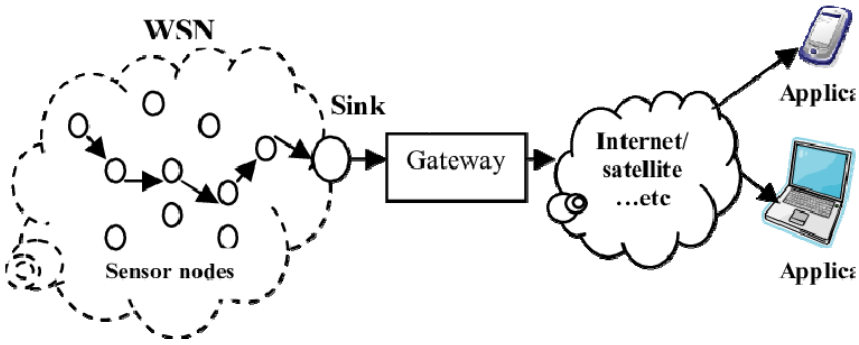
Wireless Sensor Network

- WSN: Things (sensor nodes) connected without a wire to gather some data.
- Wireless sensor network (WSN) refers to a group of dedicated sensors for monitoring and recording the physical conditions of the environment and organizing the collected data at a central location.
- Nodes are not directly connected to the Internet. Nodes route traffic to reach the sink node.
- WSN is sometime referred to as a subset of IoT.



The Internet of things (IoT)

IOT: WSN + Any physical object (Thing) + IP address + Internet + App + Cloud computing+ etc…

- Sensors send their data directly to the Internet because they have Internet Connection. The Internet of things (IoT) is the network of physical devices, vehicles, home appliances, and other items embedded with electronics, software, sensors, actuators, and connectivity which enable these objects to connect and exchange data.
- Each thing is uniquely identifiable through its embedded computing system but is able to inter-operate within the existing Internet infrastructure.
- All sensor data further processed and analyzed in the data analyzing area.

IoT Definitions

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

The Internet of Things (IoT) describes the network of physical objects—"things"—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet.

The Internet of things (IoT) describes physical objects (or groups of such objects) with sensors, processing ability, software, and other technologies that connect and exchange data with other devices and systems over the Internet or other communications networks.

The Internet of Things (IoT) is a network of physical objects that are fitted with sensors, software and other technologies. Connected to the Internet, these 'things' are able to exchange real time data with other connected devices and systems over networks. These connected devices combine with automated systems to gather IoT data that can be analyzed to assist with tasks or learn how to improve a process.

IoT Definitions

- While the Carnegie mellon University's vending machine was installed in 1982, this can't really be called the start of the IoT as a whole.
- The notion of the IoT was created in 1991 and further developed through the 1990s with Reza Raji describing the concept at the IEEE Spectrum in 1994.
- The actual term 'the Internet of Things' was created by Kevin Ashton in 1999, although the time when objects were connected directly to the Internet really began between 2008 and 2009.
- With more than 7 billion connected IoT devices today, experts are expecting this number to grow to 10 billion by 2020 and 22 billion by 2025. Oracle has a network of device partners.

IoT Characteristics

Connectivity –
Connectivity is an important requirement of the IoT infrastructure. Things of IoT should be connected to the IoT infrastructure. Anyone, anywhere, anytime can connect, this should be guaranteed at all times.

Intelligence and Identity
The extraction of knowledge from the generated data is very important.

Scalability
With the growing ubiquity of the Internet of Things, it has become increasingly important to consider scalability in your system design. Scalability is often defined as the ability of a system to grow without affecting its performance. This can be achieved by adding more hardware resources or by adding additional software layers to an existing system. The data generated as an outcome is enormous, and it should be handled appropriately.

Dynamic and Self-Adapting (Complexity)
IoT devices should dynamically adapt themselves to the changing contexts and scenarios.

IoT Characteristics

Architecture – Common Ecosystem
In the internet of things, there are many manufacturers and products that are using the architecture to support their own devices. With the increase in the number of devices, the importance of the architecture is heterogeneous (ability to support diverse technologies, protocols, and devices). The architecture is mostly responsible for making sure that the devices work together and communicate with each other. It also is a key component in ensuring that the devices do not interfere with each other.

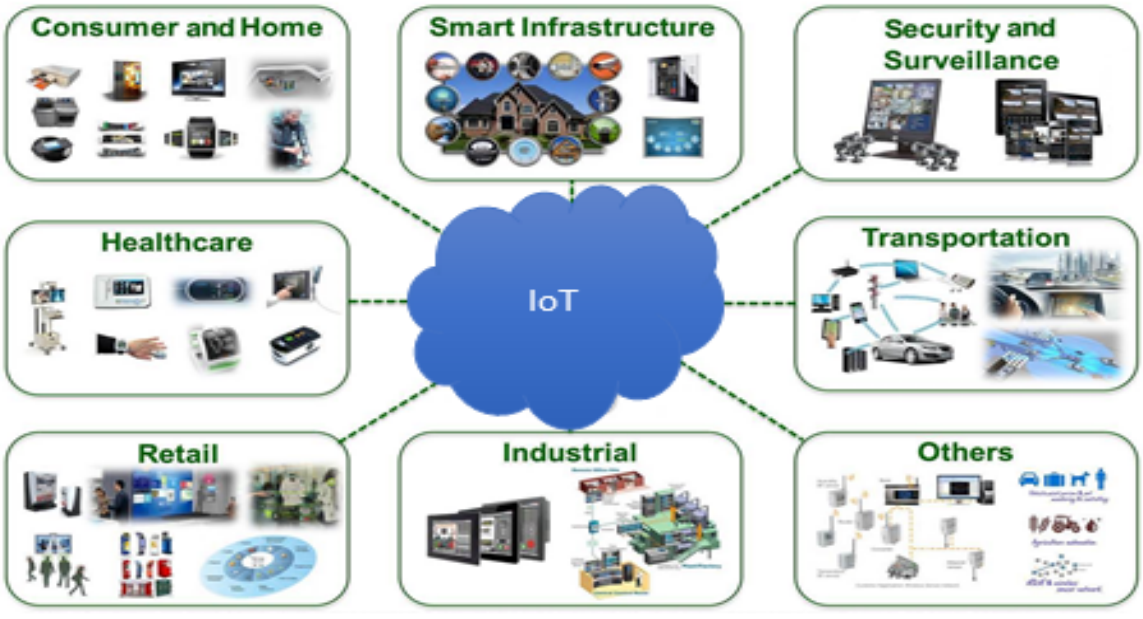
Self Configuring – This is one of the most important characteristics of IoT. IoT devices are able to upgrade their software in accordance with requirements with a minimum of user participation. Additionally, they can set up the network, allowing for the addition of new devices to an already-existing network.

Unique Identity of Things
Identity is the unique characteristic of a person, group, place, or thing. Every identity has a name and an identification number. Identity is a concept that is found in many aspects of IoT. Device Identity is the one thing that makes an IoT device unique and identifiable. Identity can be used to distinguish between different devices, give them a name, and allow them to be controlled.

IoT Applications

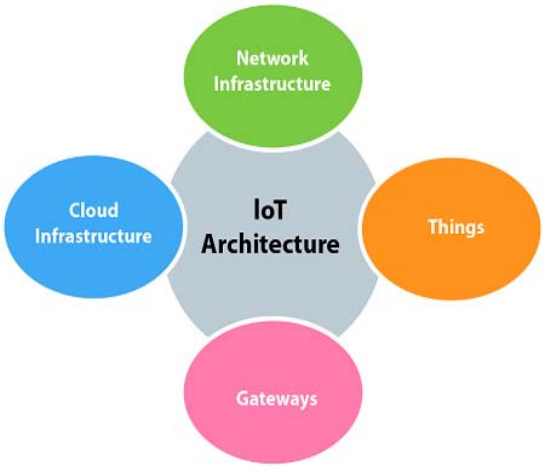
Smart Home	Smart Lightening
	Smart Appliances
	Intrusion Detection
	Smoke/Gas Detection
Smart Cities	Smart Parking
	Smart Road
	Structural Health Monitoring
	Emergency Response
Smart Environment	Weather Monitoring
	Air Pollution Monitoring
	Noise Pollution Monitoring
	Forest Fire Detection
Smart Energy	Smart Grids
	Renewable Energy Systems
	Prognostics

Smart Retail	Inventory Management
	Smart Payments
	Smart Vending Machine
Smart Logistics	Route Generation and Scheduling
	Fleet Tracking
	Ship Monitoring
	Remote Vehicle Diagnostics
Smart Agriculture	Smart Irrigation
	Green House Control
	Pest Spraying Control
	Plants Disease Detection
Smart Industry	Machine Diagnosis & Prognosis
	Indoor Air Quality Management
Smart Health & Lifestyle	Health & Fitness Monitoring
	Wearable Sensors

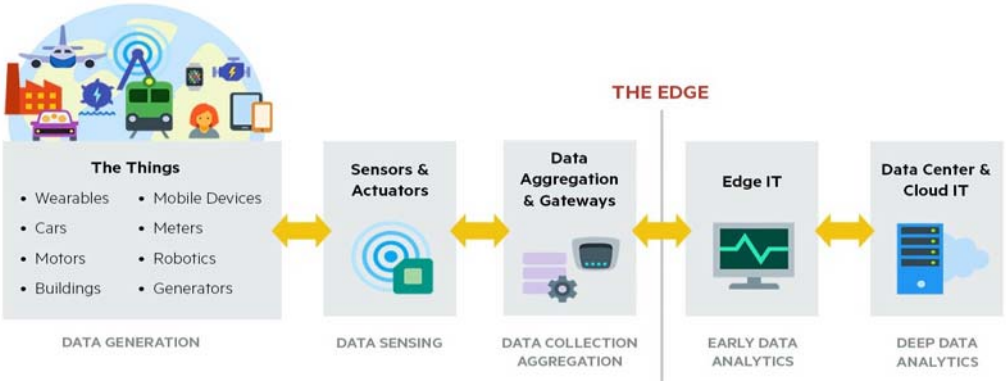


IoT Architecture

An IoT architecture is a mix of hardware and software components that interact together to make up a smart cyber-digital system. Interoperating with one another, these components make up a base for an IoT solution to be built upon. Before we dive into the details, let's get things straight: there's no one-size-fits-all approach to designing an IoT architecture. Still, the basic layout stays largely the same no matter the solution.



IoT Architecture



- **Devices:** This stage is about the actual devices in the IoT solutions. These devices could be sensors or actuators in the Perception layer. Those devices will generate data (in the case of sensors) or act on their environment (in the case of actuators). The data produced is converted in a digital form and transmitted to the internet gateway stage. Unless a critical decision must be made, the data is typically sent in a raw state to the next stage due to the limited resources of the devices themselves.

IoT Architecture

- **Internet gateways:** The internet gateway stage will receive the raw data from the devices and pre-process it before sending it to the cloud. This internet gateway could be physically attached to the device or a stand-alone device that could communicate with sensors over low power networks and relay the data to the internet.
- **Edge or fog computing:** In order to process data as quickly as possible, you might want to send your data to the edge of the cloud. This will let you analyze the data quickly and identify if something requires immediate attention. This layer typically would only be concerned with recent data that is required for time-critical operations. Some pre-processing might be done at this stage, too, to limit the data that is ultimately transferred to the cloud.
- **Cloud or data center:** In this final stage, the data is stored for later processing. The application and business layers live in this stage, where dashboards or management software can be fed through the data stored in the cloud. Deep analysis or resource-intensive operations such as machine learning training will happen at this stage.

IoT Designing

An IoT application structure have basically four design paradigms.

1. **Physical Design**
   - IoT Device
   - IoT Layer Architectures (Protocols)
2. **Logical Design**
   - IoT Functional Blocks
   - IoT Communication Models
   - IoT Communication APIs
3. **IoT Enabling Technologies**
   - Wireless Sensor Networks
   - Cloud Computing
   - Big Data Analytics
   - Communication Protocols
   - Embedded System
   - Artificial Intelligence & Machine Learning

IoT Designing

4. IoT Levels

- IoT Level 1
- IoT Level 2
- IoT Level 3
- IoT Level 4
- IoT Level 5
- IoT Level 6

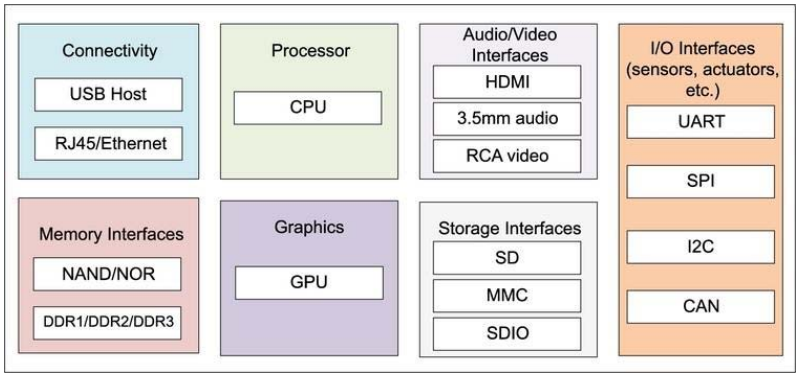
IoT Physical Design

IoT Things:

The things in IoT refers to IoT devices which have unique identities and perform remote sensing, actuating and monitoring capabilities. IoT devices can exchange data with other connected devices applications. It collects data from other devices and process data either locally or remotely.

An IoT device may consist of several interfaces for communication to other devices both wired and wireless. These includes

(i) I/O interfaces for sensors,
(ii) Interfaces for internet connectivity
(iii) Memory and storage interfaces
(iv) Audio/Video interfaces.



IoT Physical Design

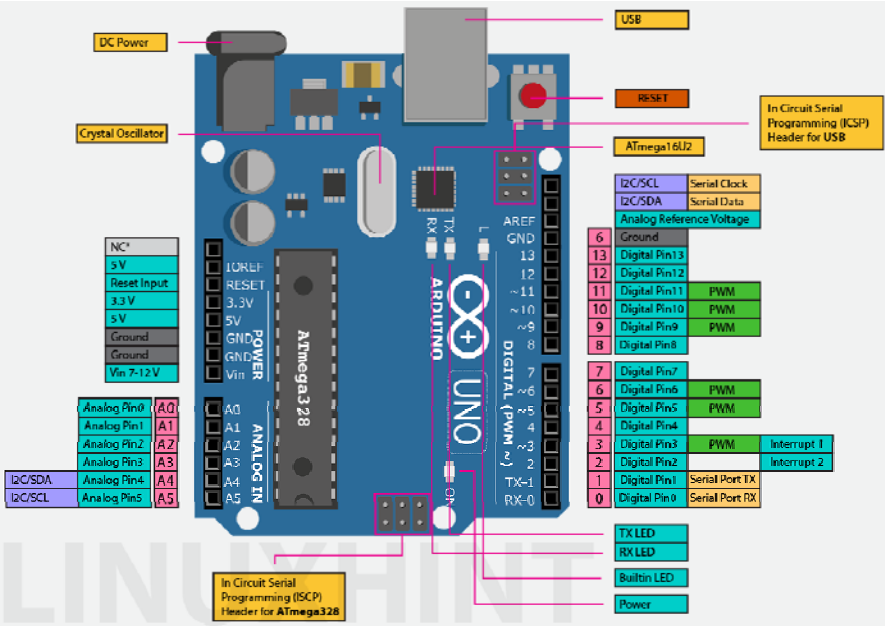
UART/RS232 - UART is usually an individual (or part of an) integrated circuit (IC) used for **serial communications over a computer or peripheral device serial port**. One or more UART peripherals are commonly integrated in microcontroller chips. Specialized UARTs are used for automobiles, smart cards and SIMs.

SPI - Serial peripheral interface (SPI) is one of the most widely used interfaces between microcontroller and peripheral ICs such as sensors, ADCs, DACs, shift registers, SRAM, and others.

I²C—Shorthand for inter-integrated circuit or I squared C, this is a low-speed serial bus that is used for sending data from one chip to another on the same PC board or over short cables between two pieces of equipment. I2C combines the best features of SPI and UARTs. With I2C, you can connect multiple slaves to a single master (like SPI) and you can have multiple masters controlling single, or multiple slaves. This is really useful when you want to have more than one microcontroller logging data to a single memory card or displaying text to a single LCD.

CAN Controller Area Network Interface Bus (CAN)—This is controller area network serial bus in an interface used to link micros together in small networks. It is used in automotive and industrial applications. The usual cable is twisted pair and a ground. Speeds range from 10 kbps to 1 Mbps with 20 kbps being typical. The data is sent in specifically formatted frames as determined by a standard protocol. It uses start and stop bits like the RS-232 and some similar control codes defined in ASCII. It also includes error detection and correction capability, so it is very reliable.

IoT Physical Design

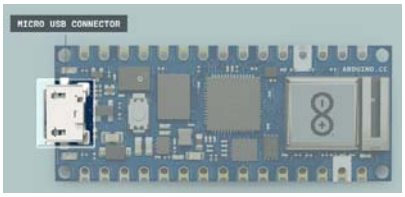


IoT Physical Design

IoT Physical Design

Powering Alternatives

Arduino boards have **five** options in which they can be powered:



1. Powering via USB connector
2. Powering via the onboard barrel jack connector (if available on the board)
3. Powering via the onboard battery connector (if available on the board)
4. Powering via the VIN (Voltage In) pin
5. Powering via the 3V3/5V pin*

*Powering your board via the 3V3/5V pins is not recommended, as it can damage your board's voltage regulator.

Powering via USB connector

- The USB connector provides a regulated 5V line to power the board's electronics. However, **5V from the USB connector can also power external components through the 5V pin** that can be found in Arduino boards.
- Arduino boards that run at 5V use the USB-regulated 5V line directly, boards that run at 3V3 regulate the 5V line from the USB connector to 3V3 using their onboard voltage regulator. Output current rating from the 5V pin will vary, depending on the 5V power source.
- Current from USB ports of computers is usually limited to 500mA.

IoT Physical Design

Powering via the onboard barrel jack connector (if available on the board)

The voltage line from the barrel jack connector is regulated in Arduino boards using their onboard voltage regulator; usually, it is first regulated to 5V and then regulated again to 3V3 in most Arduino boards. The **recommended voltage and current ratings for external regulated DC power supplies** connected to the barrel jack connector are summarized in the table below:

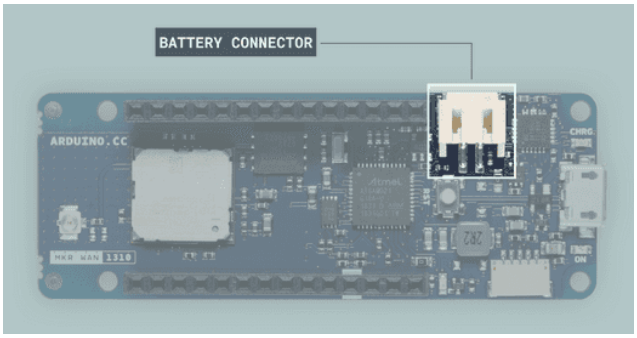
Board	External Power Supply Voltage (V)	External Power Supply Current (A)
Arduino UNO Rev3	7-12	1
Arduino UNO WiFi Rev2	7-12	1.5
Arduino Leonardo	7-12	1
Arduino Mega 2560 Rev3	7-12	1
Arduino Due	7-12	1.5
Arduino Zero	5-18	1

IoT Physical Design

Powering via the onboard battery connector (if available on the board)

Some Arduino boards have an **onboard battery connector** to connect a battery to the board and use it as its primary or secondary power supply. The Arduino boards with an onboard battery connector are the following:

- Arduino Portenta H7
- Arduino Nicla Sense ME
- Arduino Nicla Vision
- Arduino MKR NB 1500
- Arduino MKR Vidor 4000
- Arduino MKR WiFi 1010
- Arduino MKR ZERO
- Arduino MKR WAN 1310
- Arduino MKR GSM 1400



Battery connector of the Arduino MKR WAN 1310 board

Pro family boards use a 3-pin, 1.2mm SMD ACH battery connector; MKR family boards use a 2-pin, 2mm SMD PH battery connector.

IoT Physical Design

Powering via the VIN (Voltage In) pin

- The VIN pin in Arduino boards is a power pin with a dual function.
- This pin can work as a **voltage input for regulated external power supplies** that do not use a barrel jack connector.
- This pin can also work as a **voltage output when an external power supply is connected to the barrel jack connector** present in some Arduino boards.
- An important consideration is that the VIN pin is connected directly to the input pin of the onboard voltage regulator on Arduino boards.
- Since the VIN pin is directly connected to the voltage regulator, the **VIN pin does not have reverse polarity protection**.
- Use the VIN pin carefully to avoid damaging your Arduino board since it does not have reverse polarity protection.
- The **minimum and maximum voltages** that can be applied to the VIN pin are determined by the onboard voltage regulator on Arduino boards, varying from board to board. Those voltages are summarized in the table below:

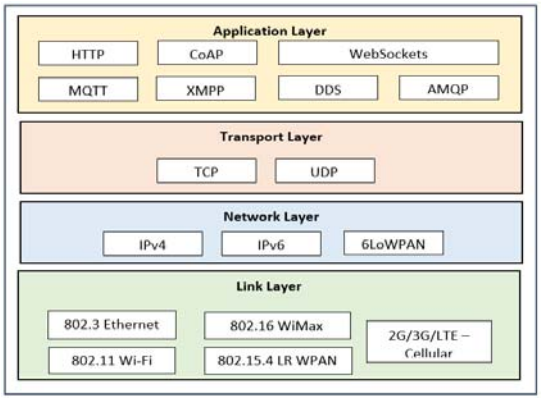
Board	VIN Voltage (V)
UNO Mini	5-18
UNO Rev3	7-12
UNO WiFi Rev2	7-12
UNO Rev3 SMD	7-12
Leonardo	7-12
Mega 2560 Rev3	7-12
Due	7-12
Micro	7-12
Zero	5-18
Portenta H7	5
Nicla Sense ME	5
Nano RP2040 Connect	5-18
MKR NB 1500	5-7
MKR GSM 1400	5-7
MKR Vidor 4000	5-7
MKR WiFi 1010	5-7
MKR Zero	5-5.5
MKR1000 WIFI	5-5.5
MKR WAN 1300	5-5.5
MKR WAN 1310	5-7
Nano	7-12
Nano Every	7-18
Nano 33 IoT	5-18
Nano 33 BLE	5-18
Nano 33 BLE Sense	5-18

IoT Physical Design

IoT Protocols

Link Layer

Protocols determine how data is physically sent over the network's physical layer or medium. Local network connect to which host is attached. Hosts on the same link exchange data packets over the link layer using link layer protocols. Link layer determines how packets are coded and signalled by the h/w device over the medium to which the host is attached.



IoT Physical Design

Protocols:

- **802.3-Ethernet:** IEEE802.3 is collection of wired Ethernet standards for the link layer. E.g.:
  - 802.3 uses co-axial cable;
  - 802.3i uses copper twisted pair connection;
  - 802.3j uses fiber optic connection;
  - 802.3ae uses Ethernet over fiber.
- **802.11-WiFi:** IEEE802.11 is a collection of wireless LAN(WLAN) communication standards including extensive description of link layer. E. g.
  - 802.11a operates in 5GHz band,
  - 802.11b and 802.11g operates in 2.4GHz band,
  - 802.11n operates in 2.4/5GHz band,
  - 802.11ac operates in 5GHz band,
  - 802.11ad operates in 60Ghz band.
- **802.16 - WiMAX:** IEEE802.16 is a collection of wireless broadband standards including exclusive description of link layer. WiMAX provide data rates from 1.5 Mb/s to 1Gb/s.

IoT Physical Design

- **802.15.4-LR-WPAN:** IEEE802.15.4 is a collection of standards for low rate wireless personal area network (LR-WPAN). Basis for high level communication protocols such as ZigBee. Provides data rate from 40kb/s to 250kb/s. These communications provide low cost and low speed communication for power constrained devices.
- **2G/3G/4G-Mobile Communication:** These are the different generations of mobile communications standards:
  - 2G including GSM and CDMA,
  - 3G including UMTS and CDMA3000,
  - 4G including LTE,
  - Data rates from 9.6kb/s(2G) up to100Mb/s(4G).

Aspects	ZigBee	WiFi	Bluetooth
standard	802.15.4	802.11a,b,g	802.15.1
application	automation, control	Web, e-mail, video	replacement of cableing
data rate	50 - 60 kbyte	> 1 Mbyte	> 250 kbyte
battery lifetime	> 1000	1 -5	1 -7
network size	65535	32	7
bandwith (kb/s)	20 - 250	11000	720
transmission distance	100+ m	100 m	10 m
advantage	reliability, performance, cost	data rate, flexibility	cost, comfortable

IoT Physical Design

LoRa

- LoRa, which is an abbreviation of “long range”, from RF technology for an LPWAN.
- LoRa is one of the most prominent wireless technologies in the low-power wide-area network (LPWAN) family.
- LoRa is a patented energy-efficient wireless communication protocol that achieves very low-power and very long-range transmissions, of over 10 km in line-of-sight, trading-off data rate, and time-on-air.
- LoRa’s duty cycle is limited by regional regulations because it operates using unlicensed sub-GHz radio bands, mostly on the 433, 868, and 915 MHz frequency bands.
- The data transmission rate supported by LoRa varies from 300 bps to 50 kbps, depending on spreading factor (SF) and channel bandwidth settings.
- Taking into account this and the low-transmission bandwidth, LoRa is naturally most suitable for applications where transmissions are sparse in time and payloads are relatively small.

IoT Physical Design

Network/Internet Layer:

Responsible for sending IP datagrams from source network to destination network. Performs the host addressing and packet routing. Datagrams contains source and destination address.

Protocols:

- **IPv4:** Internet Protocol version4 is used to identify the devices on a n/w using a hierarchical addressing scheme. 32-bit address. Allows total of 2^{32}addresses. These addresses got exhausted in the year of 2011. IPv4 has been succeeded by IPv6. IP protocols establish connection over packet networks, but do not guarantee delivery of packets.

- **IPv6:**It is newest version of IPv4. Internet Protocol version6 uses 128-bit address scheme and allows 2^{128} addresses.

- **6LOWPAN**:(IPv6 over Low Power Wireless Personal Area Network) operates in 2.4 GHz frequency range and data transfer 250 kb/s. It brings IP protocol for low power devices having limited processing capability.

Transport Layer:

Provides end-to-end message transfer capability independent of the underlying n/w. Set up on connection with ACK as in TCP and without ACK as in UDP. Provides functions such as error control, segmentation, flow control and congestion control.

Protocols:

- **TCP:** Transmission Control Protocol used by web browsers (along with HTTP and HTTPS), email(along with SMTP, FTP). Connection oriented and stateless protocol. IP Protocol deals with sending packets, TCP ensures reliable transmission of protocols in order. TCP also provides the error detection capability so that duplicate packets can be discarded and loss packets are retransmitted. Avoids n/w congestion and congestion collapse. It also ensures flow control to adjust the mismatch of transmission and receiving speed of sender and receiver.

- **UDP:** User Datagram Protocol is connectionless protocol. Useful in time sensitive applications, very small data units to exchange. Transaction oriented and stateless protocol. Does not provide guaranteed delivery, ordering of messages and duplicate elimination.

Application Layer:

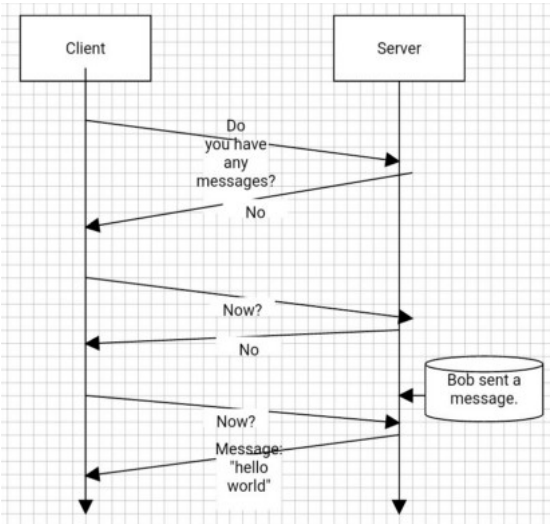
Defines how the applications interface with lower layer protocols to send data over the n/w. Enables process-to-process communication using ports.

HTTP

- Hyper Text Transfer Protocol that forms foundation of WWW.
- Follow request response model Stateless protocol.
- It was designed for communication between web browsers and web servers, but it can also be used for other purposes.
- HTTP follows a classical client-server model, with a client opening a connection to make a request, then waiting until it receives a response.
- HTTP is a stateless protocol, meaning that the server does not keep any data (state) between two requests.

If you are a developer, you probably know what HTTP **(or HTTPS)** is. It is seen every day, in your browser. A request is needed to respond; **you to constantly ask the server if there are new messages in order to receive them.**

You should also know that HTTP allows you to have different types of requests such as post, get or put, each with a different purpose.



Application Layer:

Web Socket:

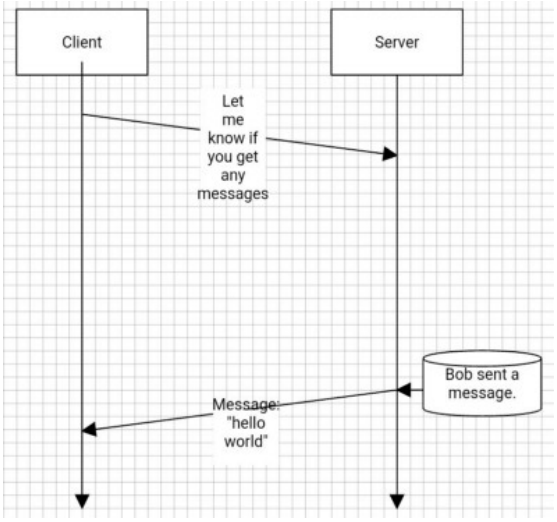
It allows full duplex communication over a single socket connection. This protocol defines a full duplex communication from the ground up. Web sockets take a step forward in bringing desktop rich functionalities to the web browsers.

The main features of web sockets are as follows −

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

Web Sockets on the other hand don't need you to send a request in order to respond. They allow bidirectional data flow so you just have to listen for any data.

You can just listen to the server and it will send you a message when it's available.



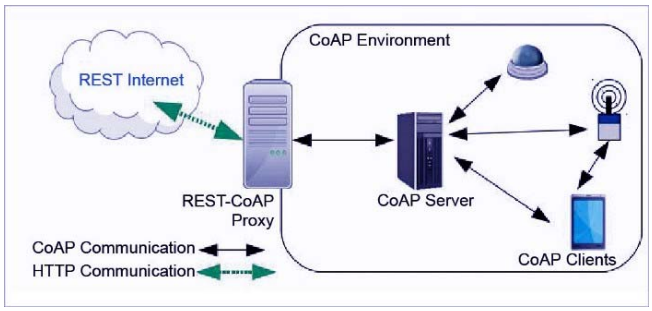
Web Socket Applications

- Real-time applications
- Chat apps
- IoT (internet of things)
- Online multiplayer games

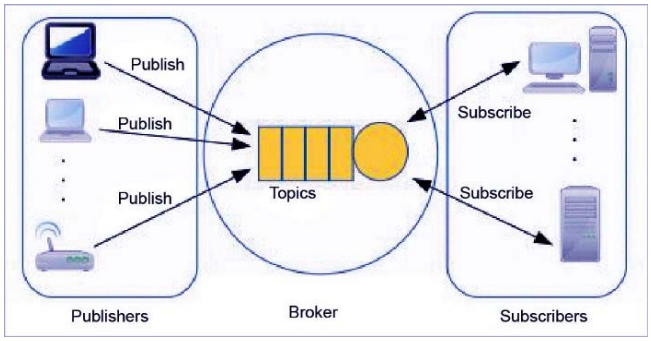
Application Layer:

CoAP

- Constrained Application Protocol for machine-to-machine (M2M) applications with constrained devices, constrained environment and constrained n/w.
- Uses client server architecture. Like HTTP, CoAP is a document transfer protocol.
- Unlike HTTP, CoAP is designed for the needs of constrained devices.
- CoAP packets are much smaller than HTTP TCP flows.
- CoAP runs over UDP, not TCP.
- CoAP allows UDP broadcast and multicast to be used for addressing.
- CoAP follows a client/server model. Clients make requests to servers, servers send back responses.
- Clients may GET, PUT, POST and DELETE resources.



- CoAP is designed to interoperate with HTTP and the RESTful web at large through simple proxies. Because CoAP is datagram based, it may be used on top of SMS and other packet based communications protocols.



Application Layer:

MQTT

- Message Queue Telemetry Transport is light weight messaging protocol based on publish-subscribe model.
- Uses client server architecture. Well suited for constrained environment.
- MQTT is a publish/subscribe messaging protocol designed for lightweight M2M communications.
- It was originally developed by IBM and is now an open standard.
- MQTT has a client/server model, where every sensor is a client and connects to a server, known as a broker, over TCP. MQTT is message oriented.

Application Layer:

XMPP

- Extensible Message and Presence Protocol for real time communication and streaming XML data between network entities. Support client-server and server-server communication.
- The XMPP protocols are free, open, public, and easily understandable; in addition, multiple implementations exist in the form clients, servers, server components, and code libraries.
- XMPP applications beyond IM include network management, content syndication, collaboration tools, file sharing, gaming, remote systems monitoring, web services, lightweight middleware, cloud computing, and much more.
- The Internet Engineering Task Force (IETF) has formalized the core XML streaming protocols as an approved instant messaging and presence technology.

Application Layer:

DDS

- Data Distribution Service is data centric middleware standards for device-to-device or machine-to-machine communication. It is an IoT protocol developed for M2M (Machine to Machine) Communication by OMG (Object Management Group).
- DDS is a networking middleware that simplifies complex network programming. It implements a publish–subscribe pattern for sending and receiving data, events, and commands among the nodes.
- Nodes that produce information (publishers) create "topics" (e.g., temperature, location, pressure) and publish "samples". DDS delivers the samples to subscribers that declare an interest in that topic.
- DDS handles transfer chores: message addressing, data marshalling and de-marshalling (so subscribers can be on different platforms from the publisher), delivery, flow control, retries, etc. Any node can be a publisher, subscriber, or both simultaneously
- It enables data exchange via publish-subscribe methodology.

Application Layer:

DDS

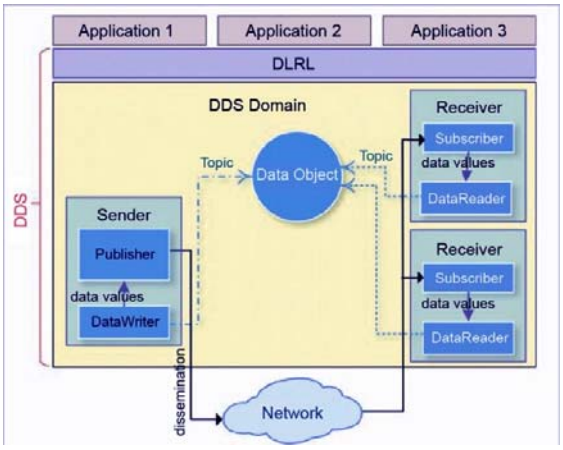
- DDS makes use of broker less architecture unlike MQTT and CoAP protocols.
- It uses multicasting to bring high quality QoS to the applications.
- DDS protocol can be deployed from low footprint devices to cloud.
- The DDS interoperability demonstration used scenarios such as:
  - ❑ Basic connectivity to network using Internet Protocol (IP)
  - ❑ Discovery of publishers and subscribers
  - ❑ Quality of service (QoS) Compatibility between requester and offerer
  - ❑ Delay-tolerant networking
  - ❑ Multiple topics and instances of topics
  - ❑ Exclusive ownerships of topics
  - ❑ Content filtering of topic data including time and geographic

Application Layer:

DDS

- DDS addresses the needs of applications like aerospace and defence, air-traffic control, autonomous vehicles, medical devices, robotics, power generation, simulation and testing, smart grid management, transportation systems, and other applications that require real-time data exchange.

The DDS — IoT protocols have fundamental layers: facts centric submit-subscribe (DCPS) and statistics-local reconstruction layer (DLRL). DCPS plays the task of handing over the facts to subscribers, and the DLRL layer presents an interface to DCPS functionalities, permitting the sharing of distributed data amongst IoT enabled objects.



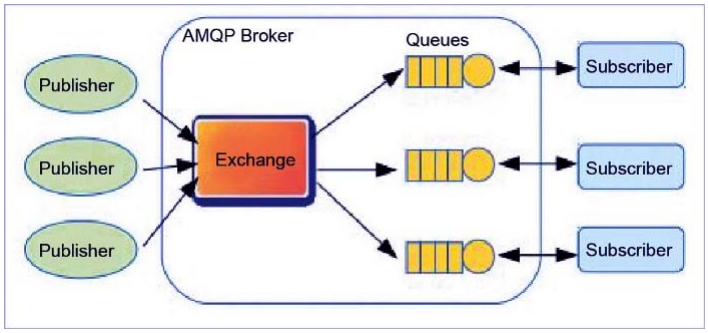
Application Layer:

AMQP:

- Advanced Message Queuing Protocol is open application layer protocol for business messaging.
- Supports both point-to-point and publish-subscribe model.
- AMQP stands for Advanced Messaging Queuing Protocol. It is a standard for cross platform messaging.
- It is a wire protocol and aims to enable functional compatibility between clients (from various platforms) and messaging brokers.
- AMQP will be used to design interoperable, high quality messaging products.
- Enterprise applications are being written in a number of dynamic languages such as Ruby, Perl and Python.
- AMQP will enable platform independence, and fill in the gap to integrate these applications using the messaging model.

The AMQP protocol enables patron programs to talk to the dealer and engage with the AMQP model. This version has the following three additives, which might link into processing chains in the server to create the favoured capability.

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



LOGICAL DESIGN OF IOT

The logical design of an IoT system refers to an abstract representation of entities and processes without going into the low-level specifies of implementation. it uses Functional Blocks, Communication Models, and Communication APIs to implement a system.

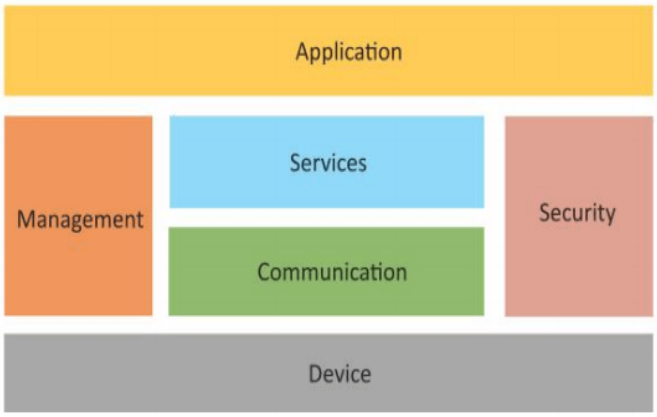
IoT Functional Blocks

IoT Communication Models

IoT Communication APIs

IoT Functional blocks

An IoT system consist number of functional blocks like Devices, services, communication, security, and application that provides the capability for sensing, actuation, identification, communication, and management.



LOGICAL DESIGN OF IOT

These functional blocks consist of devices that provide monitoring control functions, handle communication between host and server, manage the transfer of data, secure the system using authentication and other functions, and interface to control and monitor various terms.

- **Device:** These devices are used to provide sensing and monitoring control functions that collect the data from the outer environment.
- **Communication:** This block handles the communication between the client and cloud-based server and sends/receives the data using protocols.
- **Services:** This functional block provides some services like monitoring and controlling a device and publishing and deleting the data and restore the system.
- **Management:** This functional block provides various functions that are used to manage an IoT system.
- **Security:** This block is used to secure an IoT system using some functions like authorization, data security, authentication, 2 step verification, etc.
- **Application:** It is an interface that provides a control system that use by users to view the status and analyse of system.

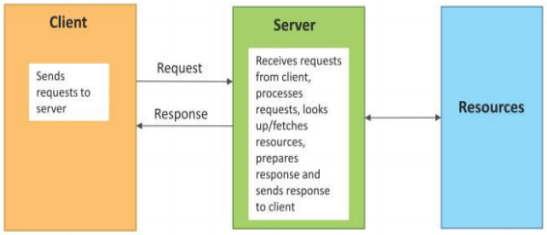
LOGICAL DESIGN OF IOT

IoT Communication Models

There are several different types of models available in an IoT system that used to communicate between the system and server like the request-response model, publish-subscribe model, push-pull model, and exclusive pair model, etc.

Request-Response Communication Model

This model is a communication model in which a client sends the request for data to the server and the server responds according to the request. when a server receives a request it fetches the data, retrieves the resources and prepares the response, and then sends the data back to the client.



In simple terms, we can say that in the request-response model server send the response of equivalent on the request of the client. in this model, HTTP works as a request-response protocol between a client and server.

Example

When we search a query on a browser then the browser submits an HTTP request to the server and then the server returns a response to the browser(client).

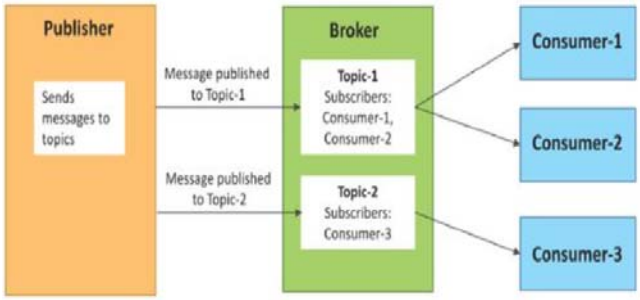
LOGICAL DESIGN OF IOT

Publish-Subscribe Communication Model

In this communication model, we have a broker between publisher and consumer. here publishers are the source of data but they are not aware of consumers. they send the data managed by the brokers and when a consumer subscribes to a topic that managed by the broker and when the broker receives data from the publisher it sends the data to all the subscribed consumers.

Example

On the website many times we subscribed to their newsletters using our email address. these email addresses managed by some third-party services and when a new article published on the website it directly sends to the broker and then the broker send these new data or post to all the subscribers.



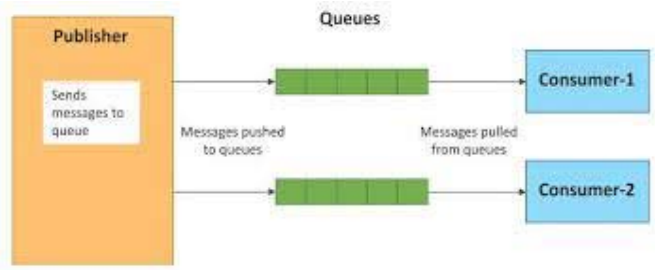
LOGICAL DESIGN OF IOT

Push-Pull Communication Model

It is a communication model in which the data push by the producers in a queue and the consumers pull the data from the queues. here also producers are not aware of the consumers.

Example

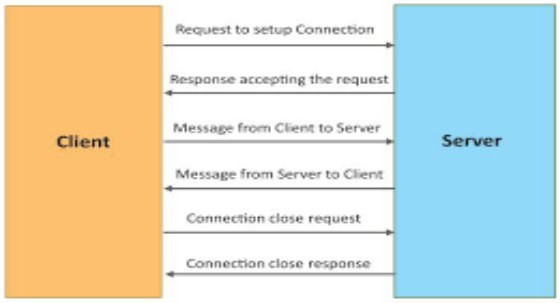
When we visit a website we saw a number of posts that published in a queue and according to our requirements, we click on a post and start reading it.



LOGICAL DESIGN OF IOT

Exclusive Pair Communication Model

It is a bidirectional fully duplex communication model that uses a persistent connection between the client and server. here first set up a connection between the client and the server and remains open until the client sends a close connection request to the server.



LOGICAL DESIGN OF IOT

IoT communication APIs

Generally, we used Two APIs for IoT Communication. These IoT Communication APIs are:

- REST-based Communication APIs
- Web Socket-based Communication API

REST-based Communication APIs

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

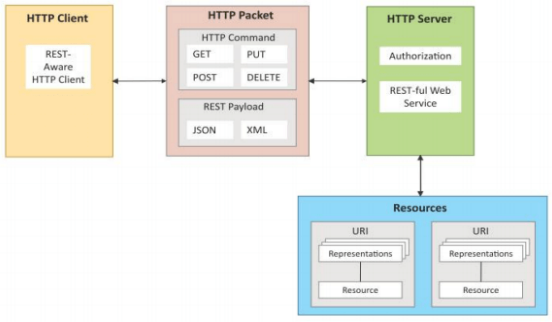
The rest architectural constraint are as follows:

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

LOGICAL DESIGN OF IOT

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

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



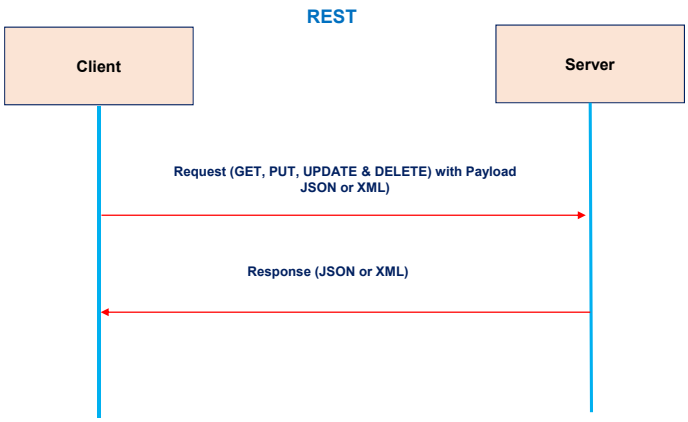
LOGICAL DESIGN OF IOT

- **Layered system** – layered system constraints, constrains the behaviour of components such that each component cannot see beyond the immediate layer with they are interacting. For example, the client cannot tell whether it is connected directly to the end server or two an intermediary along the way. System scalability can be improved by allowing intermediaries to respond to requests instead of the end server, without the client having to do anything different.
- **Uniform interface** – uniform interface constraints requires that the method of communication between client and server must be uniform. Resources are identified in the requests (by URIsin web based systems) and are themselves is separate from the representations of the resources data returned to the client. When a client holds a representation of resources it has all the information required to update or delete the resource you (provided the client has required permissions). Each message includes enough information to describe how to process the message.
- **Code on demand** – Servers can provide executable code or scripts for clients to execute in their context. this constraint is the only one that is optional.

LOGICAL DESIGN OF IOT

Request Response Model used by REST API:

A RESTful web service is a “WebAPI” implemented using HTTP and REST principles. REST is most popular IoT Communication APIs.



LOGICAL DESIGN OF IOT

HTTP Request Methods and Actions

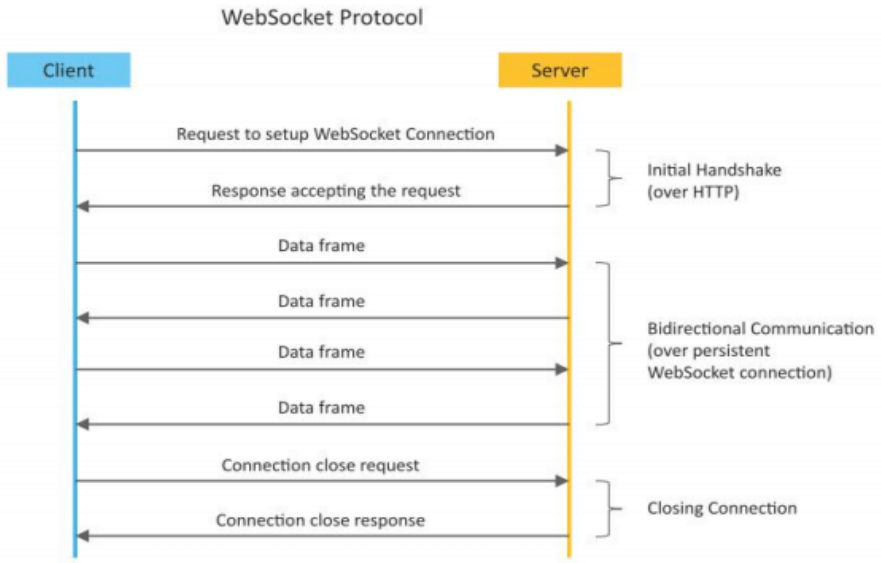
Uniform Resource Identifier (URI)	GET	PUT	PATCH	POST	DELETE
Collection, such as https://api.example.com/resources/	List the URIs and perhaps other details of the collection's members.	Replace the entire collection with another collection.	Not generally used	Create a new entry in the collection. The new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

LOGICAL DESIGN OF IOT

WebSocket based communication API

- This type of API allows bi-directional full-duplex communication between server and client using the exclusive pair communication model.
- This API uses full-duplex communication so it does not require a new connection setup every time when it requests new data.
- WebSocket communication begins with a connection setup request sent by the client to the server. The request (called WebSocket handshake) is sent over HTTP and the server interprets it is an upgrade request. If the server supports WebSocket protocol, the server responds to the WebSocket handshake response.
- WebSocket API begins with a connection setup between the server and client and if the WebSocket is supported by the server then it responds back to the client with the successful response and after setup of a connection server and client can send data to each other in full-duplex mode.
- This type of API reduces the traffic and latency of data and makes sure that each time when we request new data it cannot terminate the request.

LOGICAL DESIGN OF IOT



IOT ENABLING TECHNOLOGIES

There are several technologies which play important role in IoT Applications. The technologies are:

- Wireless Sensor Networks
- Cloud Computing
- Big Data Analytics
- Embedded Systems
- Communication Protocols
- Security Protocols
- Web Services
- Mobile Internet
- Semantic Search Engines

IOT ENABLING TECHNOLOGIES

Cloud Computing

- Cloud computing is the use of off-site systems to help computers store, manage, process, and/or communicate information. These off-site systems are hosted on the cloud (or the internet) instead of on your computer or other local storage. They can encompass anything from email servers to software programs, data storage, or even increasing your computer's processing power.
- The "cloud" is a term that simply means "the internet." Computing involves the infrastructures and systems that allow a computer to run and build, deploy, or interact with information.
- In cloud computing, this means that instead of hosting infrastructure, systems, or applications on your hard drive or an on-site server, you're hosting it on virtual/online servers that connect to your computer through secure networks.

The main types of cloud computing include software as a service, platform as a service, and infrastructure as a service. Serverless computing, also known as function as a service (FaaS), is also a popular method of cloud computing for businesses.

IOT ENABLING TECHNOLOGIES

Cloud Computing

- **SaaS or Software as a Service.** SaaS means instead of installing software on your computer, you access the platform online. Examples would include:
  - Square, which processes payments online
  - Google Apps such as Google Drive or Calendar
  - Slack, which allows collaboration and chat between other users

- **IaaS or Infrastructure as a Service.** IaaS provides infrastructure components such as servers, storage, networking, security, and moreover the cloud. Examples would include:
  - Dropbox, a file storage and sharing system
  - Microsoft Azure, which offers backup and disaster recovery services, hosting, and more
  - Rackspace, which offers data, security, and infrastructure services.

- **PaaS or Platform as a Service.** PaaS provides computing platforms such as operating systems, programming language execution environments, databases, and web servers. Examples would include:
  - Google App Engine and Heroku, which allow developers to develop and serve apps

- **Serverless Computing.** Serverless computing (also called simply "Serverless") is simply using a server on the cloud. This offers more elasticity, easier maintenance, and is often more price effective than hosting servers on-site.

IOT ENABLING TECHNOLOGIES

Big Data Analytics

- Big Data is a massive amount of data sets that cannot be stored, processed, or analyzed using traditional tools.
- **So, what makes data "big"?**

  Big data is characterized by the five V's: volume, velocity, variety, variability, and value. It's complex, so making sense of all of the data in the business requires both innovative technologies and analytical skills.
- Today, there are millions of data sources that generate data at a very rapid rate. These data sources are present across the world. Some of the largest sources of data are social media platforms and networks. Let's use Facebook as an example—it generates more than 500 terabytes of data every day. This data includes pictures, videos, messages, and more.
- Big data analytics describes the process of uncovering trends, patterns, and correlations in large amounts of raw data to help make data-informed decisions. These processes use familiar statistical analysis techniques—like clustering and regression—and apply them to more extensive datasets with the help of newer tools.

IOT ENABLING TECHNOLOGIES

Big Data Analytics

- Big data analytics refers to collecting, processing, cleaning, and analyzing large datasets to help organizations operationalize their big data.
  - ❑ Collect Data: Data collection looks different for every organization. With today's technology, organizations can gather both structured and unstructured data from a variety of sources — from cloud storage to mobile applications to in-store IoT sensors and beyond.
  - ❑ Process Data: Once data is collected and stored, it must be organized properly to get accurate results on analytical queries, especially when it's large and unstructured.
  - ❑ Clean Data: Data big or small requires scrubbing to improve data quality and get stronger results; all data must be formatted correctly, and any duplicative or irrelevant data must be eliminated or accounted for. Dirty data can obscure and mislead, creating flawed insights.

IOT ENABLING TECHNOLOGIES

Big Data Analytics

- Analyze Data: Getting big data into a usable state takes time. Once it's ready, advanced analytics processes can turn big data into big insights. Some of these big data analysis methods include:
  - **Data mining** sorts through large datasets to identify patterns and relationships by identifying anomalies and creating data clusters.
  - **Predictive analytics** uses an organization's historical data to make predictions about the future, identifying upcoming risks and opportunities.
  - **Deep learning** imitates human learning patterns by using artificial intelligence and machine learning to layer algorithms and find patterns in the most complex and abstract data.

IOT ENABLING TECHNOLOGIES

Big data analytics tools and technology

Big data analytics cannot be narrowed down to a single tool or technology. Instead, several types of tools work together to help you collect, process, cleanse, and analyze big data. Some of the major players in big data ecosystems are listed below.

- **Hadoop** is an open-source framework that efficiently stores and processes big datasets on clusters of commodity hardware. This framework is free and can handle large amounts of structured and unstructured data, making it a valuable mainstay for any big data operation.
- **NoSQL databases** are non-relational data management systems that do not require a fixed scheme, making them a great option for big, raw, unstructured data. NoSQL stands for "not only SQL," and these databases can handle a variety of data models.
- **MapReduce** is an essential component to the Hadoop framework serving two functions. The first is mapping, which filters data to various nodes within the cluster. The second is reducing, which organizes and reduces the results from each node to answer a query.
- **YARN** stands for "Yet Another Resource Negotiator." It is another component of second-generation Hadoop. The cluster management technology helps with job scheduling and resource management in the cluster.

IOT ENABLING TECHNOLOGIES

Big data analytics tools and technology

- **Spark** is an open source cluster computing framework that uses implicit data parallelism and fault tolerance to provide an interface for programming entire clusters. Spark can handle both batch and stream processing for fast computation.
- **Tableau** is an end-to-end data analytics platform that allows you to prep, analyze, collaborate, and share your big data insights. Tableau excels in self-service visual analysis, allowing people to ask new questions of governed big data and easily share those insights across the organization.
- **MongoDB** is used on datasets that change frequently
- **Talend** is used for data integration and management
- **Cassandra** is a distributed database used to handle chunks of data
- **STORM** is an open-source real-time computational system
- **Kafka** is a distributed streaming platform that is used for fault-tolerant storage

IOT ENABLING TECHNOLOGIES

The big benefits of big data analytics

- **Cost savings.** Helping organizations identify ways to do business more efficiently
- **Product development.** Providing a better understanding of customer needs
- **Market insights.** Tracking purchase behavior and market trends
- **Strategic business decisions:** The ability to constantly analyze data helps businesses make better and faster decisions, such as cost and supply chain optimization.
- **Customer experience:** Data-driven algorithms help marketing efforts (targeted ads, as an example) and increase customer satisfaction by delivering an enhanced customer experience.
- **Risk management:** Businesses can identify risks by analyzing data patterns and developing solutions for managing those risks.

IOT ENABLING TECHNOLOGIES

Artificial Intelligence & Machine Learning

- **Artificial intelligence** as an academic discipline was founded in 50s. Actually the "AI" term was coined by John McCarthy, an American computer scientist, back in 1956 at The Dartmouth Conference. According to John McCarthy, AI is "The science and engineering of making intelligent machines, especially intelligent computer programs".
- **Machine Learning** is a subset of AI, and consists of the more advanced techniques and models that enable computers to figure things out from the data and deliver AI applications. ML is the science of getting computers to act without being explicitly programmed.
- **Deep Learning** is a newer area of ML that that uses multi-layered artificial neural networks to deliver high accuracy in tasks such as object detection, speech recognition, language translation and other recent breakthroughs that you hear in the news. Beauty and strength of DL is they can automatically learn/extract/translate the features from data sets such as images, video or text, without introducing traditional hand-coded code or rules.

IOT ENABLING TECHNOLOGIES