drones for Farm Surveillance, and more. These will allow farmers to come up with a more productive yield and take care of the concerns better.

6. Wearables

Wearables remain a hot topic in the market, even today. These devices serve a wide range of purposes ranging from medical, wellness to fitness. Of all the IoT startups, Jawbone, a wearables maker, is second to none in terms of funding.

7. Smart Grids

One of the many useful IoT examples, a smart grid, is a holistic solution that applies an extensive range of Information Technology resources that enable existing and new gridlines to reduce electricity waste and cost. A future smart grid improves the efficiency, reliability, and economics of electricity.

8. Industrial Internet

The Industrial Internet of Things consists of interconnected sensors, instruments, and other devices connected with computers' industrial applications like manufacturing, energy management, etc. While still being unpopular in comparison to IoT wearables and other uses, market researches like Gartner, Cisco, etc., believe the industrial internet to have the highest overall potential.

9. Telehealth

Telehealth, or Telemedicine, hasn't completely flourished yet. Nonetheless, it has great future potential. IoT Examples of Telemedicine include the digital communication of Medical Imaging, Remote Medical Diagnosis & Evaluations, Video Consultations with Specialists, etc.

10. Smart Supply-chain Management

Supply-chains have stuck around in the market for a while now. A common example can be Solutions for tracking goods while they are on the road. Backed with IoT technology, they are sure to stay in the market for the long run.

Basic IoT Sensors

Sensors are everywhere. They're in our homes and workplaces, our shopping centers and hospitals. They're embedded in smart phones and an integral part of the Internet of Things (IoT). Sensors have been around for a long time. The first thermostat was introduced in the late 1880s and infrared sensors have been around since the late

1940s. The IoT and its counterpart, the Industrial Internet of Things (IIoT), are bringing sensor usage to a new level.

Broadly speaking, sensors are devices that detect and respond to changes in an environment. Inputs can come from a variety of sources such as light, temperature, motion and pressure. Sensors output valuable information and if they are connected to a network, they can share data with other connected devices and management systems.



Temperature Sensors

Temperature sensors measure the amount of heat energy in a source, allowing them to detect temperature changes and convert these changes to data. Machinery used in manufacturing often requires environmental and device temperatures to be at specific levels. Similarly, within agriculture, soil temperature is a key factor for crop growth.



2. Humidity Sensors

These types of sensors measure the amount of water vapor in the atmosphere of air or other gases. Humidity sensors are commonly found in heating, vents and air conditioning (HVAC) systems in both industrial and residential domains. They can be found in many other areas including hospitals, and meteorology stations to report and predict weather.



3. Pressure Sensors

A pressure sensor senses changes in gases and liquids. When the pressure changes, the sensor detects these changes, and communicates them to connected systems. Common use cases include leak testing which can be a result of decay. Pressure sensors are also useful in the manufacturing of water systems as it is easy to detect fluctuations or drops in pressure.



4. Proximity Sensors

Proximity sensors are used for non-contact detection of objects near the sensor. These types of sensors often emit electromagnetic fields or beams of radiation such as infrared. Proximity sensors have some interesting use cases. In retail, a proximity sensor can detect the motion between a customer and a product in which he or she is interested. The user can be notified of any discounts or special offers of products located near the sensor. Proximity sensors are also used in the parking lots of malls, stadiums and airports to indicate parking availability. They can also be used on the assembly lines of chemical, food and many other types of industries.



5. Level Sensors

Level sensors are used to detect the level of substances including liquids, powders and granular materials. Many industries including oil manufacturing, water treatment and beverage and food manufacturing factories use level sensors. Waste

management systems provide a common use case as level sensors can detect the level of waste in a garbage can or dumpster.



6. Accelerometers

Accelerometers detect an object's acceleration i.e. the rate of change of the object's velocity with respect to time. Accelerometers can also detect changes to gravity. Use cases for accelerometers include smart pedometers and monitoring driving fleets. They can also be used as anti-theft protection alerting the system if an object that should be stationary is moved.



7. Gyroscope

Gyroscope sensors measure the angular rate or velocity, often defined as a measurement of speed and rotation around an axis. Use cases include automotive, such as car navigation and electronic stability control (anti-skid) systems. Additional use cases include motion sensing for video games, and camera-shake detection systems.



8. Gas Sensors

These types of sensors monitor and detect changes in air quality, including the presence of toxic, combustible or hazardous gasses. Industries using gas sensors include mining, oil and gas, chemical research and manufacturing. A common consumer use case is the familiar carbon dioxide detectors used in many homes.



9. Infrared Sensors

These types of sensors sense characteristics in their surroundings by either emitting or detecting infrared radiation. They can also measure the heat emitted by objects. Infrared sensors are used in a variety of different IoT projects including healthcare as they simplify the monitoring of blood flow and blood pressure. Televisions use infrared sensors to interpret the signals sent from a remote control. Another interesting application is that of art historians using infrared sensors to see hidden layers in paintings to help determine whether a work of art is original or fake or has been altered by a restoration process.



10. Optical Sensors

Optical sensors convert rays of light into electrical signals. There are many applications and use cases for optical sensors. In the auto industry, vehicles use optical sensors to recognize signs, obstacles, and other things that a driver would notice when driving or parking. Optical sensors play a big role in the development of driverless cars. Optical sensors are very common in smart phones. For example, ambient light sensors can extend battery life. Optical sensors are also used in the biomedical field including breath analysis and heart-rate monitors.

Basic IoT Actuators

Actuators convert an electrical signal into a corresponding physical quantity such as movement, force, sound etc. An actuator is also classed as a transducer because it changes one type of physical quantity into another and is usually activated or operated by a low voltage command signal. Actuators can be classed as either binary or continuous devices based upon the number of stable states their output has.

For example, a relay is a binary actuator as it has two stable states, either energized and latched or de-energised and unlatched, while a motor is a continuous actuator because it can rotate through a full 360o motion.

Let's explore some of the basic actuators you may use in your IoT projects –

1. Servo Motors:



A Servo is a small device that incorporates a two wire DC motor, a gear train, a potentiometer, an integrated circuit, and a shaft (output spine). The shaft can be positioned to specific angular positions by sending the servo a coded signal. Of the three wires that stick out from the servo casing, one is for power, one is for ground, and one is a control input line.

When a control signal is applied to a Servo that represents a desired output position of the servo shaft, it (servo) applies power to its DC motor until its shaft turns to that position. It uses the position-sensing device to determine the rotational position of the shaft, so it knows which way the motor must turn to move the shaft to the commanded position.

2. Stepper Motors:



Stepper motors are DC motors that move in discrete steps. They have multiple coils that are organized in groups called “phases”. By energizing each phase in sequence, the motor will rotate, one step at a time. With a computer controlled stepping, you can achieve very precise positioning and/or speed control.

A servomotor consumes power as it rotates to the commanded position but then the servomotor rests. Stepper motors continue to consume power to lock in and hold the commanded position.

3. DC Motors (Continuous Rotation Motors):



Direct Current (DC) motor is the most common actuator used in electronics projects. They are simple, cheap, and easy to use. Also, they come in a great variety of sizes, to accommodate different tasks. DC motors convert electrical into mechanical energy. They consist of permanent magnets and loops of wire inside. When current is applied, the wire loops generate a magnetic field, which reacts against the outside field of the static magnets.

4. Linear actuator:



A linear actuator is an actuator that creates motion in a straight line, in contrast to the circular motion of a conventional electric motor. Linear actuators are used in machine tools and industrial machinery, in computer peripherals such as disk drives and printers, in valves and dampers, and in many other places where linear motion is required.

5. Relay:



A relay is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. The advantage of relays is that it takes a relatively small amount of power to operate the relay coil, but the relay itself can be used to control motors, heaters, lamps or AC circuits which themselves can draw a lot more electrical power.

6. Solenoid:



A solenoid is simply a specially designed electromagnet. Solenoids are inexpensive, and their use is primarily limited to on-off applications such as latching, locking, and triggering. They are frequently used in home appliances (e.g. washing machine valves), office equipment (e.g. copy machines), automobiles (e.g. door latches and the starter solenoid), pinball machines (e.g., plungers and bumpers), and factory automation.

Key Components of IoT

We can list down the below components as key parts of an IoT ecosystem.

1. Sensors or End Devices

For any IoT use case, the components of the endpoint are sensors. Sensors capture electric pulse or analog signals which are passed through the IoT ecosystems. Based on the use case and domains RFID, temperature sensors, light sensors, electromagnetic sensors, etc. are used. For example, smartphones and smart wearables are equipped with sensors like accelerometer, Gyroscope sensors, etc. Data obtained from these IoT endpoints can be used in various domains like Human activity recognition, medical stability, etc. Based on the use case and precision requirements sensors can be chosen keeping the following parameters in mind

- Accuracy of the input readings
- Reliability percentage of the inputs
- The purpose of the use case, for example, sensors required for a temperature-dependent use case, will differ from use cases based on motions.
- Industry grade IoT systems can be integrated with multi-technology, cross-functional and cross-vendor products. Based on the complexity and compatibility sensors are chosen for a particular use case.

2. Network or Connectivity Layer

In a typical IoT ecosystem, sensors are connected with computation layers and intelligent layers via network or connectivity layers. IoT endpoints need to be always connected with various other components seamlessly over the connectivity layer. Based on the scale of the implementations IoT components can be connected over LANs, MANs or WANs. It can also be connected through telephony networks like LTE (Long Term Evolution or popularly known as 4G Network) or light-based technologies like Li-Fi (where light is used as a mode of communication to maintain interconnections). For local use cases, Bluetooth and Wi-Fi can also be used.

An IoT network consists of various network components like routers, gateways, switches, various network protocols, etc. Based on the use case and domain proper network infrastructure is needed to be chosen.

3. Security Layers

The heart of any industry-grade IoT user story is 'data'. In a standard use case, analog or digital signal is acquired by sensors and the signal is then converted to a format on top of which AI/ML components can work. In the total flow of data, proper security systems and methodologies need to be enforced. The data can be compromised in any layers starting from the data acquisition to business insights derivations. We can enforce proper security by using strong encryption in various layers of communication, using proper firmware and anti-malware systems, etc.

4. Compute Engines

Industry grade IoT systems typically use multiple technology stacks inside an umbrella. For example in insurance premiums can be calculated as a variable component as per the driving pattern of the insurer. The data collected from smart devices are converted and preprocessed to a format on which machine learning models are developed. Customers can use any cloud partners of their choice or develop their own infrastructure to execute a use case.

For example, the compute engines from PaaS (Product as a service) or IaaS (IoT as service) will differ from on-premise systems.

5. Technology and Governance Standards

Sensitive information flow over the various components of the IoT ecosystem. To cope up with this the systems need to adhere to proper technique and governance standard and KPIs

- Typical Technical standards: Wi-Fi, WAN, etc.
- Network Protocols: HTTP, TCP/IP, UDP, etc.
- Data management standards: ETL, CAP (for distributed systems), etc.
- IoT systems need to follow the regulations and quality standards of respective regulatory authorities and business standards.

6. *Intelligent Insights and Actions*

- Most of the practical and industry-grade IoT use cases are intended to derive business insights or actionable recommendations. The preprocessed data need to be integrated with ML components and the trained models are deployed to the production environment. The choice of the technology stack to develop the intelligent business component is dependent on the compatibility with the in house existing systems, the scale of the business, the complexity of the use case, and precision and latency requirements of the domain, company partnerships, etc.

Challenges for IoT

The **Internet Of Things** has been facing many areas like Information Technology, Healthcare, Data Analytics and Agriculture. The main focus is on protecting privacy as it is the primary reason for other challenges including government participation. Integrated effort from the government, civil society and private sectors would play a vital role in protecting the following values given below in to prevent IoT from getting hampered:

Scalability:

Billions of internet-enabled devices get connected in a huge network, large volumes of data are needed to be processed. The system that stores, analyses the data from these IoT devices needs to be scalable. In present, the era of IoT evolution everyday objects are connected with each other via

Internet. The raw data obtained from these devices need big data analytics and cloud storage for interpretation of useful data.

Interoperability:

Technological standards in most areas are still fragmented. These technologies need to be converged. Which would help us in establishing a common framework and the standard for the IoT devices. As the standardization process is still lacking, interoperability of IoT with legacy devices should be considered critical. This lack of interoperability is preventing us to move towards the vision of truly connected everyday interoperable smart objects.

Lack of government support:

Government and Regulatory bodies like FDA should come up and bring up regulations by setting up a standard committee for safety and security of devices and people.

Safety Of Patients:

Most Of IoT devices are left unattended, as they are connected with real-world objects. If used on patients as wearable devices, any technical error in security can be life-threatening for patient.

Security And Personal Privacy:

There has been no research in security vulnerabilities and its improvements. It should ensure Confidentiality, Integrity and Availability of personal data of patient.

Design Based Challenge:

With the development in technology design challenges are increasing at a faster rate. There have been issues regarding design like limited computation power, limited energy and limited memory which need to be sorted out.

IOT Networking

An IoT network refers to a collection of interconnected devices that communicate with other devices without the need for human involvement, such as autonomous cars, smart appliances, and wearable tech.

Terminologies

The major terminologies used in computer networking –

WAN

It stands for Wide Area Network and covers a wide area such as a city.

LAN

It stands for Local Area Network and covers a small area such as a small office or home. It physically connects all the computers located in the premises.

Internet

It is a computer network system that connects the computers of the world. It is normally connecting through WAN and LAN.

Intranet

It is a close room computer network system, as it covers a small area and only authorized people can access it.

Extranet

It is also a sort of Internet the access to which is granted only to a few.

World Wide Web (WWW)

It is the service that is used on Internet to view and search contents (in the form of web-pages).

Instant messaging (IM)

It is an online facility that facilitates us to chat or talk. Such service is provided by Skype, Google Talk, Windows Live Messenger, Yahoo Messenger, etc.

Voice over Internet Protocol (VoIP)

It is a Protocol, which is used especially for voice transfer over IP network. Likewise, it facilitates users to make phone-calls by using internet.

Really Simple Syndication (RSS)

It is a technique, which is used for the dissemination of information, articles, etc. Users normally subscribe to the RSS channel in order to receive news. After

Asymmetric Digital Subscriber Line (ADSL)

It is a sort of digital subscriber line (DSL) technology that facilitates faster data transmission.

Download

It is a process that saves data from Internet onto a personal computer.

Upload

It is a process that transfers the saved data from a personal computer to Internet server.

Dial-up

It is a technique in which a phone line is used in order to connect to the Internet.

Broadband

- ❖ It is a wide bandwidth data transmission that transports multiple signals and traffic types swiftly.

Gateway Prefix allotment

- which can forward packets from the nodes, to the Internet, only via these routers.
- These routers assign prefixes to gateways under them, so that the gateways can be identified with them.

Impact of mobility on Addressing

- ❖ Mobility facilitates employee productivity and increased profit as a result, fitting right in with the increased connectivity of IoT.
- ❖ IoT is enabled by enterprise mobility because companies rely on the connected devices and big data analytics that mobility provides.

Multihoming

- ❖ Multihoming is a mechanism used to configure one computer with more than one network interface and multiple IP addresses.
- ❖ It provides enhanced and reliable Internet connectivity without

compromising efficient performance.

- ❖ The multihoming computer is known as the host and is directly or indirectly connected to more than one network.

IoT identification and Data protocols

The protocols into the following layers to provide some level of organization:

1. **Infrastructure** (ex: 6LowPAN, IPv4/IPv6, RPL)
2. **Identification** (ex: EPC, uCode, IPv6, URIs)
3. **Comms / Transport** (ex: Wifi, Bluetooth, LPWAN)
4. **Discovery** (ex: Physical Web, mDNS, DNS-SD)
5. **Data Protocols** (ex: MQTT, CoAP, AMQP, Websocket, Node)
6. **Device Management** (ex: TR-069, OMA-DM)
7. **Semantic** (ex: JSON-LD, Web Thing Model)
8. **Multi-layer Frameworks** (ex: Alljoyn, IoTivity, Weave, Homekit)

IOT Communication Protocol

Bluetooth



An important short-range communications technology is of course Bluetooth, which has become very important in computing and many consumer product markets. It is expected to be key for wearable products in particular, again connecting to the IoT albeit probably via a smartphone in many cases. The new Bluetooth Low-Energy (BLE) – or Bluetooth Smart, as it is now branded – is a significant protocol for IoT applications. Importantly, while it offers similar range to Bluetooth it has been designed to offer significantly reduced power

consumption.

However, Smart/BLE is not really designed for file transfer and is more suitable for small chunks of data.

- Standard: Bluetooth 4.2 core specification
- Frequency: 2.4GHz (ISM)
- Range: 50-150m (Smart/BLE)
- Data Rates: 1Mbps (Smart/BLE)

Zigbee



ZigBee, like Bluetooth, has a large installed base of operation, although perhaps traditionally more in industrial settings. ZigBee PRO and ZigBee Remote Control (RF4CE), among other available ZigBee profiles, are based on the IEEE802.15.4 protocol, which is an industry-standard wireless networking technology operating at 2.4GHz targeting applications that require relatively infrequent data exchanges at low data-rates over a restricted area and within a 100m range such as in a home or building.

ZigBee/RF4CE has some significant advantages in complex systems offering low-power operation, high security, robustness and high scalability with high node counts and is well positioned to take advantage of wireless control and sensor networks in M2M and IoT applications. The [latest version of ZigBee](#) is the recently launched 3.0, which is essentially the unification of the various ZigBee wireless standards into a single standard. An example product and kit for ZigBee development are TI's CC2538SF53RTQT ZigBee System-On-Chip IC and CC2538 ZigBee Development Kit.

- Standard: ZigBee 3.0 based on IEEE802.15.4
- Frequency: 2.4GHz
- Range: 10-100m
- Data Rates: 250kbps

Z-Wave

IOT Communication Protocol



Z-Wave is a low-power RF communications technology that is primarily designed for home automation for products such as lamp controllers and sensors among many others. Optimized for reliable and low-latency communication of small data packets with data rates up to 100kbit/s, it operates in the sub-1GHz band and is impervious to interference from WiFi and other wireless technologies in the 2.4-GHz range such as Bluetooth or ZigBee. It supports full mesh networks without the need for a coordinator node and is very scalable, enabling control of up to 232 devices. Z-Wave uses a simpler protocol than some others, which can enable faster and simpler development, but the only maker of chips is Sigma Designs compared to multiple sources for other wireless technologies such as ZigBee and others.

- Standard: Z-Wave Alliance ZAD12837 / ITU-T G.9959
- Frequency: 900MHz (ISM)
- Range: 30m
- Data Rates: 9.6/40/100kbit/s

6LowPAN

6LoWPAN

A key IP (Internet Protocol)-based technology is 6LowPAN (IPv6 Low-power wireless Personal Area Network). Rather than being an IoT application protocols technology like Bluetooth or ZigBee, 6LowPAN is a network protocol that defines encapsulation and header compression mechanisms. The standard has the freedom of frequency band and physical layer and can also be used across multiple communications platforms, including Ethernet, Wi-Fi, 802.15.4 and sub-1GHz ISM. A key attribute is the IPv6 (Internet Protocol version 6) stack, which has been a very important introduction in recent years to enable the IoT. IPv6 is the successor to IPv4 and offers approximately 5×10^{28} addresses for every person in the world, enabling any embedded object or device in the world to have its own unique IP address and connect to the Internet. Especially designed for home or building automation, for example, IPv6 provides a basic transport mechanism to produce complex control systems and to communicate with devices in a cost-effective manner via a low-power wireless network.

Designed to send IPv6 packets over IEEE802.15.4-based networks and implementing open IP standards including TCP, UDP, HTTP, COAP, MQTT, and websockets, the standard offers end-to-end addressable nodes, allowing a router to connect the network to IP. 6LowPAN is a mesh network that is robust, scalable and self-healing. Mesh router devices can route data destined for other devices, while hosts are able to sleep for long periods of time. An explanation of 6LowPAN is available [here](#), courtesy of TI.

- Standard: RFC6282

IOT Communication Protocol

- Frequency: (adapted and used over a variety of other networking media including Bluetooth Smart (2.4GHz) or ZigBee or low-power RF (sub-1GHz))
- Range: N/A
- Data Rates: N/A

Thread



A very new IP-based IPv6 networking protocol aimed at the home automation environment is Thread. Based on 6LoWPAN, and also like it, it is not an IoT applications protocol like Bluetooth or ZigBee. However, from an application point of view, it is primarily designed as a complement to WiFi as it recognises that while WiFi is good for many consumer devices that it has limitations for use in a home automation setup.

Launched in mid-2014 by the [Thread Group](#), the royalty-free protocol is based on various standards including IEEE802.15.4 (as the wireless air-interface protocol), IPv6 and 6LoWPAN, and offers a resilient IP-based solution for the IoT. Designed to work on existing IEEE802.15.4 wireless silicon from chip vendors such as Freescale and Silicon Labs, Thread supports a mesh network using IEEE802.15.4 radio transceivers and is capable of handling up to 250 nodes with high levels of authentication and encryption. A relatively simple software upgrade should allow users to run thread on existing IEEE802.15.4-enabled devices.

- Standard: Thread, based on IEEE802.15.4 and 6LoWPAN
- Frequency: 2.4GHz (ISM)
- Range: N/A
- Data Rates: N/A

WiFi



WiFi connectivity is often an obvious choice for many developers, especially given the pervasiveness of WiFi within the home environment within LANs. It requires little further explanation except to state the obvious that clearly there is a wide existing infrastructure as well as offering fast data transfer and the ability to handle high quantities of data.

IOT Communication Protocol

Currently, the most common WiFi standard used in homes and many businesses is 802.11n, which offers serious throughput in the range of hundreds of megabit per second, which is fine for file transfers, but may be too power-consuming for many IoT applications. A series of [RF development kits](#) designed for building WiFi-based applications are available from RS.

- Standard: Based on 802.11n (most common usage in homes today)
- Frequencies: 2.4GHz and 5GHz bands
- Range: Approximately 50m
- Data Rates: 600 Mbps maximum, but 150-200Mbps is more typical, depending on channel frequency used and number of antennas (latest 802.11-ac standard should offer 500Mbps to 1Gbps)

Cellular



< Any IoT application that requires operation over longer distances can take advantage of GSM/3G/4G cellular communication capabilities. While cellular is clearly capable of sending high quantities of data, especially for 4G, the expense and also power consumption will be too high for many applications, but it can be ideal for sensor-based low-bandwidth-data projects that will send very low amounts of data over the Internet. A key product in this area is the [SparqEE](#) range of products, including the original tiny CELLv1.0 low-cost development board and a series of shield connecting boards for use with the Raspberry Pi and Arduino platforms.

- Standard: GSM/GPRS/EDGE (2G), UMTS/HSPA (3G), LTE (4G)
- Frequencies: 900/1800/1900/2100MHz
- Range: 35km max for GSM; 200km max for HSPA
- Data Rates (typical download): 35-170kps (GPRS), 120-384kbps (EDGE), 384Kbps-2Mbps (UMTS), 600kbps-10Mbps (HSPA), 3-10Mbps (LTE)

NFC



NFC (Near Field Communication) is a technology that enables simple and safe two-way interactions between electronic devices, and especially applicable for smartphones, allowing consumers to perform contactless payment transactions, access digital content and connect electronic devices. Essentially it extends the capability of contactless card technology and enables

IOT Communication Protocol

devices to share information at a distance that is less than 4cm. Further information is available [here](#).

- Standard: ISO/IEC 18000-3
- Frequency: 13.56MHz (ISM)
- Range: 10cm
- Data Rates: 100–420kbps

Sigfox



An alternative wide-range technology is [Sigfox](#), which in terms of range comes between WiFi and cellular. It uses the ISM bands, which are free to use without the need to acquire licenses, to transmit data over a very narrow spectrum to and from connected objects. The idea for Sigfox is that for many M2M applications that run on a small battery and only require low levels of data transfer, then WiFi's range is too short while cellular is too expensive and also consumes too much power. Sigfox uses a technology called Ultra Narrow Band (UNB) and is only designed to handle low data-transfer speeds of 10 to 1,000 bits per second. It consumes only 50 microwatts compared to 5000 microwatts for cellular communication, or can deliver a typical stand-by time 20 years with a 2.5Ah battery while it is only 0.2 years for cellular.

Already deployed in tens of thousands of connected objects, the network is currently being rolled out in major cities across Europe, including ten cities in the UK for example. The network offers a robust, power-efficient and scalable network that can communicate with millions of battery-operated devices across areas of several square kilometres, making it suitable for various M2M applications that are expected to include smart meters, patient monitors, security devices, street lighting and environmental sensors. The Sigfox system uses silicon such as the [EZRadioPro wireless transceivers](#) from Silicon Labs, which deliver industry-leading wireless performance, extended range and ultra-low power consumption for wireless networking applications operating in the sub-1GHz band.

- Standard: Sigfox
- Frequency: 900MHz
- Range: 30-50km (rural environments), 3-10km (urban environments)
- Data Rates: 10-1000bps

Neul

IOT Communication Protocol



Similar in concept to Sigfox and operating in the sub-1GHz band, [Neul](#) leverages very small slices of the TV White Space spectrum to deliver high scalability, high coverage, low power and low-cost wireless networks. Systems are based on the Icen1 chip, which communicates using the white space radio to access the high-quality UHF spectrum, now available due to the analogue to digital TV transition. The communications technology is called Weightless, which is a new wide-area wireless networking technology designed for the IoT that largely competes against existing GPRS, 3G, CDMA and LTE WAN solutions. Data rates can be anything from a few bits per second up to 100kbps over the same single link; and devices can consume as little as 20 to 30mA from 2xAA batteries, meaning 10 to 15 years in the field.

- Standard: Neul
- Frequency: 900MHz (ISM), 458MHz (UK), 470-790MHz (White Space)
- Range: 10km
- Data Rates: Few bps up to 100kbps

LoRaWAN



Again, similar in some respects to Sigfox and Neul, [LoRaWAN](#) targets wide-area network (WAN) applications and is designed to provide low-power WANs with features specifically needed to support low-cost mobile secure bi-directional communication in IoT, M2M and smart city and industrial applications. Optimized for low-power consumption and supporting large networks with millions and millions of devices, data rates range from 0.3 kbps to 50 kbps.

- Standard: LoRaWAN
- Frequency: Various
- Range: 2-5km (urban environment), 15km (suburban environment)
- Data Rates: 0.3-50 kbps

IOT Communication Protocol

Difference between TCP and UDP

TCP	UDP
<i>Reliability:</i> TCP is connection-oriented protocol. When a file or message send it will get delivered unless connections fails. If connection lost, the server will request the lost part. There is no corruption while transferring a message.	<i>Reliability:</i> UDP is connectionless protocol. When you a send a data or message, you don't know if it'll get there, it could get lost on the way. There may be corruption while transferring a message.
<i>Ordered:</i> If you send two messages along a connection, one after the other, you know the first message will get there first. You don't have to worry about data arriving in the wrong order.	<i>Ordered:</i> If you send two messages out, you don't know what order they'll arrive in i.e. no ordered
<i>Heavyweight:</i> – when the low level parts of the TCP “stream” arrive in the wrong order, resend requests have to be sent, and all the out of sequence parts have to be put back together, so requires a bit of work to piece together.	<i>Lightweight:</i> No ordering of messages, no tracking connections, etc. It's just fire and forget! This means it's a lot quicker, and the network card / OS have to do very little work to translate the data back from the packets.
<i>Streaming:</i> Data is read as a “stream,” with nothing distinguishing where one packet ends and another begins. There may be multiple packets per read call.	<i>Datagrams:</i> Packets are sent individually and are guaranteed to be whole if they arrive. One packet per one read call.
<i>Examples:</i> World Wide Web (Apache TCP port 80), e-mail (SMTP TCP port 25 Postfix MTA), File Transfer Protocol (FTP port 21) and Secure Shell (OpenSSH port 22) etc.	<i>Examples:</i> Domain Name System (DNS UDP port 53), streaming media applications such as IPTV or movies, Voice over IP (VoIP), Trivial File Transfer Protocol (TFTP) and online multiplayer games etc

IOT Communication Protocol

Networking - Socket Programming

What is Sockets?

Sockets are the endpoints of a bidirectional communications channel. Sockets may communicate within a process, between processes on the same machine, or between processes on different continents.

Sockets may be implemented over a number of different channel types: Unix domain sockets, TCP, UDP, and so on. The *socket* library provides specific classes for handling the common transports as well as a generic interface for handling the rest.

Sockets have their own vocabulary –

S.No.	Term & Description
1	domain The family of protocols that is used as the transport mechanism. These values are constants such as AF_INET, PF_INET, PF_UNIX, PF_X25, and so on.
2	type The type of communications between the two endpoints, typically SOCK_STREAM for connection-oriented protocols and SOCK_DGRAM for connectionless protocols.
3	protocol Typically zero, this may be used to identify a variant of a protocol within a domain and type.
4	hostname

IOT Communication Protocol

	<p>The identifier of a network interface –</p> <ul style="list-style-type: none">• A string, which can be a host name, a dotted-quad address, or an IPV6 address in colon (and possibly dot) notation• A string "<broadcast>", which specifies an INADDR_BROADCAST address.• A zero-length string, which specifies INADDR_ANY, or• An Integer, interpreted as a binary address in host byte order.
5	<p>port</p> <p>Each server listens for clients calling on one or more ports. A port may be a Fixnum port number, a string containing a port number, or the name of a service.</p>

The socket Module

To create a socket, you must use the `socket.socket()` function available in the socket module, which has the general syntax –

```
s = socket.socket (socket_family, socket_type, protocol = 0)
```

Here is the description of the parameters –

- **socket_family** – This is either AF_UNIX or AF_INET, as explained earlier.
- **socket_type** – This is either SOCK_STREAM or SOCK_DGRAM.
- **protocol** – This is usually left out, defaulting to 0.

Once you have `socket` object, then you can use the required functions to create your client or server program. Following is the list of functions required –

Server Socket Methods

S.No.	Method & Description
-------	----------------------

IOT Communication Protocol

1	s.bind() This method binds address (hostname, port number pair) to socket.
2	s.listen() This method sets up and start TCP listener.
3	s.accept() This passively accept TCP client connection, waiting until connection arrives (blocking).

Client Socket Methods

S.No.	Method & Description
1	s.connect() This method actively initiates TCP server connection.

General Socket Methods

S.No.	Method & Description
1	s.recv() This method receives TCP message
2	s.send() This method transmits TCP message
3	s.recvfrom()

IOT Communication Protocol

	This method receives UDP message
4	s.sendto() This method transmits UDP message
5	s.close() This method closes socket
6	socket.gethostname() Returns the hostname.

A Simple Server

```
import socket

# create a socket object
serversocket = socket.socket(
    socket.AF_INET, socket.SOCK_STREAM)

# get local machine name
host = socket.gethostname()

port = 9999

# bind to the port
serversocket.bind((host, port))
```


IOT Communication Protocol

```
# queue up to 5 requests
serversocket.listen(5)

while True:
    # establish a connection
    clientsocket,addr = serversocket.accept()

    print("Got a connection from %s" % str(addr))

    msg='Thank you for connecting'+ "\r\n"
    clientsocket.send(msg.encode('ascii'))
    clientsocket.close()
```

Client side

```
import socket
# create a socket object
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
# get local machine name
host = socket.gethostname()
port = 9999
# connection to hostname on the port.
s.connect((host, port))
# Receive no more than 1024 bytes
msg = s.recv(1024)
s.close()

print (msg.decode('ascii'))
```

IOT Communication Protocol

Output

```
# Following would start a server in background.  
$ python server.py &  
  
# Once server is started run client as follows:  
  
$ python client.py
```

You will get this type of output

```
on server terminal  
Got a connection from ('192.168.1.10', 3747)  
On client terminal  
Thank you for connecting
```

Networking (Socket programming) using UDP

Server side

```
>>> import socket  
  
>>> sock=socket.socket(socket.AF_INET,socket.SOCK_DGRAM)  
  
>>> address=("192.168.1.3",5002)  
  
>>> sock.bind(address)  
  
>>> while True:  
    data,addr=sock.recvfrom(1024)  
    print(data)  
    print(addr)
```

Client side

IOT Communication Protocol

```
>>> import socket  
>>> sock=socket.socket(socket.AF_INET,socket.SOCK_DGRAM)  
Sock.sendto("IOT Training",("192.168.3.1",5002))
```

Now run both program you will get output.

IOT Communication Protocol

HTTP application layer protocol

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

This is the foundation for data communication for the World Wide Web (i.e. internet) since 1990.

HTTP is a generic and stateless protocol which can be used for other purposes as well using extensions of its request methods, error codes, and headers.

Basically, HTTP is a TCP/IP based communication protocol, that is used to deliver data (HTML files, image files, query results, etc.) on the World Wide Web.

The default port is TCP 80, but other ports can be used as well. It provides a standardized way for computers to communicate with each other.

HTTP specification specifies how clients' request data will be constructed and sent to the server, and how the servers respond to these requests.

Basic Features

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

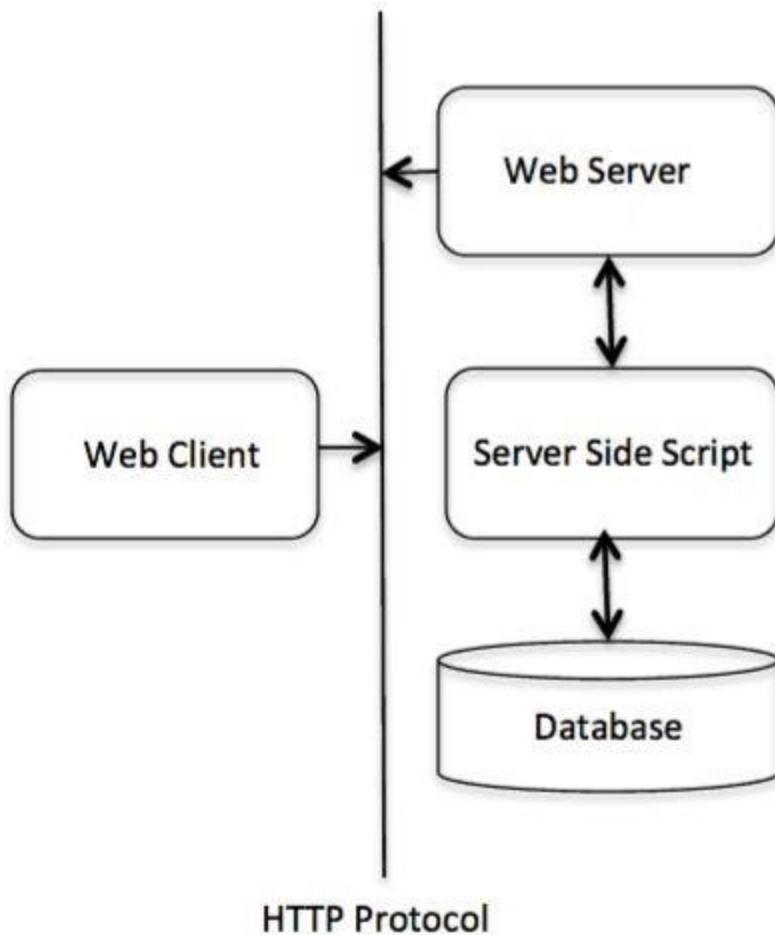
- **HTTP is connectionless:** The HTTP client, i.e., a browser initiates an HTTP request and after a request is made, the client disconnects from the server and waits for a response. The server processes the request and re-establishes the connection with the client to send a response back.
- **HTTP is media independent:** It means, any type of data can be sent by HTTP as long as both the client and the server know how to handle the data content. It is required for the client as well as the server to specify the content type using appropriate MIME-type.

IOT Communication Protocol

- **HTTP is stateless:** As mentioned above, HTTP is connectionless and it is a direct result of HTTP being a stateless protocol. The server and client are aware of each other only during a current request. Afterwards, both of them forget about each other. Due to this nature of the protocol, neither the client nor the browser can retain information between different requests across the web pages.

Basic Architecture

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



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

Client

IOT Communication Protocol

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

Server

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

MQTT protocol

MQTT is a Client Server publish/subscribe messaging transport protocol. It is light weight, open, simple, and designed so as to be easy to implement. These characteristics make it ideal for use in many situations, including constrained environments such as for communication in Machine to Machine (M2M) and Internet of Things (IoT).

Two aspects of MQTT:

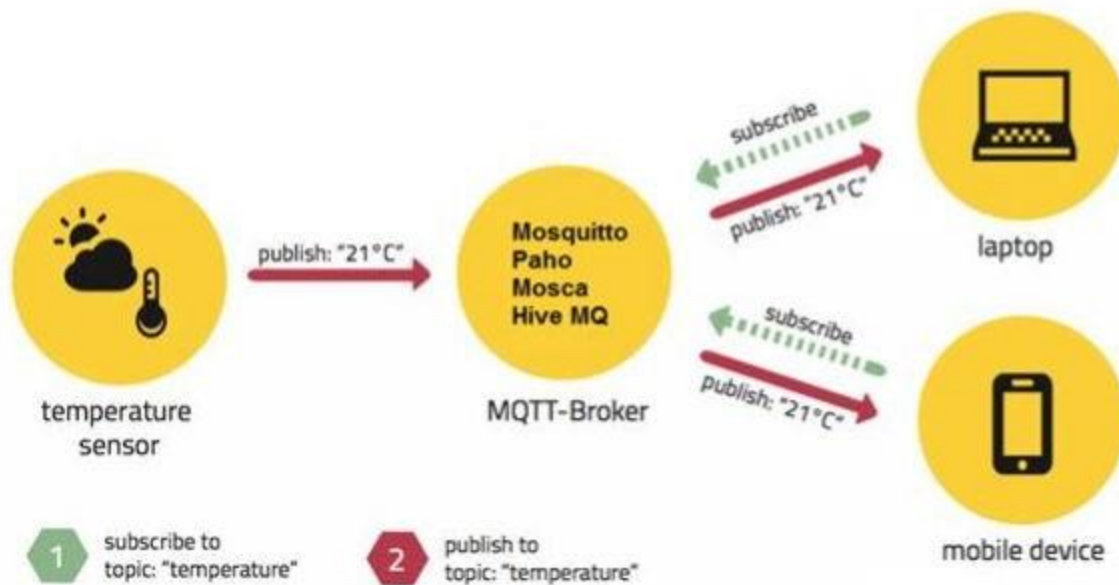
Client

MQTT client includes publisher or subscribers. A MQTT client is any device from a micro controller up to a full fledged server, that has a MQTT library running and is connecting to an MQTT broker over any kind of network

Broker

MQTT broker is the heart of any publish/subscribe protocol. A broker can handle up to thousands of concurrently connected MQTT clients. The broker is primarily responsible for receiving all messages, filtering them, decide who is interested in it and then sending the message to all subscribed clients. It also holds the session of all persisted clients including subscriptions and missed messages. Another responsibility of the broker is the authentication and authorization of clients.

IOT Communication Protocol



How MQTT Works?

The MQTT protocol is based on top of TCP/IP and both client and broker need to have a TCP/IP stack.

The protocol uses a publish/subscribe architecture in contrast to HTTP with its request/response paradigm.

Publish/Subscribe is event-driven and enables messages to be pushed to clients. The central communication point is the MQTT broker, it is in charge of dispatching all messages between the senders and the rightful receivers.

Each client that publishes a message to the broker, includes a topic into the message. The topic is the routing information for the broker.

IOT Communication Protocol

Each client that wants to receive messages subscribes to a certain topic and the broker delivers all messages with the matching topic to the client. Therefore the clients don't have to know each other, they only communicate over the topic.

This architecture enables highly scalable solutions without dependencies between the data producers and the data consumers.

Quality of Service

MQTT defines three levels of Quality of Service (QoS).

The QoS defines how hard the broker/client will try to ensure that a message is received.

Messages may be sent at any QoS level, and clients may attempt to subscribe to topics at any QoS level. This means that the client chooses the maximum QoS it will receive.

For example, if a message is published at QoS 2 and a client is subscribed with QoS 0, the message will be delivered to that client with QoS 0.

If a second client is also subscribed to the same topic, but with QoS 2, then it will receive the same message but with QoS 2.

For a second example, if a client is subscribed with QoS 2 and a message is published on QoS 0, the client will receive it on QoS 0.

Higher levels of QoS are more reliable, but involve higher latency and have higher bandwidth requirements.

- 0: The broker/client will deliver the message once, with no confirmation.
- 1: The broker/client will deliver the message at least once, with confirmation required.
- 2: The broker/client will deliver the message exactly once by using a four step handshake.

MQTT Security Fundamentals: TLS / SSL

MQTT provide TLS and SSL unable secure communication.

IOT Communication Protocol

TLS (Transport Layer Security) and SSL (Secure Sockets Layer) provide a secure communication channel between a client and a server.

At its core, TLS and SSL are cryptographic protocols which use a handshake mechanism to negotiate various parameters to create a secure connection between the client and the server. After the handshake is completed, an encrypted communication between client and server is established and no attacker can eavesdrop any part of the communication. Servers provide a X509 certificate, typically issued by a trusted authority, which clients use to verify the identity of the server.

Contents

- Introduction
- Architecture of WSN
- WSN Node
- Architecture of sensor node
- WSN vs MANET
- Characteristics
- Applications
- Design Challenges
- Advantages
- Disadvantages
- Future
- References
- Conclusion

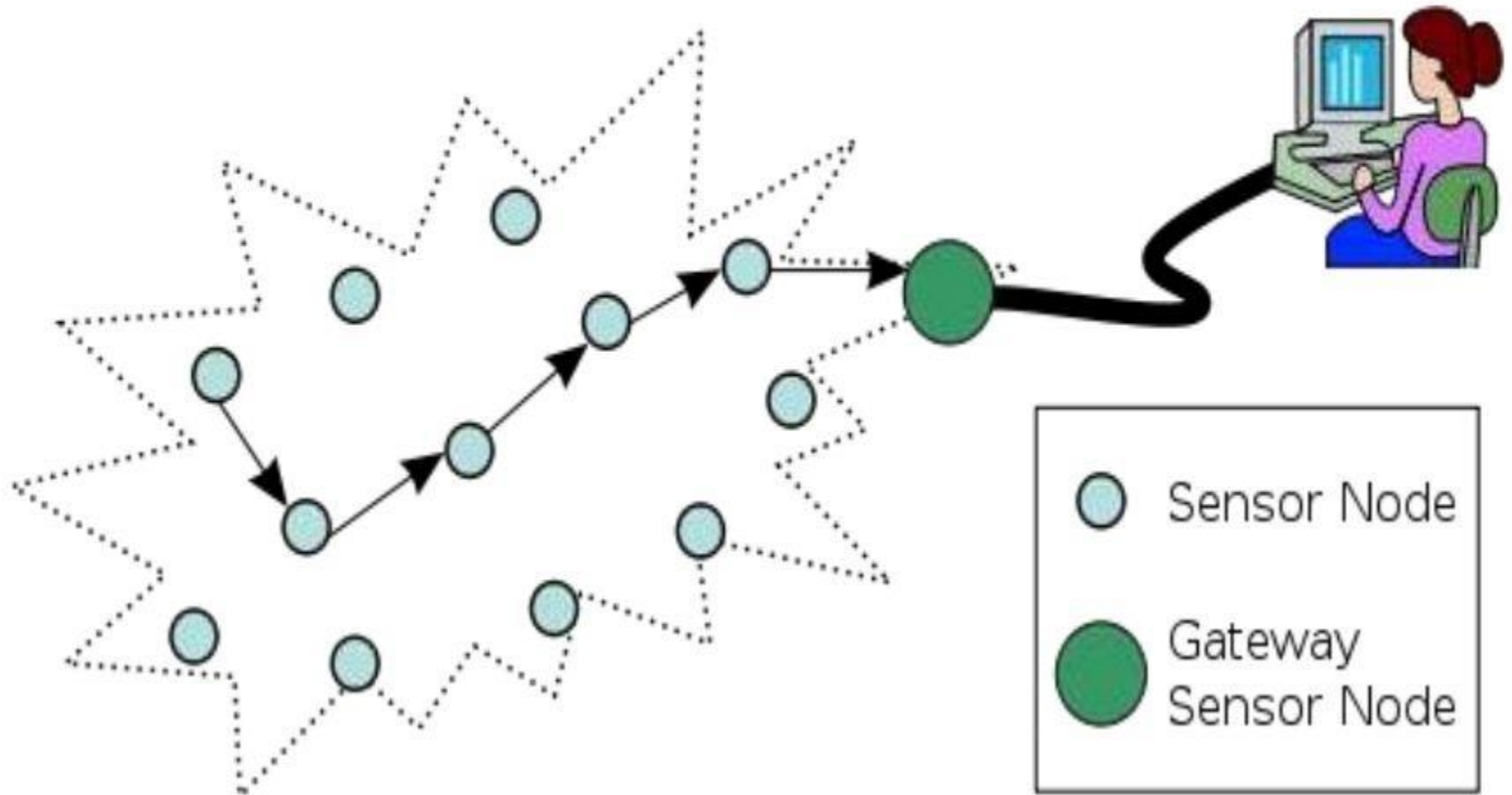
Introduction

- A wireless sensor network (WSN) is a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations.
- A collection of sensing devices that can communicate wirelessly.

Wireless Sensor Networks(WSN)

- Even though wireless sensors has limited resources in memory, computation power, bandwidth, and energy.
- With small physical size. It Can be embedded in the physical environment.
- Self-organizing multi-hop ad-doc networks

Wireless Sensor Network Architecture



Architecture for a WSN

Special addressing requirement

- Local unique addresses
- Data-centric
- *Example: Each node has an unique number.*

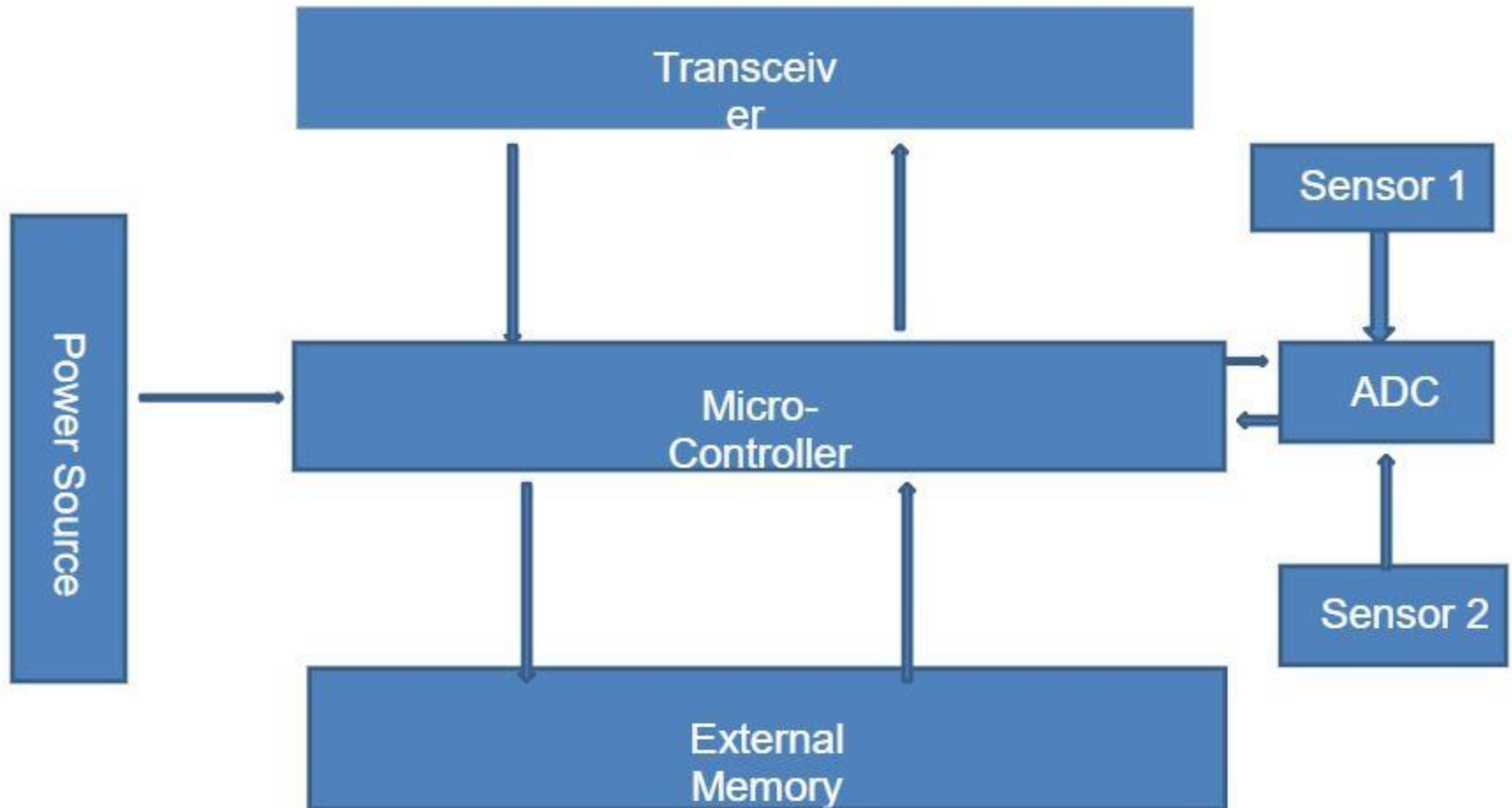
Attribute-based naming architecture

- Data is named by one or more attributes.
- *Example: Each node is distinguished by an attribute – GPS sensors are practical for this.*

Wireless Sensor Node

- **sensor**
 - A transducer
 - converts physical phenomenon e.g. heat, light, motion, vibration, and sound into electrical signals
- **sensor node**
 - basic unit in sensor network
 - contains on-board sensors, processor, memory, transceiver, and power supply
- **sensor network**
 - consists of a large number of sensor nodes
 - nodes deployed either inside or very close to the sensed phenomenon

Architecture of Sensor Node



Data Aggregation in WSNs

- Solves implosion and overlap problem
- Energy efficient

Wireless Sensor Network(WSN) vs. Mobile Ad Hoc Network (MANET)

	WSN	MANET
Similarity	Wireless	Multi-hop networking
Security	Symmetric Key Cryptography	Public Key Cryptography
Routing	Support specialized traffic pattern. Cannot afford to have too many node states and packet overhead	Support any node pairs Some source routing and distance vector protocol incur heavy control traffic
Resource	Tighter resources (power, processor speed, bandwidth)	Not as tight.

Characteristics

- Power consumption constraints for nodes using batteries or energy harvesting
- Ability to cope with node failures (resilience)
- Mobility of nodes
- Heterogeneity of nodes
- Scalability to large scale of deployment
- Ability to withstand harsh environmental conditions
- Ease of use
- Cross-layer design

Factors Influencing WSN Design

- Fault tolerance
- Scalability
- Production costs
- Hardware constraints
- Sensor network topology
- Environment
- Transmission media
- Power Consumption
 - Sensing
 - Communication

- Data processing

Applications

- Military Applications
- Environmental Applications
- Health Applications
- Home and Office Applications
- Automotive Applications
- Other Commercial Applications

Advantages

- It avoids a lot of wiring .
- It can accommodate new devices at any time .
- It's flexible to go through physical partitions .
- It can be accessed through a centralized monitor

Disadvantages

- Lower speed compared to wired network.
- Less secure because hacker's laptop can act as Access Point. If you connected to their laptop, they'll read all your information (username, password.. etc).
- More complex to configure than wired network.
- Gets distracted by various elements like Blue-tooth .
- Still Costly at large.
- It does not make sensing quantities in buildings easier.
- It does not reduce costs for installation of sensors.
- It does not allow us to do more than can be done with a wired system

Design Challenges

- **Heterogeneity**
 - The devices deployed may be of various types and need to collaborate with each other.
- **Distributed Processing**
 - The algorithms need to be centralized as the processing is carried out on different nodes.
- **Low Bandwidth Communication**
 - The data should be transferred efficiently between sensors

Continued..

- **Large Scale Coordination**
 - The sensors need to coordinate with each other to produce required results.
- **Utilization of Sensors**
 - The sensors should be utilized in a ways that produce the maximum performance and use less energy.
- **Real Time Computation**
 - The computation should be done quickly as new data is always being generated.

Operational Challenges of Wireless Sensor Networks

- Energy Efficiency
- Limited storage and computation
- Low bandwidth and high error rates
- Errors are common
 - Wireless communication
 - Noisy measurements
 - Node failure are expected
- Scalability to a large number of sensor nodes
- Survivability in harsh environments
- Experiments are time- and space-intensive

Future of WSN

Smart Home / Smart Office



- Sensors controlling appliances and electrical devices in the house.
- Better lighting and heating in office buildings.
- The Pentagon building has used sensors extensively

Wireless Sensor Network And Its Application



Outline

- Introduction
- Sensor Node Components
- Wireless Sensor Network
- A general work process of WSN
- Sensor Networks Operating Systems
- Wireless Sensor Networks Applications
- Conclusion
- References

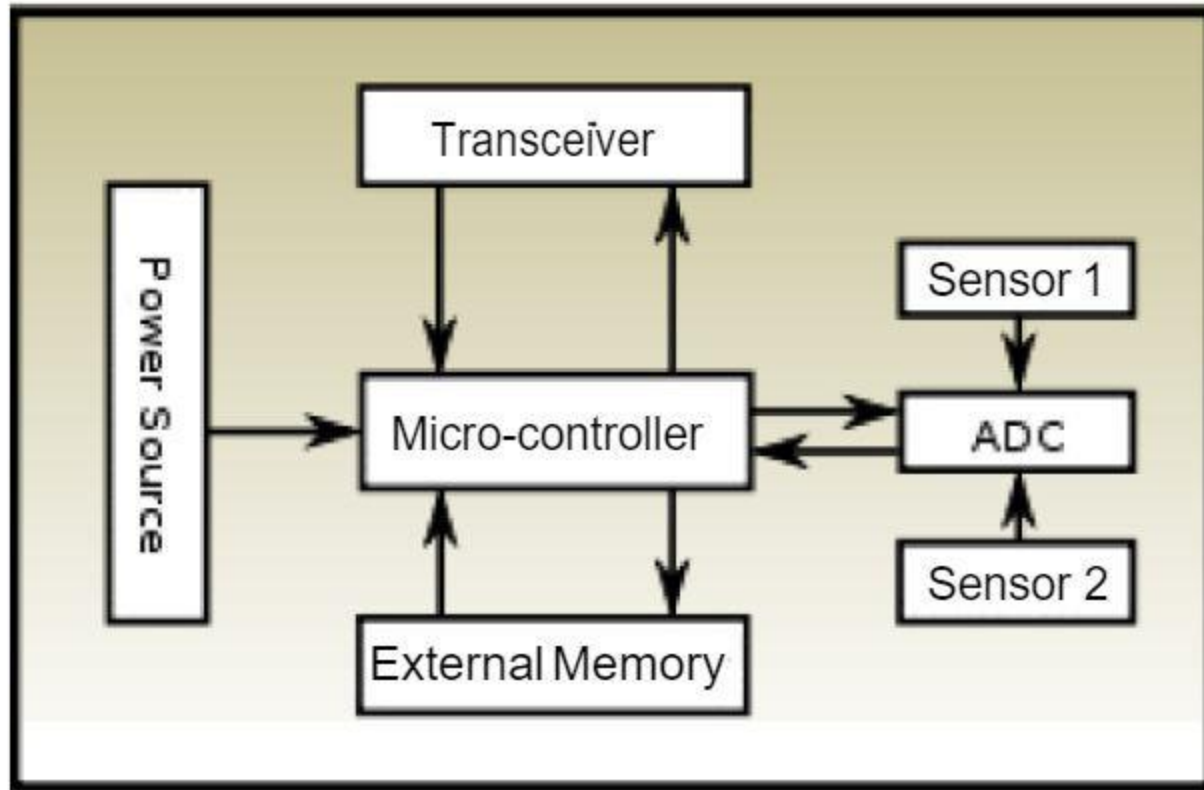


sensor and Sensor Node

- A **sensor** is a electronic device that measures a physical quantity and converts it into a signal which can be read by an observer or by an instrument.
- **Sensor Node** : Basic unit in Sensor Network



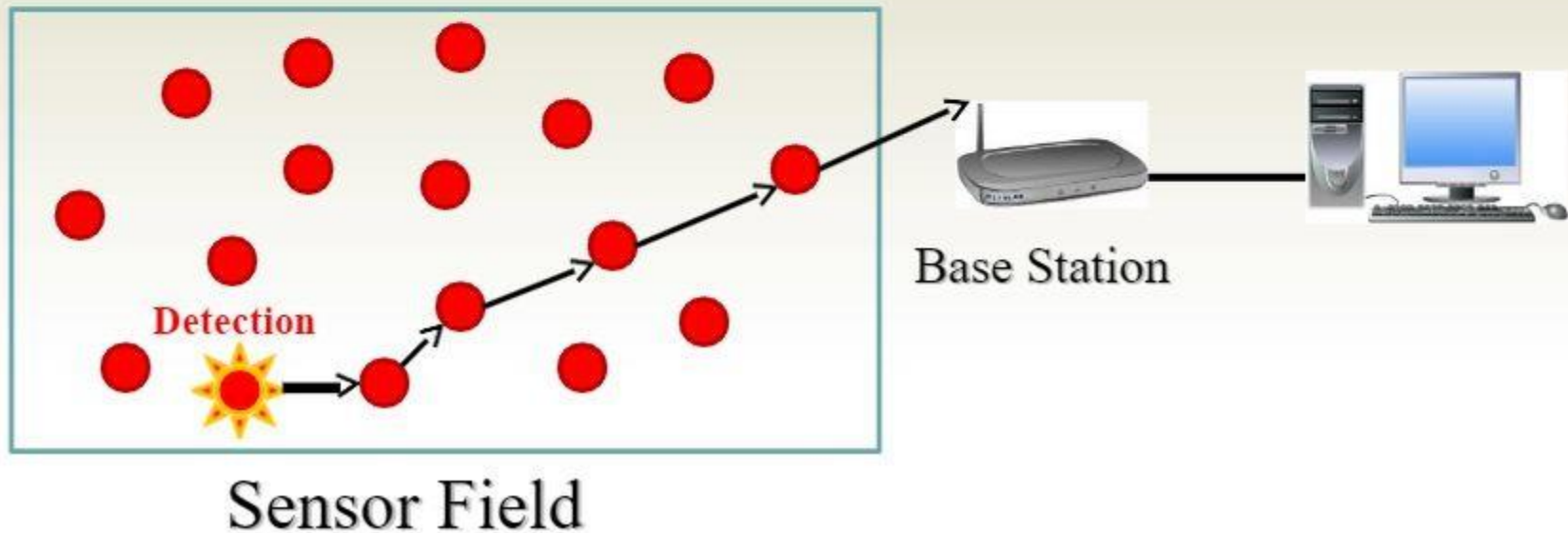
Sensor Node components



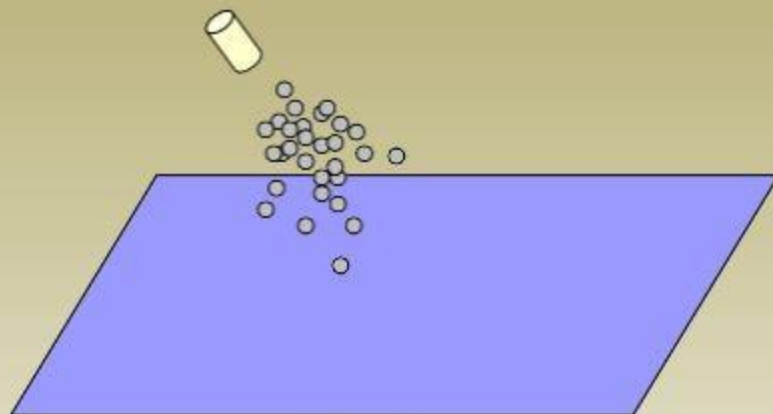


Wireless sensor Network (WSN)

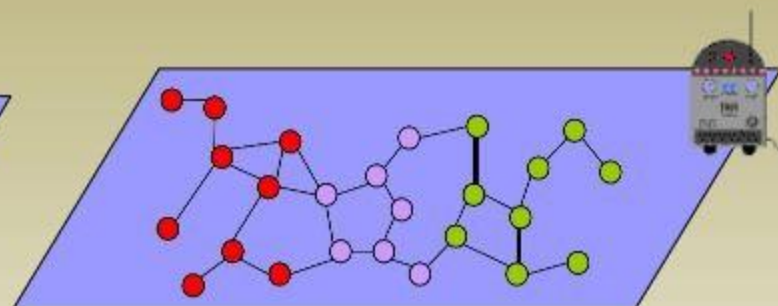
A wireless sensor network is a collection of nodes “sensors” organized into a cooperative network. The nodes **communicate wirelessly** and often self-organize after being deployed.



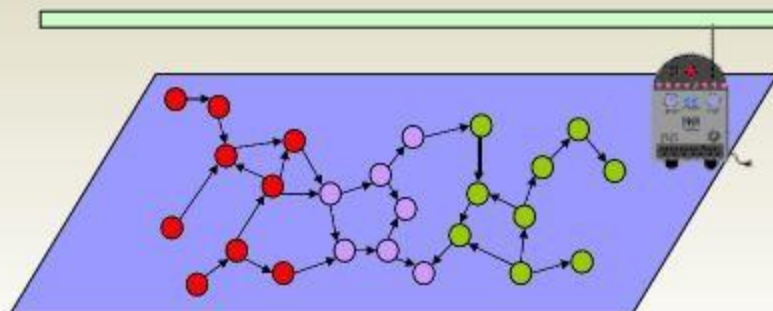
A general work process of WSN



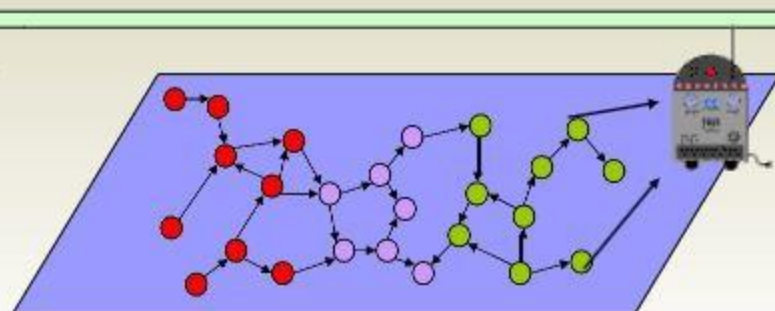
Deploy



Organize into network



Sensing and monitoring



**Data collection and
send it to base station**

Goal of the sensor node

The goal from the sensor node is to collect the data at regular intervals, then transform the data into an electrical signal and finally send the signal to the sink or the base node.





Sensor Networks Operating Systems

- Sensor Network uses TinyOS.
- It is a free open source operating system designed for wireless sensor networks.
- It is an embedded operating system written in NesC programming language, developed in 1999.





Need of TinyOS

- Problems with traditional OS
 - Multithreaded Architecture not useful
 - Large Memory Footprint
 - Does not help to conserve energy and power
- Requirements for Wireless Sensor Networks
 - Efficient utilization of energy and power
 - Small Footprint
 - Should support diversity in design and usage
 - More emphasis on Concurrent execution

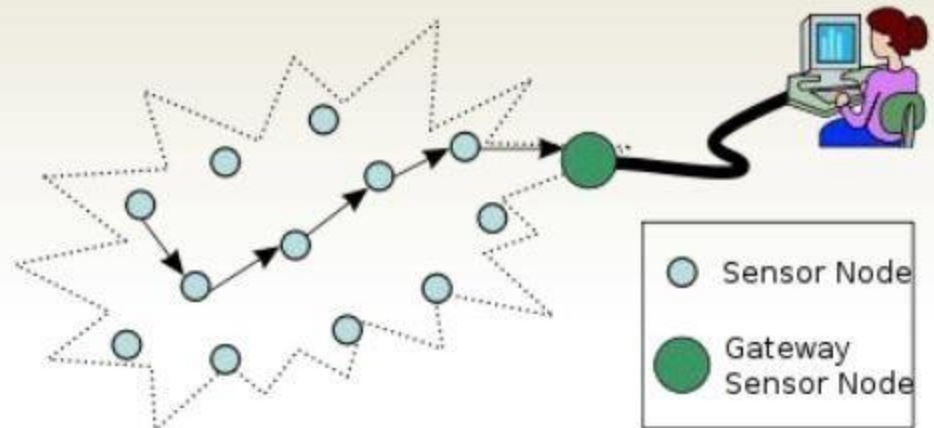


TinyOS Features

- Component based architecture allows frequent changes while still keeping the size of code minimum.
- Event based execution model means no user/kernel boundary and hence supports high concurrency.
- It is power efficient as it makes the sensors sleep as soon as possible.
- Has small footprint as it uses a non-preemptable FIFO task scheduling.

Wireless Sensor Networks Applications

- Forest fire detection
- Air pollution monitoring
- Water quality monitoring
- Land slide detection
- Automotive application
- Military application



Forest fire detection

- A network of Sensor Nodes can be installed in a forest to detect when a fire has started.
- The nodes can be equipped with sensors to measure temperature, humidity and gases which are produced by fire in the trees or vegetation.
- If the node detects fire, it sends an alarm message (along with its location) to the base station





Air pollution monitoring

- Traditional air quality monitoring methods, such as building air quality monitoring stations, are typically expensive.
- The solution to these is air quality monitoring system based on the technology of wireless sensor networks (WSNs).
- Wireless sensor networks have been deployed in several cities to monitor the concentration of dangerous gases for citizens.

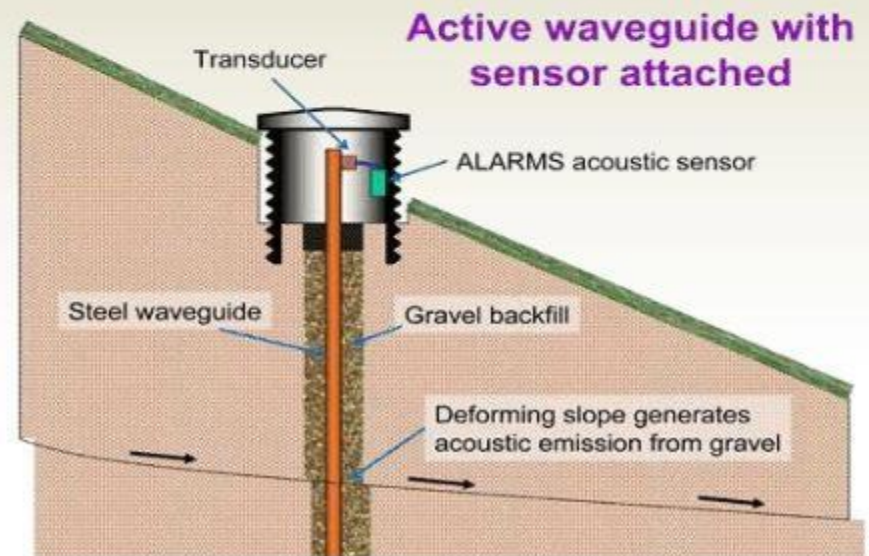


Water quality monitoring

- Water quality monitoring involves analyzing water properties in dams, rivers, lakes & oceans, as well as underground water reserves.
- Parameters considered include – temperature, turbidity and pH

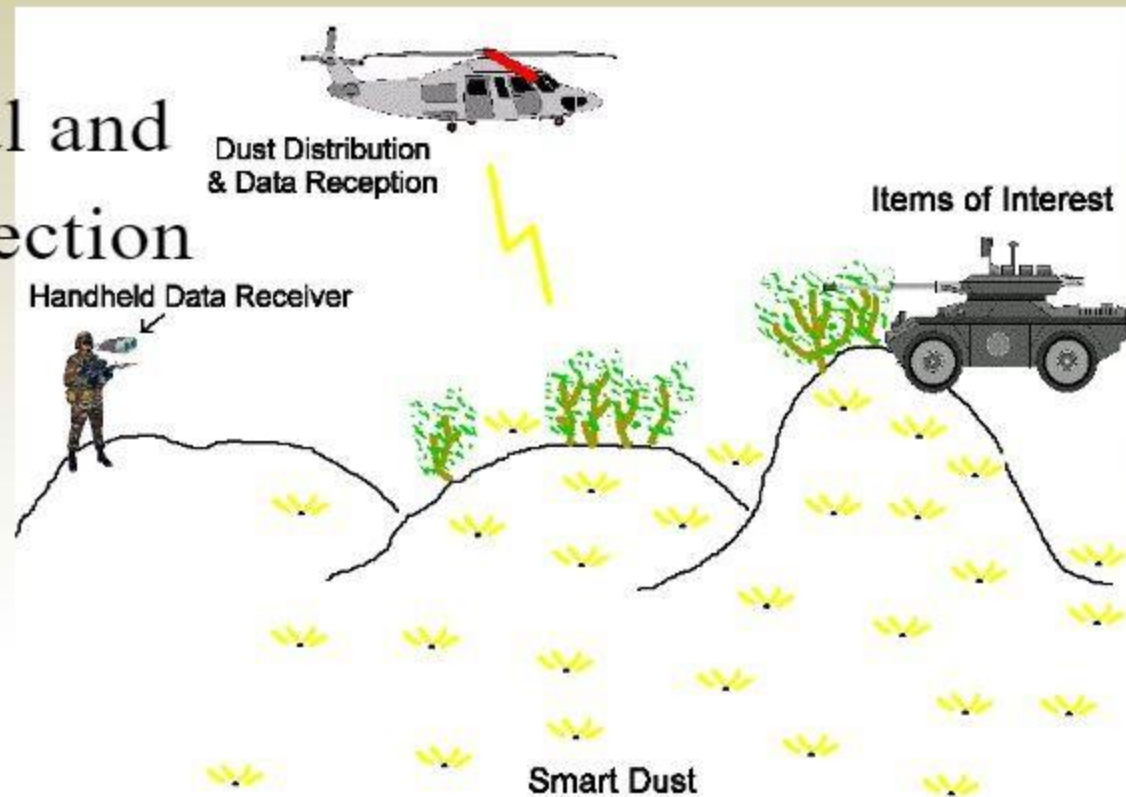
Land slide Detection

- A landslide detection system makes use of a wireless sensor network to detect the slight movements of soil and changes in various parameters that may occur before or during a landslide.
- Through the data gathered it may be possible to know the occurrence of landslides long before it actually happens.



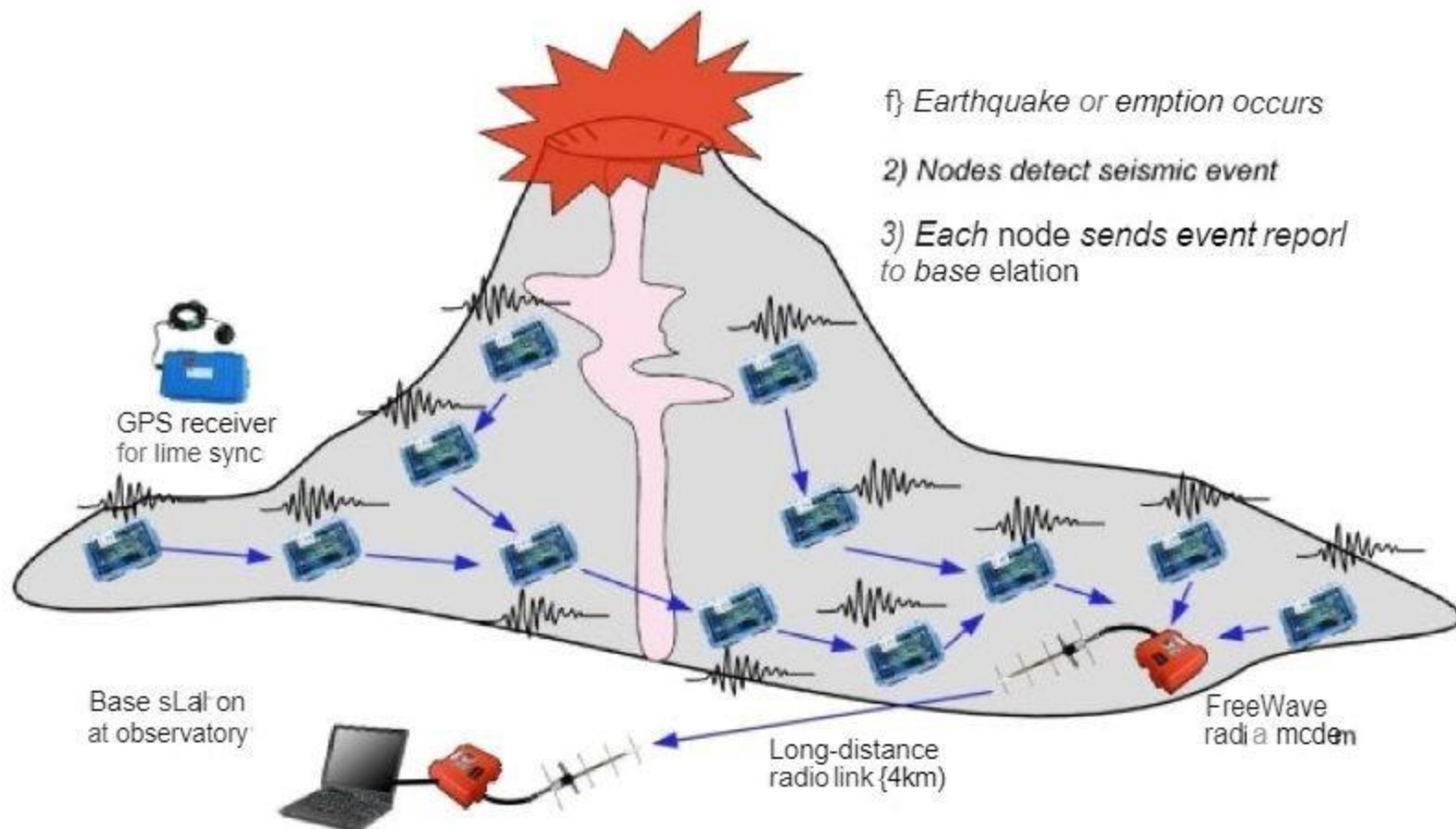
Military Surveillance

- enemy tracking, battlefield surveillance
- target detection
- Monitoring, tracking and surveillance of borders
- Nuclear, biological and chemical attack detection





Eruption



FUTURE SCOPE



They receive live information from the road authority about the state of the roads including traffic jams, accidents and weather. The car transmits information to the road authority regarding speed, distance travelled, use of windscreen wipers, etc.



Conclusions

- WSN are a widely applicable, major emerging technology.
- WSN are getting smaller and faster, increasing their potential applications in commercial, industrial, and residential environments.
- However, the limit of applications depends only upon the sensors used and the interpretation of the data obtained.

What is the M2M Communication?

Today we are talking about a new concept related to Industry 4.0 that, in recent times, is becoming increasingly important, as the rest of the technologies in this area. This is the M2M or Machine to Machine communication.

The machine to machine concept represents any technology that allows two devices to exchange information with each other, for example, communicate and send data. The communication that occurs between the machines or devices is autonomous, there is no need for human intervention for this data exchange to take place.

M2M connectivity is related to the Internet of Things (IoT). Both are part of the same concept and complement each other. Thanks to IoT, a system of machines or interrelated devices can be connected wirelessly, and exchange and analyze data automatically in the cloud. In short, IoT is enabled by integrating many M2M devices and using cloud web platforms to process all that data.

What types of connectivity are used in M2M?

There are different types of connectivity between machines, and it is possible that you already know almost everyone. But, what are the most used? First, we have the **RFID**, or radiofrequency identification. The limitation of this type of connectivity is that it has a maximum range of 10 meters. On the other hand, there are **Bluetooth** and **WiFi** that also have a limited range, from 10 to 20 meters in the case of Bluetooth and 50 meters in the case of WiFi. These types of connectivity are short range if we compare them with the following. Connectivity using **low frequency** has a range of up to 1,000 km and the **GSM** network (using SIM cards) or the satellite is worldwide. As you can see, there are many different options that allow us adapting to each problem.

applications of M2M

The applications and areas in which connectivity between machines can be applied and used are very wide.

For example, the connected *vending* machines allow the distributor to know their replacement status and to notify in cases which some product runs out.

It is also very useful in the health area. Telemedicine is a concept already implemented in some places and has meant great improvements in this area. In hospitals, processes are automated to improve efficiency and safety, for example, using devices capable of reacting faster than humans. If a patient has a drop in vital signs and is connected to an M2M device, the machine can automatically administer extra oxygen before the hospital staff reaches it.

Likewise, it is also **used in the industry**, allowing machines to be connected to each other and sending data each other. With this data, they can optimize processes automatically, notify when a machine has a breakdown or even self-repair.

In general we can establish the following industrial applications:

- Automated maintenance
- Procedure for requesting spare parts
- End of process notice
- Data collection for processing by other equipment
- Intelligent stock control

- Implementation of just-in-time systems

Programming with Arduino

Arduino Coding Basics

We have already discussed the popular Arduino Boards, Arduino IDEs, and Installation process of the Arduino software. We learned that [Arduino IDE](#) (Integrated Development Environment) allows us to draw the sketch and upload it to the various [Arduino boards](#) using code. The code is written in a simple programming language similar to [C](#) and [C++](#).

The initial step to start with Arduino is the IDE download and installation.

Let's discuss the basics to start with [Arduino](#) programming.

Brackets

There are two types of brackets used in the Arduino coding, which are listed below:

- Parentheses ()
- Curly Brackets { }

Parentheses ()

The parentheses brackets are the group of the arguments, such as method, function, or a code statement. These are also used to group the math equations.

Curly Brackets { }

The statements in the code are enclosed in the curly brackets. We always require closed curly brackets to match the open curly bracket in the code or sketch.

Open curly bracket- ' { '

Closed curly bracket - ' } '

Line Comment

There are two types of line comments, which are listed below:

- Single line comment
- Multi-line comment

// Single line comment

The text that is written after the two forward slashes are considered as a single line comment. The compiler ignores the code written after the two forward slashes. The comment will not be displayed in the output. Such text is specified for a better understanding of the code or for the explanation of any code statement.

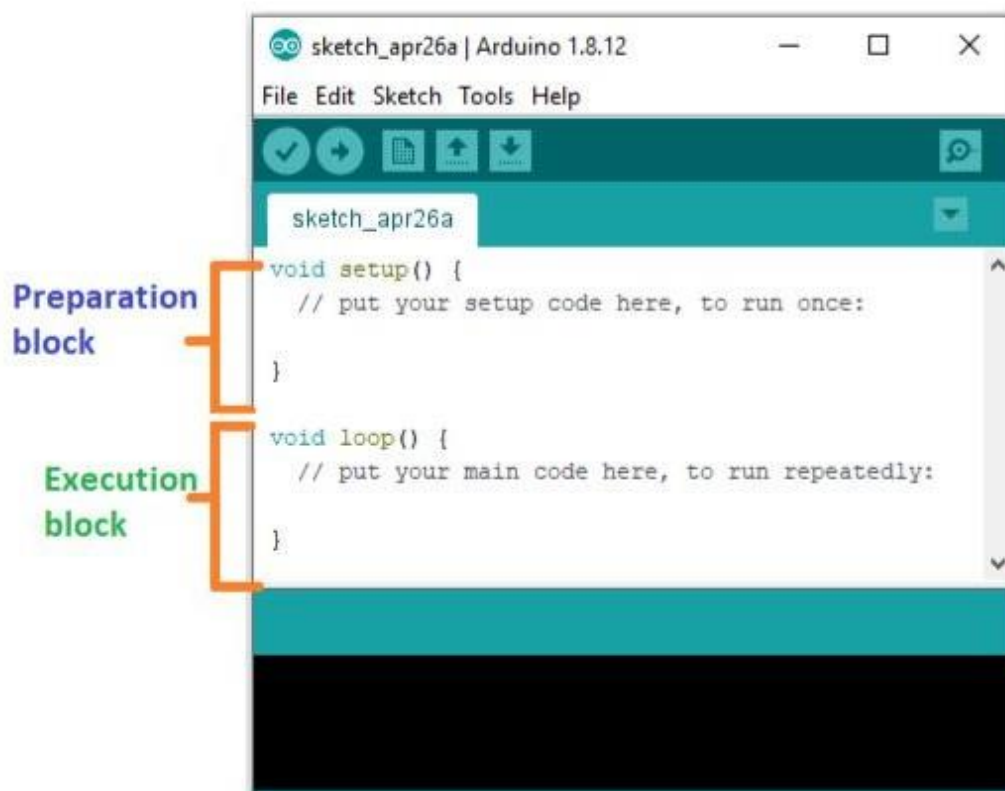
The `//` (two forward slashes) are also used to ignore some extra lines of code without deleting it.

`/* Multi - line comment */`

The Multi-line comment is written to group the information for clear understanding. It starts with the single forward slash and an asterisk symbol (`/*`). It also ends with the `/*`. It is commonly used to write the larger text. It is a comment, which is also ignored by the compiler.

Coding Screen

The coding screen is divided into two blocks. The **setup** is considered as the preparation block, while the **loop** is considered as the execution block. It is shown below:



The set of statements in the setup and loop blocks are enclosed with the curly brackets. We can write multiple statements depending on the coding requirements for a particular project.

For example:

1. **void** setup ()
2. {
3. Coding statement 1;
4. Coding statement 2;
5. .
6. .
7. .
8. Coding statement n;
9. }
10. **void** loop ()
11. {
12. Coding statement 1;
13. Coding statement 2;
14. .
15. .
16. .
17. Coding statement n;
18. }

What is Setup? What type of code is written in the setup block?

It contains an initial part of the code to be executed. The pin modes, libraries, variables, etc., are initialized in the setup section. It is executed only once during the uploading of the program and after reset or power up of the Arduino board.

Zero setup () resides at the top of each sketch. As soon as the program starts running, the code inside the curly bracket is executed in the setup and it executes only once.

What is Loop? What type of code is written in the Loop block?

The loop contains statements that are executed repeatedly. The section of code inside the curly brackets is repeated depending on the value of variables.

Time in Arduino

The time in Arduino programming is measured in a millisecond.

Where, 1 sec = 1000 milliseconds

We can adjust the timing according to the milliseconds.

For example, for a 5-second delay, the time displayed will be 5000 milliseconds.

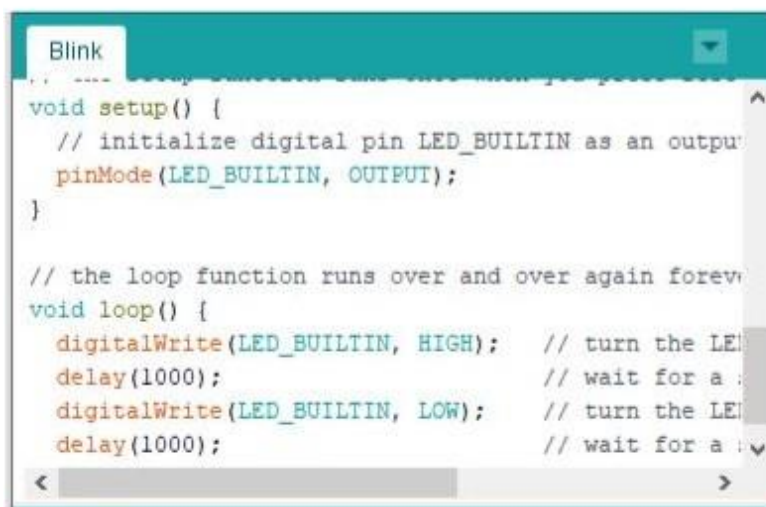
Example:

Let's consider a simple [LED blink](#) example.

The steps to open such example are:

1. Click on the **File** button, which is present on the menu bar.
2. Click on the **Examples**.
3. Click on the **Basics** option and click on the **Blink**

The example will reopen in a new window, as shown below:

A screenshot of the Arduino IDE interface. At the top, there is a teal header bar with the word "Blink" on the left and a dropdown arrow on the right. Below the header, the code editor displays the following C++ code:

```
void setup() {  
  // initialize digital pin LED_BUILTIN as an output  
  pinMode(LED_BUILTIN, OUTPUT);  
}  
  
// the loop function runs over and over again forever  
void loop() {  
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the positive voltage)  
  delay(1000); // wait for a second  
  digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the pin LOW (no voltage)  
  delay(1000); // wait for a second  
}
```

The code is color-coded: keywords like 'void', 'setup', 'loop', 'pinMode', 'digitalWrite', and 'delay' are in blue; comments are in grey; and string literals and constants are in red. The editor has a scrollbar on the right side.

- o The void setup () would include pinMode as the main function.

pinMode ()

The specific pin number is set as the INPUT or OUTPUT in the pinMode () function.

The Syntax is: **pinMode (pin, mode)**

Where,

pin: It is the pin number. We can select the pin number according to the requirements.

Mode: We can set the mode as INPUT or OUTPUT according to the corresponding pin number.

Let's understand the pinMode with an example.

Example: We want to set the 12 pin number as the output pin.

Code:

1. pinMode (12, OUTPUT);

Why is it recommended to set the mode of pins as OUTPUT?

The OUTPUT mode of a specific pin number provides a considerable amount of current to other circuits, which is enough to run a sensor or to light the [LED](#) brightly. The output state of a pin is considered as the low-impedance state.

The high current and short circuit of a pin can damage the ATmel chip. So, it is recommended to set the mode as OUTPUT.

Can we set the pinMode as INPUT?

The digitalWrite () will disable the LOW during the INPUT mode. The output pin will be considered as HIGH.

We can use the INPUT mode to use the external pull-down resistor. We are required to set the pinMode as INPUT_PULLUP. It is used to reverse the nature of the INPUT mode.

The sufficient amount of current is provided by the pull-up mode to dimly light an LED, which is connected to the pin in the INPUT mode. If the LED is working dimly, it means this condition is working out.

Due to this, it is recommended to set the pin in OUTPUT mode.

- o The void loop () would include **digitalWrite()** and **delay ()** as the main function.

digitalWrite()

The digitalWrite () function is used to set the value of a pin as HIGH or LOW.

Where,

HIGH: It sets the value of the voltage. For the 5V board, it will set the value of 5V, while for 3.3V, it will set the value of 3.3V.

LOW: It sets the value = 0 (GND).

If we do not set the pinMode as OUTPUT, the LED may light dim.

The syntax is: **digitalWrite(pin, value HIGH/LOW)**

pin: We can specify the pin number or the declared variable.

Let's understand with an example.

Example:

1. digitalWrite (13, HIGH);
2. digitalWrite (13, LOW);

The HIGH will ON the LED and LOW will OFF the LED connected to pin number 13.

What is the difference between digitalWrite () and digitalWrite ()?

The digitalWrite () function will read the HIGH/LOW value from the digital pin, and the digitalWrite () function is used to set the HIGH/LOW value of the digital pin.

delay ()

The delay () function is a blocking function to pause a program from doing a task during the specified duration in milliseconds.

For example, - delay (2000)

Where, 1 sec = 1000millisecond

Hence, it will provide a delay of 2 seconds.

Code:

1. digitalWrite (13, HIGH);
2. delay (2000);
3. digitalWrite (13, LOW);
4. delay (1000);

Here, the LED connected to pin number 13 will be ON for 2 seconds and OFF for 1 second. The task will repeatedly execute as it is in the void loop ().

We can set the duration according to our choice or project requirements.

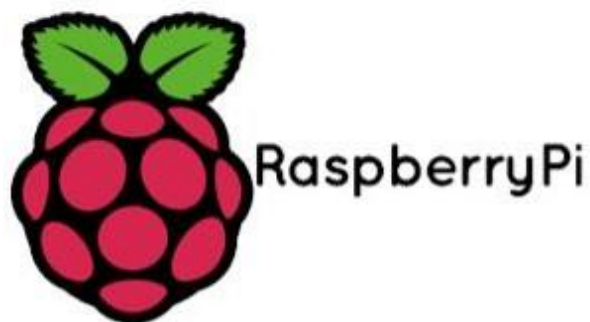
Example: To light the LED connected to pin number 13. We want to ON the LED for 4 seconds and OFF the LED for 1.5 seconds.

Code:

```
1. void setup ()
2. {
3.   pinMode ( 13, OUTPUT); // to set the OUTPUT mode of pin number 13.
4. }
5. void loop ()
6. {
7.   digitalWrite (13, HIGH);
8.   delay (4000); // 4 seconds = 4 x 1000 milliseconds
9.   digitalWrite (13, LOW);
10.  delay (1500); // 1.5 seconds = 1.5 x 1000 milliseconds
11. }
```

Programming with Raspberry Pi

It is a cheap, credit-card-sized device that uses a daily keyboard and mouse and joins to a TV or computer monitor. It is a thin weighable computer that let every person of all ages to discover programming and gain how to programme in variant languages like Python and Scratch. From exploring the internet and watching high-definition video, word-processing, to creating spreadsheets, and it can do every possible thing we'd expect a desktop computer to do and playing sports.



Including, the device has the potential to link with the outer world and has been used in a large variety of communicative developer ventures, including music machines and infra-red cameras tweeting birdhouses and parent detectors to weather stations.

Raspberry Pi (/pa ʔi /) is a span of slight single-board computers created in the United Kingdom in partnership with Broadcom by the Raspberry Pi Foundation. Earlier, the Raspberry Pi device stress on supporting basic computer science instructing in schools and in progressing countries. The real type later became much more popular than targeted, selling for works such as robotics the target outside market.

Because of its low cost, modularity, and open architecture, it is now commonly used in many fields, such as for weather forecasting.

The Foundation started a new company, called Raspberry Pi Trading, after the let out of the second form of board, and installed Eben Upton as CEO, with the control of creating technology. In schools and progressing countries, the Foundation has been dedicated again as an institutional charity to cheer the instructing of basic computer science.

Raspberry Pi became the Best-selling British computers. As on December 2019, over More than 30 million boards have been sold. Large percentage of Pis are produced in a Sony factory in Pencoed, Wales, while others are manufactured in China and Japan.

Raspberry Pi Foundation

A listing institutional charity (registration number 1129409) located in the UK is the Raspberry Pi Foundation. The target of the Foundation is to emphasis on the education of adults, and children.

The Foundation of Raspberry Pi is an aid based in the United Kingdom that targets to get the authority of computing and digital development into the hands of people in the world. The target is to more people can leverage the power of computing and appearing technology for work, solve and express themselves creatively in order to solve concerns that really matters.

What we will need?



SD card

We recommend a class 4 or class 10 microSD card of at least 8 GB. People can get a card which is pre-existed with NOOBS or Raspberry Pi OS to save the time but it's easy to establish self-card.

Display and connectivity cable

Any HDMI/DVI monitor or TV can work as a display for the Pi. Use a monitor with an HDMI input for the best results; other link types are also available for older devices.

Keyboard and mouse

Any regular USB [keyboard](#) and [mouse](#) will work with Raspberry Pi. If already paired, wireless keyboards and mice can run. For configuration options for keyboard configurations, see [raspi-config](#).

Power supply

Pi is represented by a [USB](#) Micro or [USB](#) Type-C for power supply.

Like Model 4B, 2A at 5V for 3B and 3B+, or 700mA at 5V for the earlier, low-powered Pi versions, we require good-quality power connection that can facilitate legit 3A at 5V. We suggest that one must use the real Raspberry Pi power supply that is specifically made for Raspberry Pi.

Low-current power supplies can work for normal, but if it gets too much supply, it might cause the Pi to restart. For work with the Pi 3 or 4, they are not comfortable.

Optional Peripheral

[Model B/B+/2B/3B/3B+/4B only] Ethernet (network) cable for linking our Pi to an area network or to the internet, an Ethernet cable will be required.

Wireless USB dongle

Required only if we require wireless connectivity and operating an older model without wireless qualities pre-installed.

Audio lead

It can be played by speakers or headphones working with a standard 3.5mm jack. An audio lead is required without an HDMI cable to produce sound.

An [HDMI](#) cable to link to a monitor with speakers, no separate audio lead is needed, so audio can be worked straight way by the monitor; but if we decide to get the audio worked by other more speakers, can linked one - it needs layout.

Software defined Networking

In order to understand software defined networks, we need to understand the various planes involved in networking.

Data plane:

All the activities involving as well as resulting from data packets sent by the end user belong to this plane. This includes:

- Forwarding of packets
- Segmentation and reassembly of data
- Replication of packets for multicasting

Control plane:

All activities necessary to perform data plane activities but do not involve end user data packets belong to this plane. In other words, this is the brain of the network. The activities of the control plane include:

- Making routing tables
- Setting packet handling policies

In a traditional network, each [switch](#) has its own data plane as well as control plane. The control plane of various switches exchange [topology](#) information and hence construct a forwarding table which decides where an incoming data packet has to be forwarded via the data plane.

Software defined networking (SDN) is an approach via which we take the control plane away from the switch and assign it to a centralised unit called the SDN controller. Hence, a network administrator can shape traffic via a centralised console without having to touch the individual switches. The data plane still resides in the switch and when a packet enters a switch, its forwarding activity is decided based on the entries of flow tables, which are pre-assigned by the controller. A flow table consists of match fields (like input port number and packet header) and instructions. The packet is first matched against the match fields of the flow table entries.

Then the instructions of the corresponding flow entry are executed. The instructions can be forwarding the packet via one or multiple ports, dropping the packet or adding headers to the packet. If a packet doesn't find a corresponding match in the flow table, the switch queries the controller which sends a new flow entry to the switch. The switch forwards or drops the packet based on this flow entry.

A typical SDN architecture consists of three layers.

- **Application layer:**

It contains the typical network applications like [intrusion detection](#), [firewall](#), and [load balancing](#)

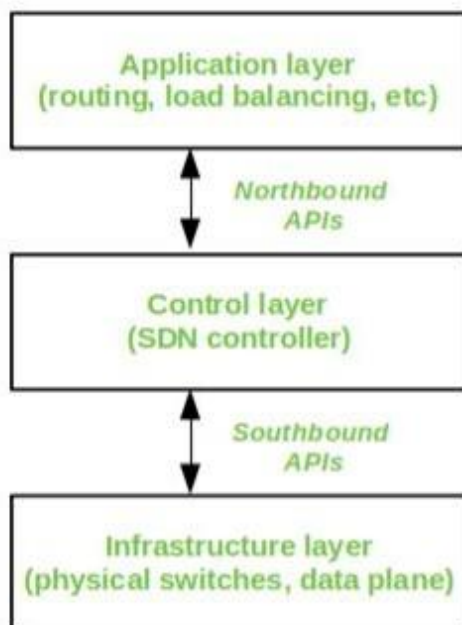
- **Control layer:**

It consists of the SDN controller which acts as the brain of the network. It also allows hardware abstraction to the applications written on top of it.

- **Infrastructure layer:**

This consists of physical switches which forms the data plane and carries out actual movement of data packets.

The layers communicate via a set of interfaces called the northbound APIs(between application and control layer) and southbound APIs(between control and infrastructure layer).



SDN architecture:

Advantages of SDN:

- Network is programmable hence can easily be modified via the controller rather than individual switches.
- Switch hardware becomes cheaper since each switch only needs a data plane.
- Hardware is abstracted, hence applications can be written on top of controller independent of switch vendor.
- Provides better security since the controller can monitor traffic and deploy security policies. For example, if the controller detects suspicious activity in network traffic, it can reroute or drop the packets.

Disadvantages of SDN:

The central dependency of the network means single point of failure, i.e. if the controller gets corrupted, the entire network will be affected.

Smart Home Technology





Introduction

- What is Smart Home?
 - *It is one of the applications of Smart technologies within our home, in creating a comfortable home and an effective lifestyle;*
 - *It improves automation control and monitoring of household devices and*
 - *And connect via standard means of communication for its operation.*
- Features of Smart Home
- Why Smart Home: the benefits
- Conclusion

Features of Smart Home

Few of the features include but limited to:

- Home Security System
- Controlled Exterior & Interior Lighting
- Automatic Window and Curtain
- Automatic Garage Door
- Distributed Entertainment Systems
(*Video and Audio systems*)
- Intercom/Telephony Systems
- Home Energy management systems



Figure 1a

Why Smart Home

Smart Home is applicable in various aspect of our home and environment to suite our different preferences.

- Comfort or Ease of control
- Entertainment
- Security
- convenience at all times,
- lifestyle personalisation and
- User friendly



Figure 2a

Why Smart Home: **Comfort/Ease of Control**

Smart Home offers ease of controlling devices around the home. This is one of its outstanding feature.

These features offers automation and remote control of devices around the home from any location within such as:

- Automatic control of garage door and gate(s);
- Automatic shut down of appliance when not in use;
- Automatic setting and maintenance of right temperature for each room;
- Automatically adjust/ regulate light intensity

based on room luminosity.



Figure 3a



Figure 3b

Why Smart Home: **Entertainment**

from any part in the house.

Easy Control

- Parental control feature on television channels for children to watch only certain programmes.

Digital Access

- Access favourite CD's, DVD's from any music system and television around the home;

Digital Storage

- System automatically stores most played music based on number of play.



observation of activities

- Capture and record video surveillance.

Emergency

- Initiate emergency alarm or audible intruder alert
- Activate sprinkler and fire extinguishing system in case fire emergency
- Automatic de-activation of electric systems and activate emergency light

Controlled Access

- Granting access using finger print, eye or face recognition
- The System informs the user if a visitor is at the door; the owner can then grant or deny access



Figure 5a



Figure 5b

SMART homes offers benefits for the disabled and elderly persons in the various ways:

- Monitory systems
- Automated alarm systems directly connected to hospitals in cases of incidents like a fall or collapse
- Alert systems for the notification on prescribed medications



Figure 8a



Figure 8a

Challenges of Smart Home Technologies

- Ethical questions about privacy and data protection on the Smart Home devices
- Cost of implementation
- More vulnerabilities in the overall systems
- Complexities in the multiple functionality of the technology



Figure 6a

Future of Smart Home Technologies

Smart home technology is one of the emerging technologies growing affordably at a competitive rate in the technological world enhancing home security, entertainment, comfort and health.

Conclusion

In conclusion, Smart Home technology makes it possible for the house to take care of you.



Figure 7a

References

- Luxury Home Living. 2010. Frontpage image. Is Your Home Smart Enough For You? Available at: <http://www.sfgalleryhouse.com/2010/10/is-your-home-smart-enough-for-you/> [accessed on 15 May 2011]
- Cine Consulte. Figure 1a. Home Automation Solutions. Available at: <http://www.home-automation-montreal.ca/> [accessed on 15/5/11]
- Control4.. Figure 2a. What is home control? Available at: <http://www.control4.com/residential/> [accessed on 15 May 2011]
- Peak. Figure 3a. Electric Garage Doors, Remote Control Garage Doors, Automatic Garage Doors.[online]. Available at: <http://peakgaragedoors.co.uk/electric-garage-doors.html> [accessed on 15 May 2011]
- India Shop online. Figure 3b. Logitech Comfort Lapdesk for Notebooks. Available at: <http://www.hydshop.in/products/Logitech-Comfort-Lapdesk-for-Notebooks.html> [accessed on 15 May 2011]
- Lewis Business Technology. Figure 4b. Monitor Audio – Speakers for High-Performance Smart Homes. Available at: <http://www.lewisbt.co.uk/smart-technology/monitor-audio> [accessed on 15 May 2011]
- Home health tech store. Figure 8a, 8b. Independent Living. Available at: <http://www.homehealthtechstore.com/> [accessed on 15/5/2011]
- Sum Security. Figure 5a. Available at: <http://www.sum-security.com/> [accessed on 15 May 2011]
- ION AudioVisual. Figure 4b, 6a. At ION we specialize in the installation of AudioVisual equipment. Available at: <http://www.ionavsecurity.co.uk/bespoke-audiovideo-systems/> [accessed on 15 May 2011]
- Rainbow group. Figure 7a. Available at: <http://www.rainbowgrp.co.uk/Insurance/Protection-Cover.aspx> [accessed on 15/05/11]

Smart Cities

The background of the slide features a blue gradient with a network of white lines and dots. On the left, a hand is shown interacting with a digital interface that displays several interlocking gears. The gears are white with blue outlines and are set against a blue background. The hand is positioned as if it is about to touch or is touching one of the gears. The overall theme is technology and smart infrastructure.

By
Amardeep Das
Assistant Professor

Smart Cities

- A smart city is an urban area that uses different types of electronic methods and sensors to collect data.





Smart Cities





Characteristics of Smart Cities

- Infrastructure Development. A **smart city** prioritises the optimal development of infrastructure in order to enhance economy, and social, cultural and urban development
- Strategies to Create a Competitive Environment
- Inclusive and **Sustainable Cities**.

Smart city Frameworks

- A Smart City Framework is a simple decision methodology that enables both the public and private sectors to plan and implement Smart City initiatives more





Layer 1: City Objectives—Improving Social, Environmental, and Economic Pillars

- At a high level, most city discussions center on policy questions such as, “If we spend money on transportation, how will it improve the city?” Or, “How do we attract jobs and increase economic growth?”



Layer 2: City Indicators—Matching Indicators to City Objectives

- city objectives are high level and somewhat ephemeral, it is important to link them to existing, published “city indicators,” which measure and benchmark cities using defined and specific methodologies



Layer 3: City Components—Detailing City Assets

- Most Smart City initiatives manifest themselves in a city's physical location (e.g., train station) and industry sector (e.g., transportation).
- This layer of the framework details the physical components of a city—utilities, transportation, real estate, and services—which are then linked to city objectives, indicators, and content



Layer 4: City Content—Mapping Objectives to Best Practices and Policies

- This layer encompasses the “how”—how Smart City solutions are implemented.
- It links directly to Layer 3 and then to Layer 1, as it provides information and enables the identification of information that is relevant to Layer 1 (city objectives).



Smart City Framework: Key Outcomes and Benefits

A Smart City Framework not only provides a detailed view of how cities function, but also enables three major outcomes:

1. Taxonomy/typology that enables cities to benchmark relevant content based on the hierarchy of physical city components
2. Stakeholder roles that define who does what. Unfortunately, this part is missing from many city discussions; its omission creates a lack of understanding in how to implement Smart City solutions.
3. Catalog system of city content that is easily accessible



Smart City Framework: Key Outcomes and Benefits

These outcomes will enable cities to:

- Customize a Smart City blueprint
- Identify where and how to implement ICT solutions in cities
- Develop government policy guidelines for enabling private-sector participation in city projects
- Conduct a city gap analysis that enables cities to benchmark themselves, consistently and accurately
- Create a structured case study template for collating multiple business models for similar Smart City initiatives



Challenges in Smart cities

- Challenge #1: Infrastructure. ...
- Challenge #2: Security and Hackers. ...
- Challenge #3: Privacy Concerns. ...
- Challenge #4: Educating & Engaging the Community.
- Challenge #5: Being Socially Inclusive.



Challenge #1: Infrastructure

- Smart Cities utilize sensor technology to gather and analyze information in an effort to improve the quality of life for residents. Sensors collect data on everything from rush hour stats to crime rates to overall air quality.
- Complicated and costly infrastructure is involved in installing and maintaining these sensors. How will they be powered? Will it involve hard-wiring, solar energy, or battery operation? Or, in case of power failure, perhaps a combination of all three?



Challenge #2: Security and Hackers

- IoT and sensor technology use expands, so does the threat level to security.
- This begs the question...is technology really considered “smart” if hackers can break into it and shut down an entire city?



Challenge #3: Privacy Concerns

- Cameras installed on every street corner may help deter crime, but they can also install fear and paranoia in law-abiding citizens.
- Another valid concern is the amount of data being collected from all the smart sensors residents come into contact with each day.



Challenge #4: Educating & Engaging the Community

- A Smart City to truly exist and thrive, it needs “smart” citizens who are engaged and actively taking advantage of new technologies.
- With any new city-wide tech project, part of the implementation process must involve educating the community on its benefits.
- This can be done through a series of in-person town hall-style meetings and email campaigns with voter registration, as well as an online education platform that keeps citizens engaged and up-to-date.



Challenge #5: Being Socially Inclusive

- Smart transit programs that give riders real-time updates are a great idea for a bustling city.
- But what if half the population of that city can't afford to take mass transit or Uber? What about a growing elderly population that doesn't use mobile devices or apps? How will smart technology reach and benefit these groups of people?



Data Fusion

- Data fusion is the process of integrating multiple data sources to produce more consistent, accurate, and useful information than that provided by any individual data source.
- Data fusion processes are often categorized as low, intermediate, or high, depending on the processing stage at which fusion takes place.

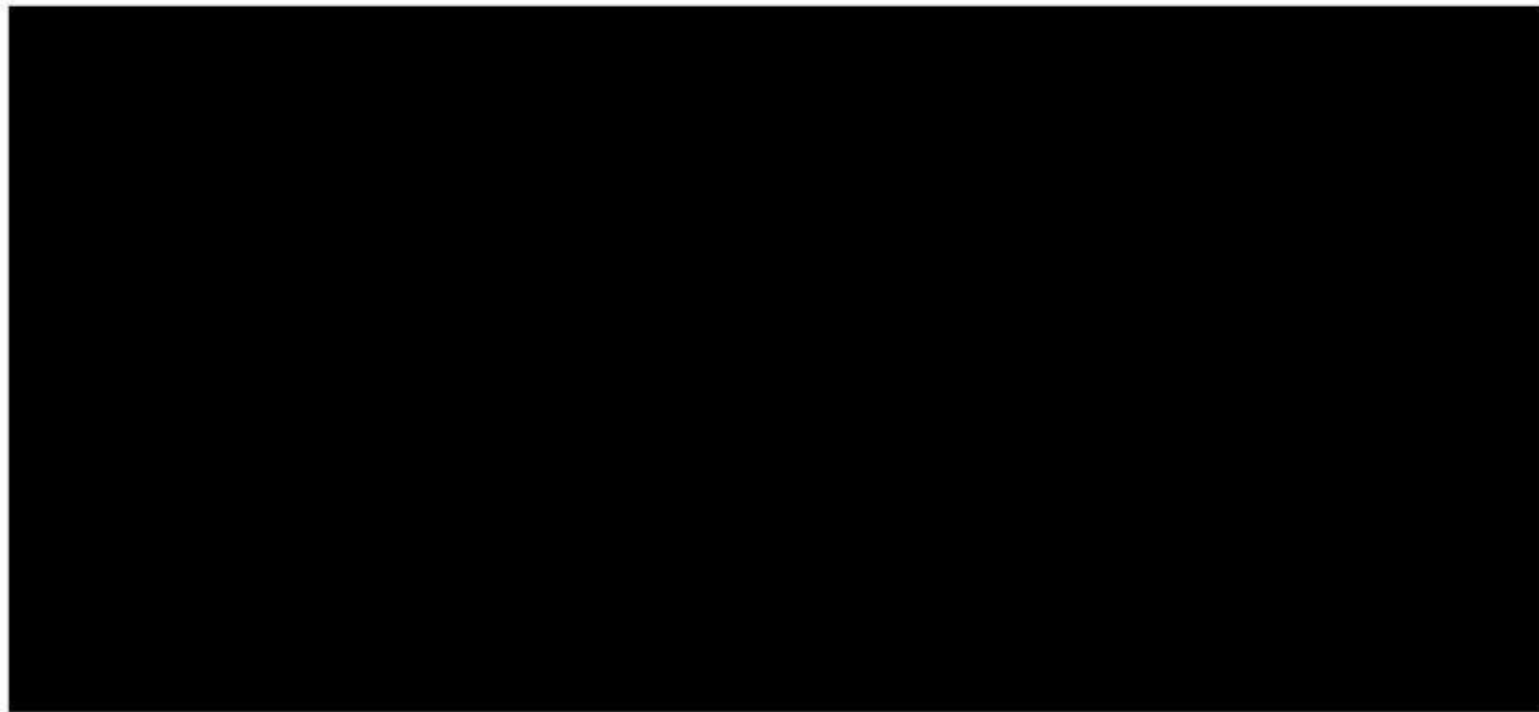


Smart Parking

- Smart Parking systems typically obtain information about available parking spaces in a particular geo- graphic area and process is real-time to place vehicles at available positions.
- It involves using low-cost sensors, real-time data collection, and mobile phone-enabled automated payment systems that allow people to reserve parking in advance or very accurately predict where they will likely find a spot.
- When deployed as a system, smart parking thus reduces car emissions in urban centers by reducing the need for people to needlessly circle city blocks searching for parking .
- It also permits cities to carefully manage their parking supply Smart parking helps one of the biggest problems on driving in urban areas; finding empty parking spaces and controlling illegal parking.
- This implies M2M technologies aims rightness/safety as well as convenience.



Smart Parking





Energy Management in Smart cities

- It is a system of computer-aided tools used by operators of the electrical grid to monitor and optimize the performance of the system.
- Analyze the effect of external factors on how energy is used

Industrial IoT

The image features a blue background with a network of white nodes and lines. In the center, there are several interlocking white gears. A hand from the top left is reaching towards the gears, and another hand from the bottom is reaching up towards them. The overall theme is industrial IoT.

By
Amardeep Das
Assistant Professor



Industrial IoT





Industrial IoT

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



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.



IIoT requirements

- Cloud Computing.
- Access (anywhere, anytime)
- Security.
- Big Data Analytics.
- UX (User Experience)
- Assets Management.
- Smart Machines.



Design considerations

Four Considerations for Manufacturing with the IIoT

- Analytics Architecture is Opening Doors to New Services. ...
- IIoT Platforms are Beginning to Converge and Overtake Traditional Software Applications. ...
- Production Efficiency and Mode of Production. ...
- Security Still Foremost on Manufacturer's Minds.

Applications of Industrial Internet of Things

- Industrial Automation. Image source: sptechlab.com. ...
- Smart Robotics. Image source: ABB. ...
- Predictive Maintenance. ...
- Integration of Smart Tools / Wearables. ...
- Smart Logistics Management. ...
- Software integration for product optimization. ...
- Smart Package Management. ...
- Enhanced Quality and Security.



Benefits of IIoT

- Increase efficiency. The biggest benefit of IIoT is that it gives manufacturers the ability to automate, and therefore optimize their operating efficiency. ...
- Reduce Errors. ...
- Predictive Maintenance. ...
- Improve Safety. ...
- Reduce Costs.



Challenges of IIoT

- High-Investment Cost. One of the more obvious **industrial IoT challenges** is the high cost of adoption. ...
- Secure Data Storage & Management. IoT devices generate a TON of data. ...
- Connectivity Outages