Unit – I
Introduction to Internet of Things (IoT)

1. Introduction to IoT: Definition, Vision & characteristics of IoT, IBM IoT Conceptual Framework, IoT Architectural View.
2. Physical design of IoT: Things in IoT, IoT Protocols.
3. Logical design of IoT: IoT Fundamental blocks, IoT Communication
4. Model, IoT Communication API's
5. IoT Enabling Technologies: Wireless Sensor Networks, Cloud Computing, Big Data Analytics, Communication Protocols, Embedded Systems
6. IoT Levels and Deployment templates — IoT Level-1, IoT Level-2, IoT Level-3, IoT Level-4, IoT Level-5, IoT Level-6.
7. Challenges in IoT.

Unit No.	Unit Title	Teaching Hours	Distribution of Theory Marks			
			R Level	U Level	A Level	Total Marks
I	**Introduction to Internet of Things**	08	04	04	06	**14**

Introduction to IoT:

Definition:

Connecting everyday things embedded with electronics, software and sensors to the internet enabling them to collect and exchange data.

OR

The Internet of things refers to a type of network to connect anything with the Internet based on stipulated protocols through information sensing equipment to conduct information exchange and communications in order to achieve smart recognitions, positioning, tracing, monitoring, and administration

Characteristics of IoT:

- **Connectivity** Things of IoT should be connected to the IoT infrastructure. Anyone, anywhere, anytime can connect, this should be guaranteed at all times. For example, connection between people through internet devices likes mobile phones and other gadgets, also connection between Internet devices such as routers, gateways, sensors, etc.

- **Intelligence and Identity**
  The extraction of knowledge from the generated data is very important. For example, a sensor generates data, but that data will only be useful if it is interpreted properly. Each IoT device has a unique identity. This identification is helpful in tracking the equipment and at times for querying its status.

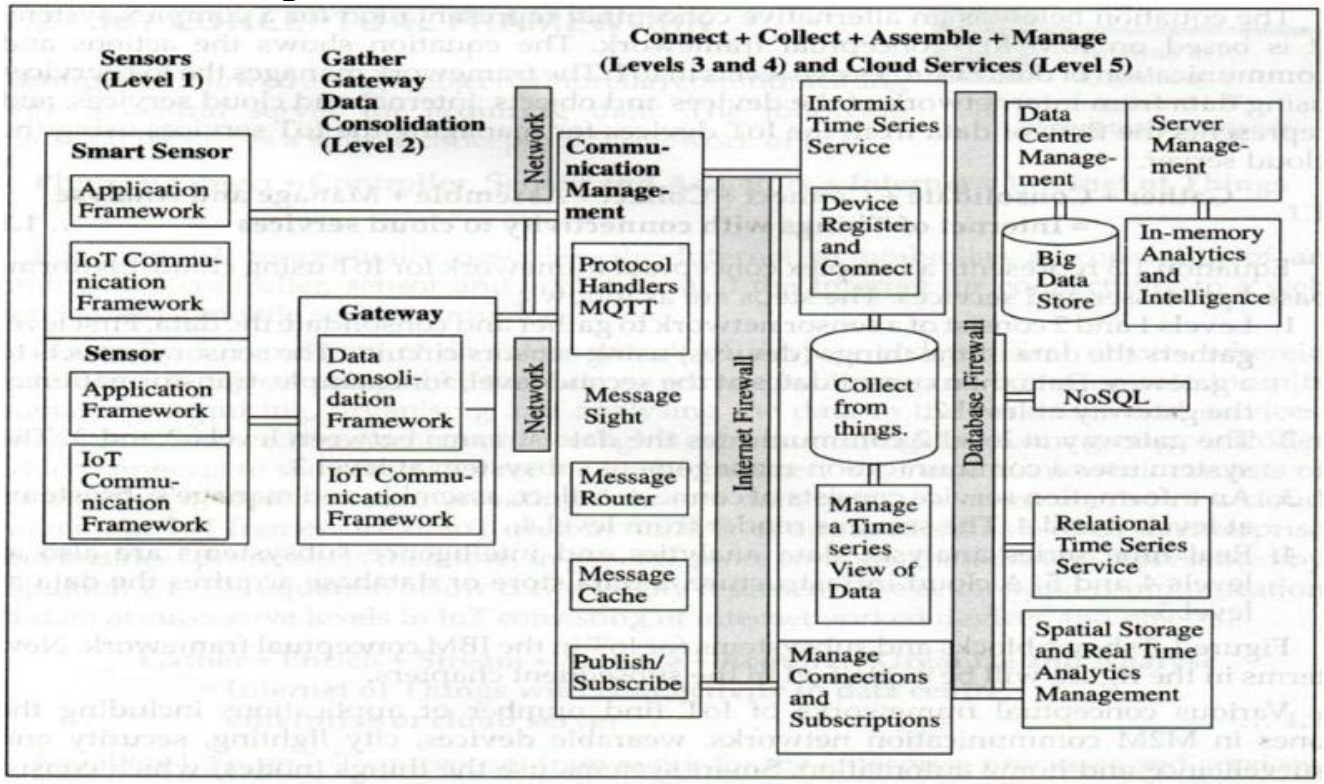
- **Scalability**
  The number of elements connected to the IoT zone is increasing day by day. Hence, an IoT setup should be capable of handling the massive expansion. The data generated as an outcome is enormous, and it should be handled appropriately.
- **Dynamic and Self-Adapting (Complexity)**
  IoT devices should dynamically adapt themselves to the changing contexts and scenarios. Assume a camera meant for the surveillance. It should be adaptable to work in different conditions and different light situations (morning, afternoon, night).
- **Architecture** –
  IoT architecture cannot be homogeneous in nature. It should be hybrid, supporting different manufacturers 'products to function in the IoT network. IoT is not owned by anyone engineering branch. IoT is a reality when multiple domains come together.
- **Safety**
  There is a danger of the sensitive personal details of the users getting compromised when all his/her devices are connected to the internet. This can cause a loss to the user. Hence, data security is the major challenge. Besides, the equipment involved is huge. IoT networks may also be at the risk. Therefore, equipment safety is also critical.
- **Self-Configuring**
  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.
- **Interconnectivity:**
  With regard to the IoT, anything can be interconnected with the global information and communication infrastructure.

Characteristics	Features	Advantages	Disadvantages
Intelligence	Device Virtualisation	Integration of Devices	Security
Connectivity	High Speed	Reliable, Secure Bi-Directional Communication	Privacy
Sensing	End Point Management	Enhanced Data Collection	Complexity
Expressing	Small Devices	Reduce Waste	Flexibility
Energy	Stream Processing	Real Time Analysis Improved Engagement	Compliance
Safety	Data Enrichment	Automation and Technology Optimization	Tier Management

IBM IoT Conceptual Framework, IoT Architectural View.

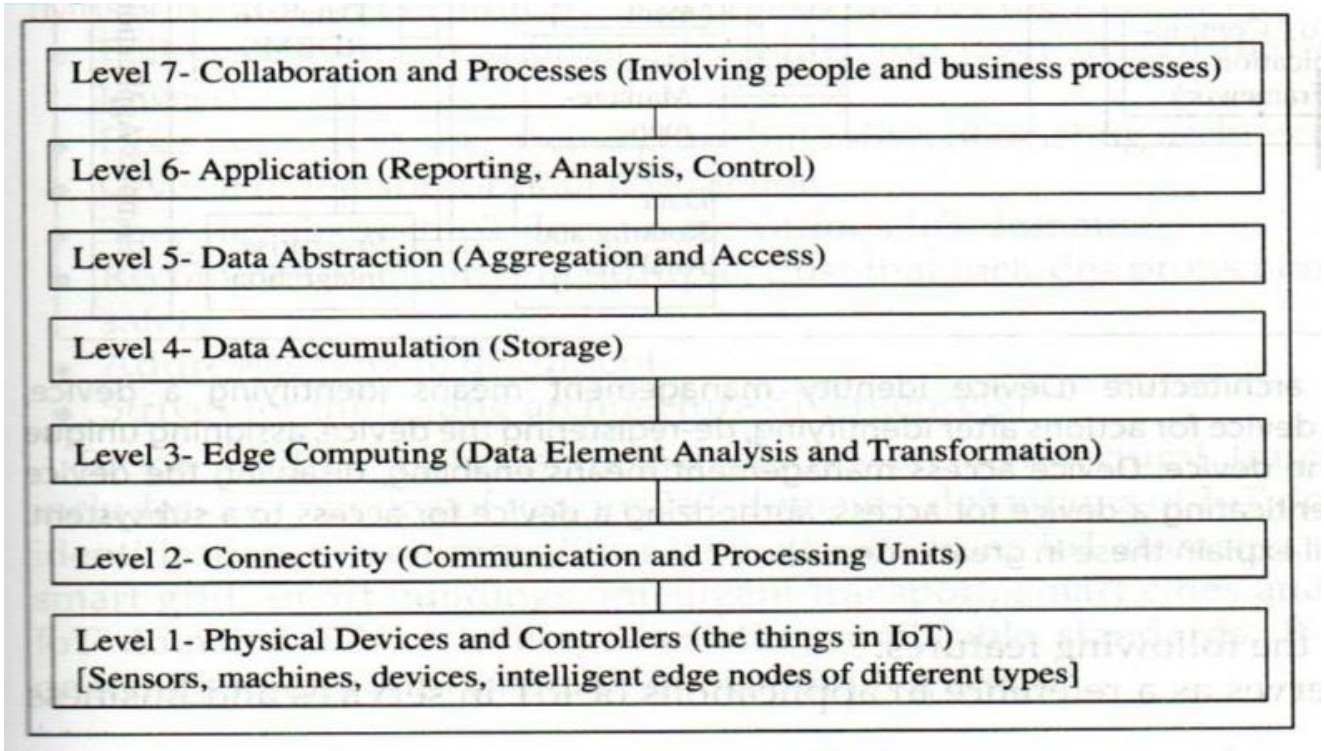
Physical Object + Controller, Sensor and Actuators + Internet = Internet of Things

Gather + Consolidate + Connect + Collect + Assemble + Manage and Analyse = Internet of Things connected to Cloud Services



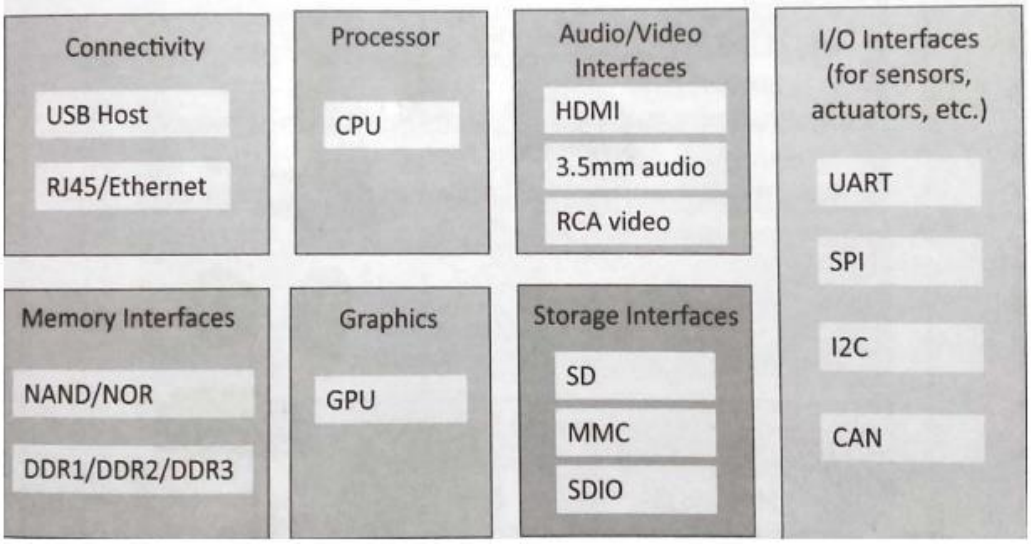
IBM IoT Conceptual Framwork

IoT Architectural View.



Physical Design of IoT

1) Things in IoT:

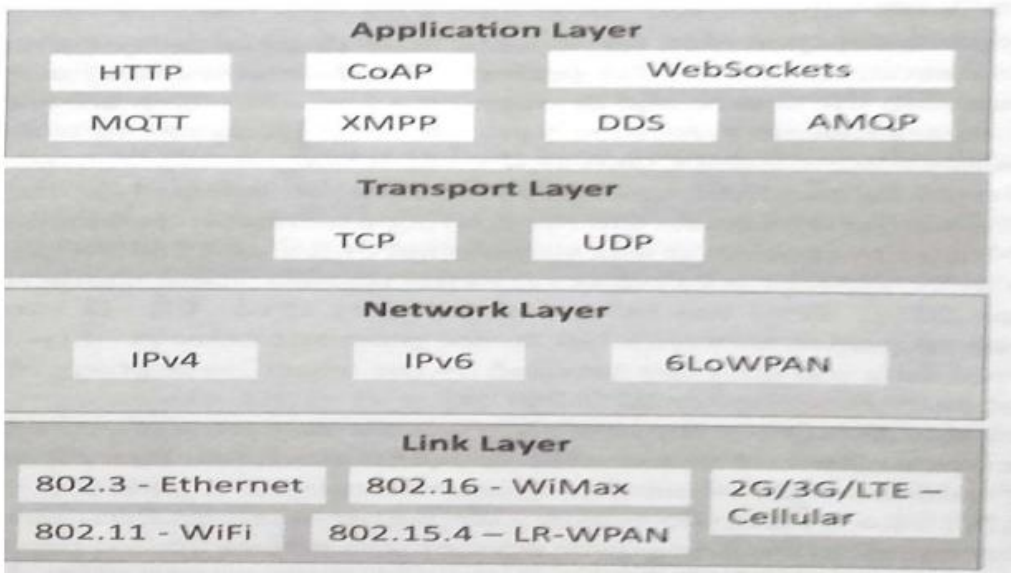


The things in IoT refers to IoT devices which have unique identities and perform remote sensing, actuating and monitoring capabilities. IoT devices can exchange data with other connected devices applications. It collects data from other devices and process data either locally or remotely.

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 and
(iv) audio/video interfaces.

2) IoT Protocols:



a) 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 signaled by the h/w device over the medium to which the host is attached.

Protocols:

- **802.3-Ethernet:**

IEEE802.3 is collection of wired Ethernet standards for the link layer.
Eg: 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.
Eg: 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 60Ghzband.

- **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.

- **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 to250kb/s.

- **2G/3G/4G-Mobile Communication:**

Data rates from 9.6kb/s(2G) to up to100Mb/s(4G).

b) Network/Internet Layer:

Responsible for sending IP datagrams from source n/w to destination n/w. 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**32addresses.

- **IPv6:**

Internet Protocol version6 uses 128 bit address scheme and allows 2**128 addresses.

- **6LOWPAN:**

(IPv6overLowpowerWirelessPersonalAreaNetwork)operates in 2.4 GHz frequency range and data transfer 250 kb/s.

c) **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. Avoids n/w congestion and congestion collapse.

- **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.

d) **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. **Protocols:**

- HTTP:

Hyper Text Transfer Protocol that forms foundation of WWW. Follow requestresponse model Stateless protocol.

- **CoAP:**

Constrained Application Protocol for machine-to-machine (M2M) applications with constrained devices, constrained environment and constrained n/w. Uses client server architecture.

- **WebSocket**:

Allows full duplex communication over a single socket connection. • MQTT: Message Queue Telemetry Transport is light weight messaging protocol based on publish-subscribe model. Uses client server architecture. Well suited for constrained environment.

- **XMPP:**

Extensible Message and Presence Protocol for real time communication and streaming XML data between network entities. Support client-server and server-server communication.

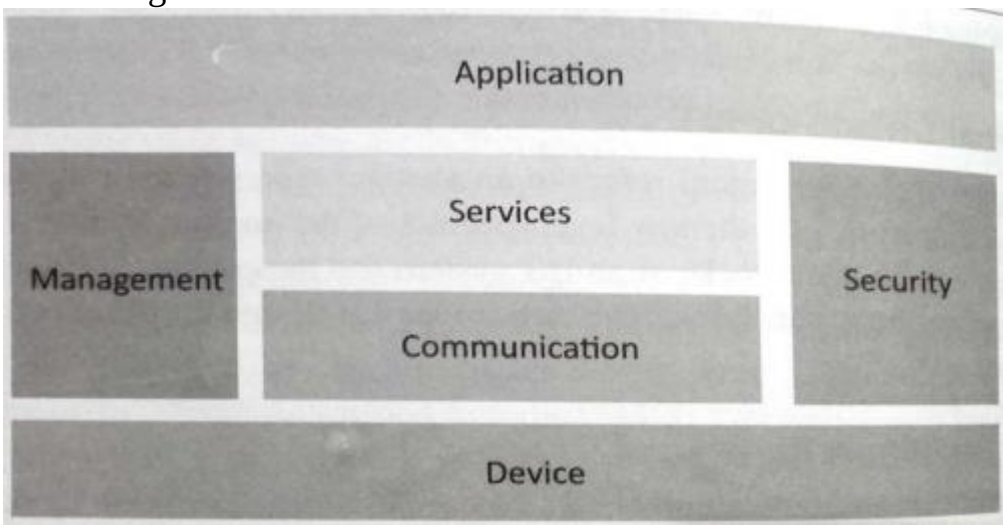
• **DDS**: Data Distribution Service is data centric middleware standards for device-to-device or machine-to-machine communication. Uses publish-subscribe model.
• AMQP: Advanced Message Queuing Protocol is open application layer protocol for business messaging. Supports both point-to-point and publish-subscribe model.

LOGICAL DESIGN of IoT

- IoT Functional Blocks
- IoT Communication Models
- IoT Comm. APIs

IoT Functional Blocks:

Provide the system the capabilities for identification, sensing, actuation, communication and management.



- Device: An IoT system comprises of devices that provide sensing, actuation, monitoring and control functions.
- Communication: handles the communication for IoTsystem.
- Services: for device monitoring, device control services, data publishing services and services for device discovery.
- Management: Provides various functions to govern the IoT system.
- Security: Secures IoT system and priority functions such as authentication , authorization, message and context integrity and data security.
- Application: IoT application provide an interface that the users can use to control and monitor various aspects of IoT system.

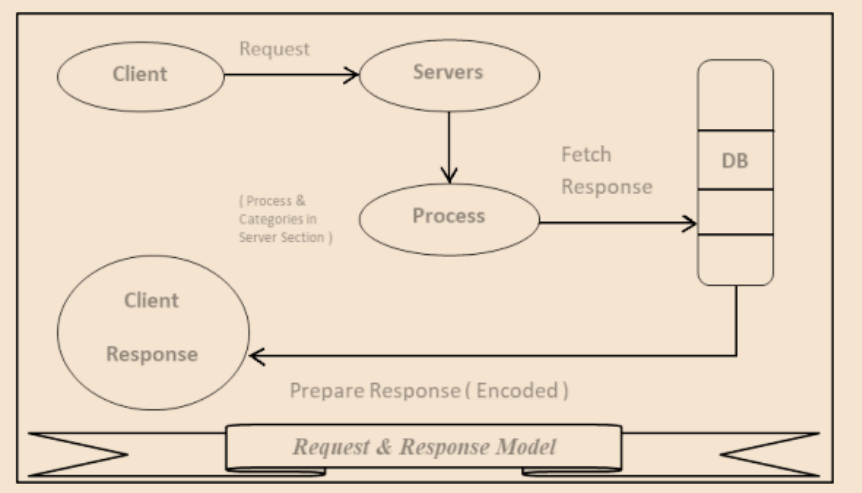
IoT Communication Models:

- **Request-Response**
- **Publish-Subscibe**
- **Push-Pull**
- **ExclusivePair**

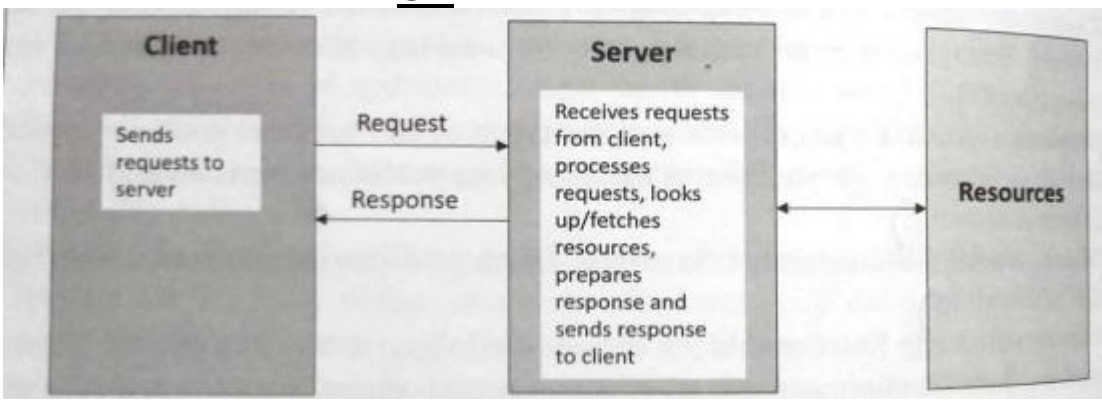
1) Request-Response Model:

This model follows client-server architecture.

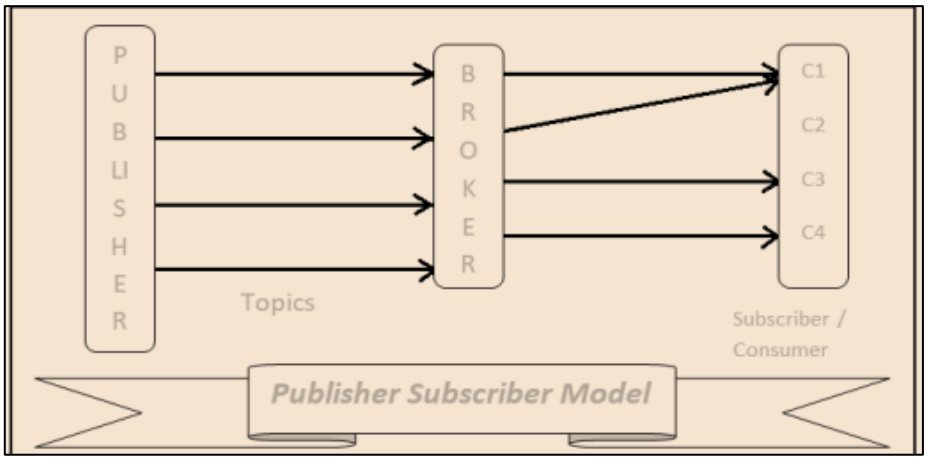
- The client, when required, requests the information from the server. This request is usually in the encoded format.
- This model is stateless since the data between the requests is not retained and each request is independently handled.
- The server Categories the request, and fetches the data from the database and its resource representation. This data is converted to response and is transferred in an encoded format to the client. The client, in turn, receives the response.
- On the other hand — In Request-Response communication model client sends a request to the server and the server responds to the request. When the server receives the request it decides how to respond, fetches the data retrieves resources, and prepares the response, and sends it to the client.



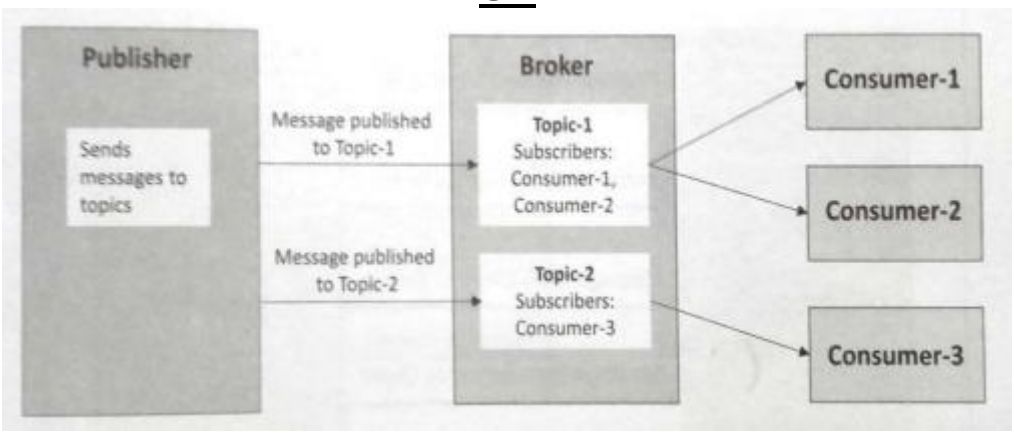
OR



2) Publish-Subscibe Model:



OR



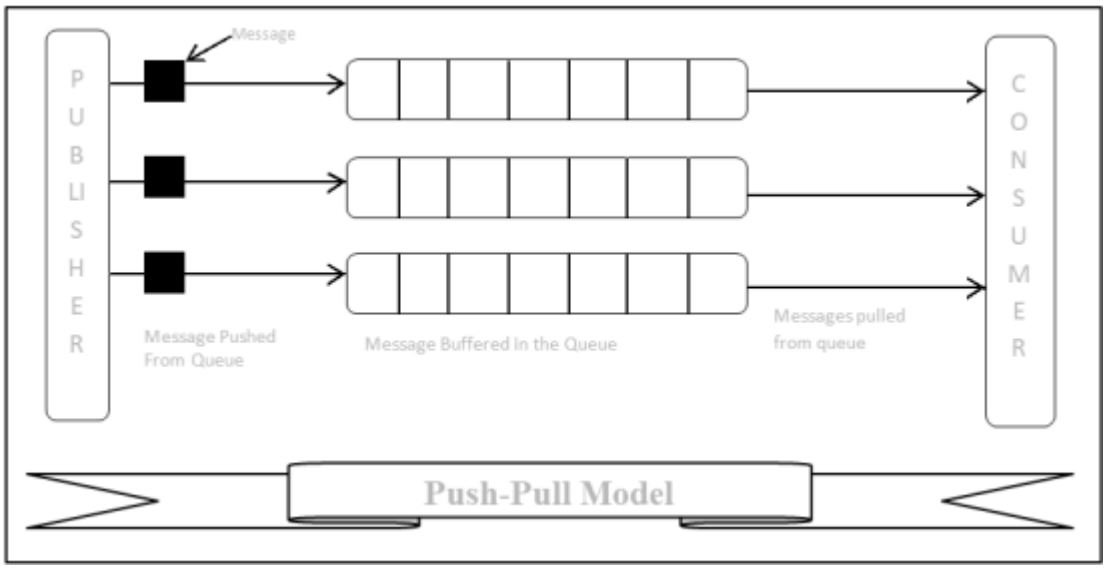
This model comprises three entities: Publishers, Brokers, and Consumers.

- Publishers are the source of data. It sends the data to the topic which are managed by the broker. They are not aware of consumers.
- Consumers subscribe to the topics which are managed by the broker.
- Hence, Brokers responsibility is to accept data from publishers and send it to the appropriate consumers. The broker only has the information regarding the consumer to which a particular topic belongs to which the publisher is unaware of.

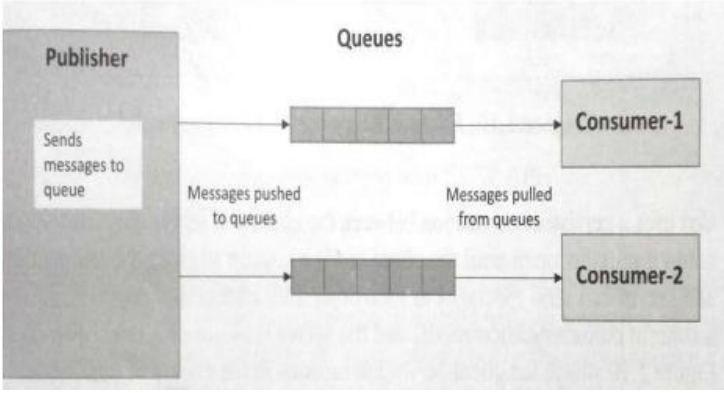
3) Push-Pull Model:

The push-pull model constitutes data publishers, data consumers, and data queues.

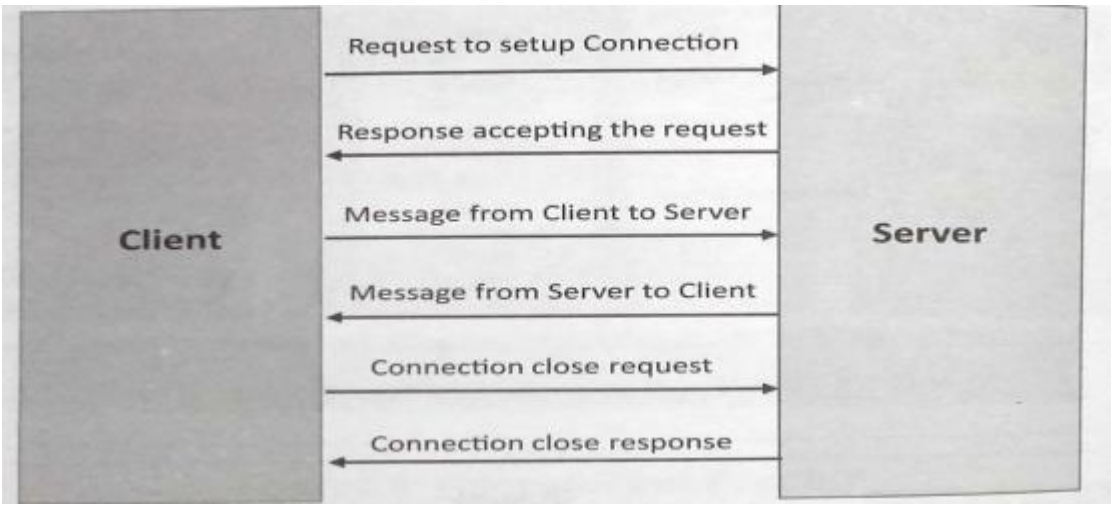
- Publishers and Consumers are not aware of each other.
- Publishers publish the message/data and push it into the queue. The consumers, present on the other side, pull the data out of the queue. Thus, the queue acts as the buffer for the message when the difference occurs in the rate of push or pull of data on the side of a publisher and consumer.
- Queues help in decoupling the messaging between the producer and consumer. Queues also act as a buffer which helps in situations where there is a mismatch between the rate at which the producers push the data and consumers pull the data.



OR



4) Exclusive Pair:



Exclusive Pair is the bi-directional model, including full-duplex communication among client and server. The connection is constant and remains open till the client sends a request to close the connection.

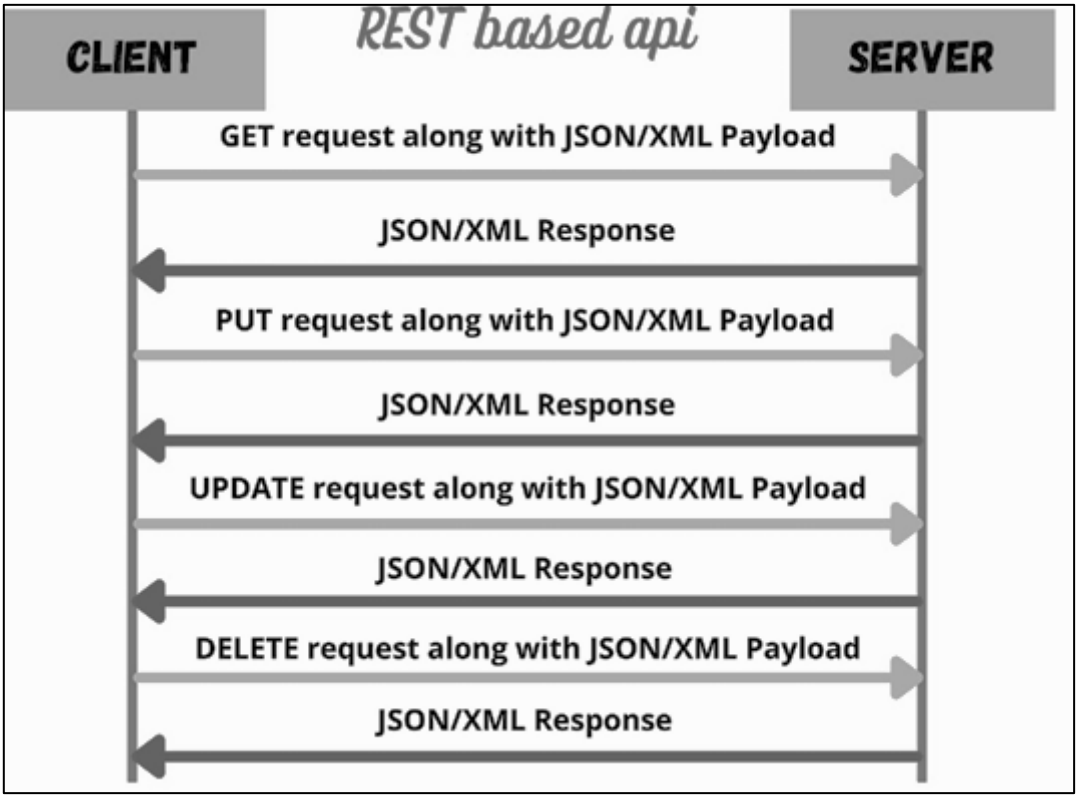
- The Server has the record of all the connections which has been opened.
- This is a state-full connection model and the server is aware of all open connections.
- WebSocket based communication API is fully based on this model.

IoT Comm. APIs

- An API is an interface used by programs to access an application.
- It enables a program to send commands to another program and receive replies from the app.
- IoT APIs are the interface points between an IoT device and the Internet and/or other network components.

Here we will talk about the REST-based API and the Websocket based API.

REST-based API



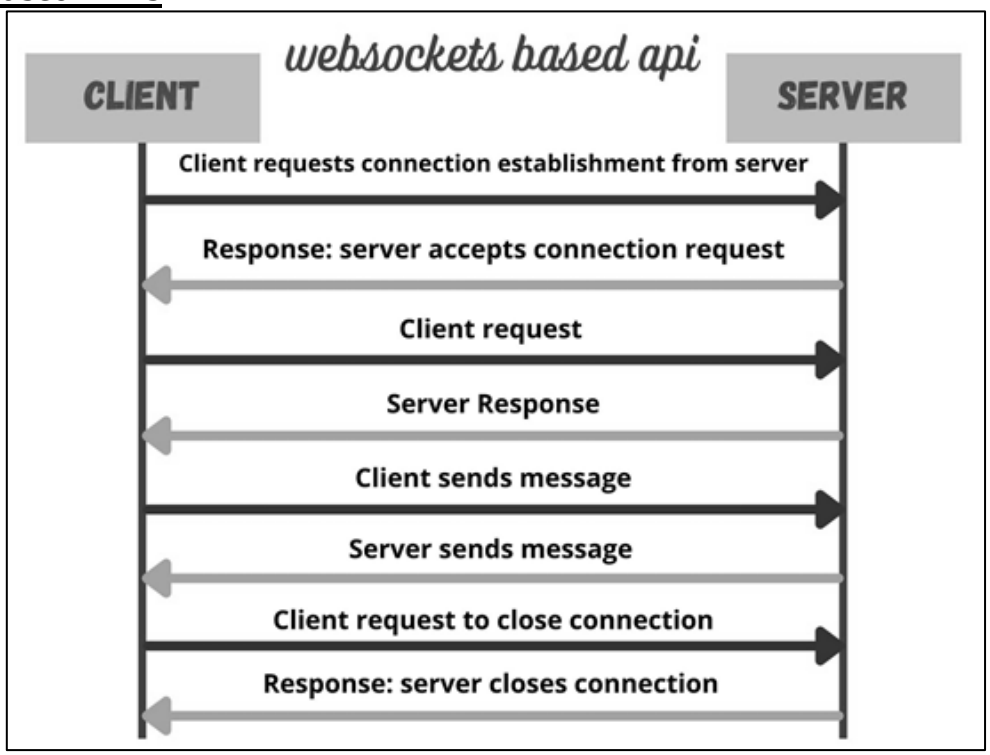
- Representational state transfer (REST) is a set of architectural principles by which you can design Web services the Web APIs that focus on the system's resources and how resource states are addressed and transferred.
- URIs(example:- example.com/api/tasks) are used to depict resources in the RESTful web service.
- Client tries to access these resources via URIs using commands like GET, PUT, POST, DELETE and so on that are defined by HTTP.
- In response, the server responds with a JSON object or XML file.
- The REST APIs follow the request-response model.

The rest architectural constraints are as follows:

- Client-server
  Let me explain it to you by giving a suitable example. The client should not be concerned with the storage of data which is a concern of the server, similarly, the server should not be concerned about the user interface, which is the concern of the client. Separation makes it possible for the client and server to be developed and updated independently.
- Stateless
  The status of the session remains entirely on the client.
- Cache-able
  This property defines whether the response to any request can be cached or not. If a response can be cached, then a client cache is granted the right to reuse that response data for subsequent matching requests.
- Layered system
  A layered system defines the boundaries of the components within each specific layer. For example, A client is unable to tell whether it is connected to the end server or an intermediate node. As simple as that!

- Uniform interface
  This specifies that the technique of communication between a client and a server must be uniform throughout the communication period.
- Code on Demand(Optional Constraint)
  Servers may provide executable code or scripts for execution by clients in their context.

Websocket based APIs



- Websocket APIs enable bi-directional and duplex communication between customers and servers.
- Unlike REST, There is no need to set up a connection every now and then to send messages between a client and a server.
- It works on the principle of the exclusive pair model. Can you recall it? Yes. Once a connection is set up, there is a constant exchange of messages between the client and the server. All we need is to establish a dedicated connection to start the process. the communication goes on unless the connection is terminated.
- It is a stateful type.
- Due to one time dedicated connection setup, there is less overhead, lower traffic and less latency and high throughput.
- So Web socket is the most suitable IoT Communication APIs for IoT System.

IoT Enabling Technologies

IoT is enabled by several technologies including Wireless Sensor Networks, Cloud Computing, Big Data Analytics, Embedded Systems, Security Protocols and architectures, Communication Protocols, Web Services, Mobile internet and semantic search engines.

1. Wireless Sensor Network (WSN): Comprises of distributed devices with sensors which are used to monitor the environmental and physical conditions.
Zig Bee is one of the most popular wireless technologies used by WSNs.
WSNs used in IoT systems are described as follows:

- Weather Monitoring System: in which nodes collect temp, humidity and other data, which is aggregated and analyzed.
- Indoor air quality monitoring systems: to collect data on the indoor air quality and concentration of various gases.
- Soil Moisture Monitoring Systems: to monitor soil moisture at various locations.
- Surveillance Systems: use WSNs for collecting surveillance data(motion data detection).
- Smart Grids : use WSNs for monitoring grids at various points.
- Structural Health Monitoring Systems: Use WSNs to monitor the health of structures(building, bridges) by collecting vibrations from sensor nodes deployed at various points in the structure.

2. Cloud Computing: Services are offered to users in different forms.

- Infrastructure-as-a-service (IaaS): provides users the ability to provision computing and storage resources. These resources are provided to the users as a virtual machine instances and virtual storage.
- Platform-as-a-Service (PaaS): provides users the ability to develop and deploy application in cloud using the development tools, APIs, software libraries and services provided by the cloud service provider.
- Software-as-a-Service (SaaS): provides the user a complete software application or the user interface to the application itself.

3. Big Data Analytics: Some examples of big data generated by IoT are

- Sensor data generated by IoT systems.
- Machine sensor data collected from sensors established in industrial and energy systems.
- Health and fitness data generated IoT devices.
- Data generated by IoT systems for location and tracking vehicles.
- Data generated by retail inventory monitoring systems.

4. Communication Protocols: form the back-bone of IoT systems and enable network connectivity and coupling to applications.

- Allow devices to exchange data over network.
- Define the exchange formats, data encoding addressing schemes for device and routing of packets from source to destination.
- It includes sequence control, flow control and retransmission of lost packets.

5. Embedded Systems: is a computer system that has computer hardware and

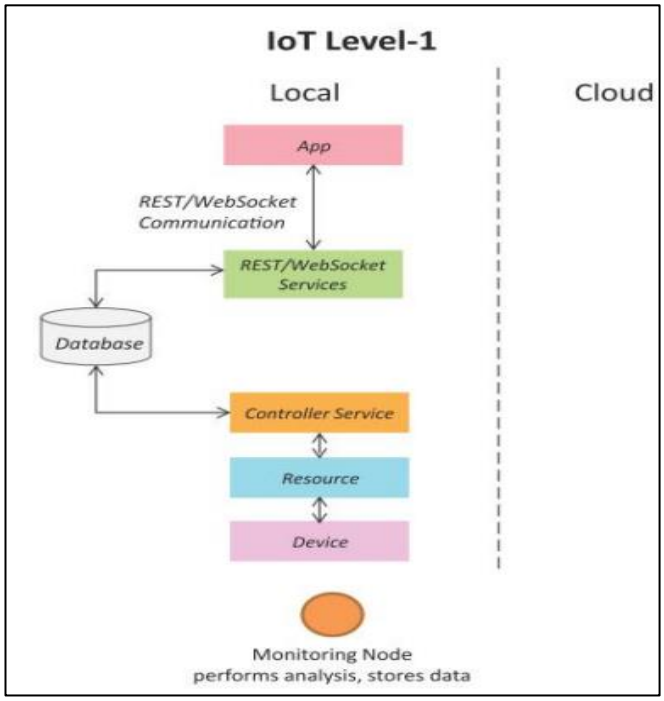
software embedded to perform specific tasks. Embedded System range from low cost miniaturized devices such as digital watches to devices such as digital cameras, POS terminals, vending machines, appliances etc.,

IoT Levels and Deployment templates

IoT Level-1

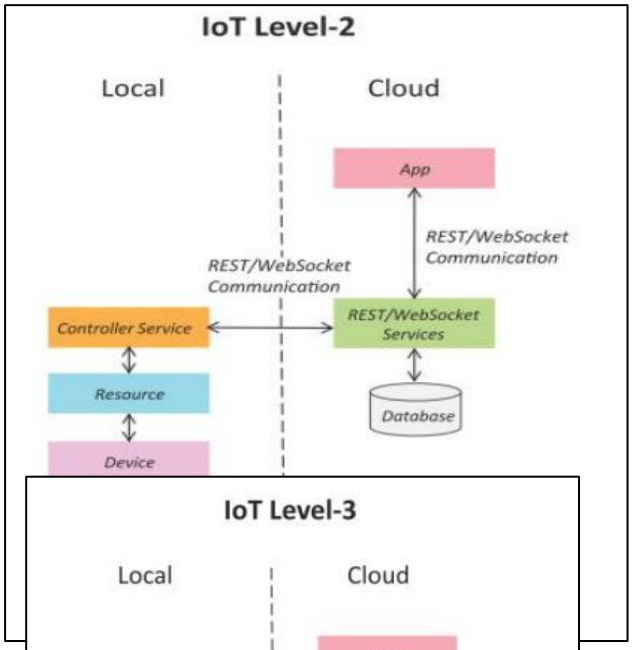
• A level-1 IoT system has a single node/device that performs sensing and/or actuation, stores data,performs analysis and hosts the application

• Level-1 IoT systems are suitable for modeling low- cost and low-complexity solutions where the data involved is not big and the analysis requirements are not computationally intensive.



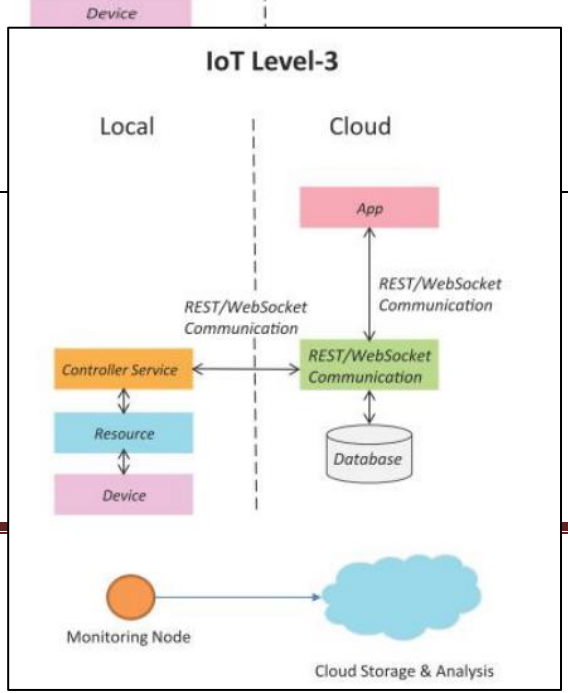
IoT Level-2

- A level-2 IoT system has a single node that performs sensing and/or actuation and local analysis.
- Data is stored in the cloud and application is usually cloud- based.
- Level-2 IoT systems are suitable for solutions where the data involved is big, however, the primary analysis requirement is not computationally intensive and can be done locally itself.



IoT Level-3

- A level-3 IoT system hasa single node. Data is stored and analyzed in the cloud and application is

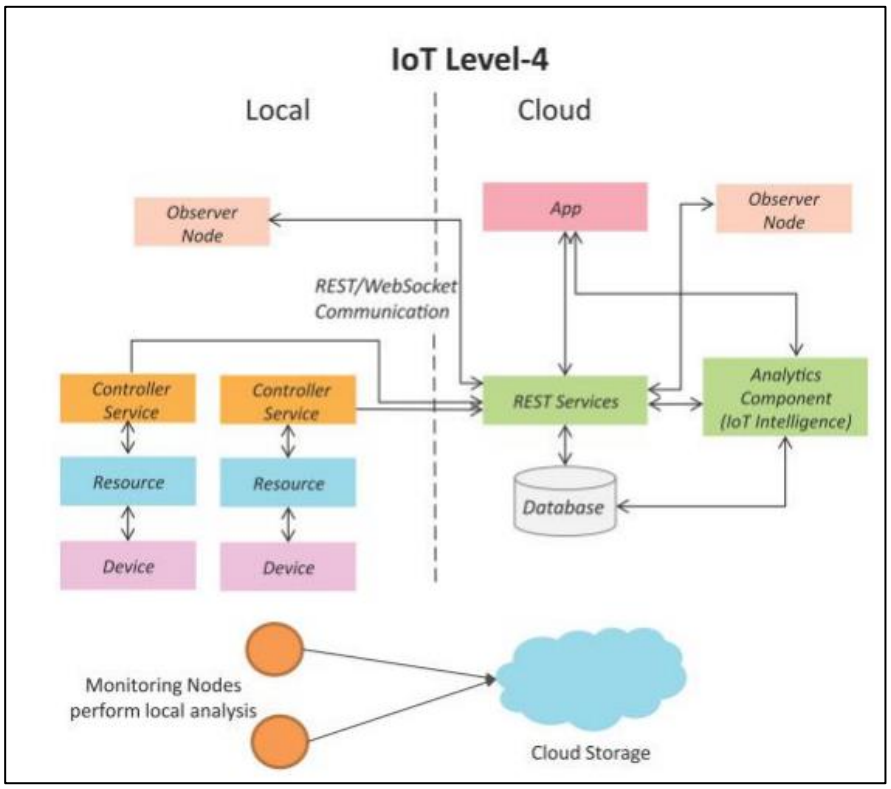


cloud- based.

- Level-3 IoT systems are suitable for solutions where the data involved is big and the analysis requirements are computationally intensive.

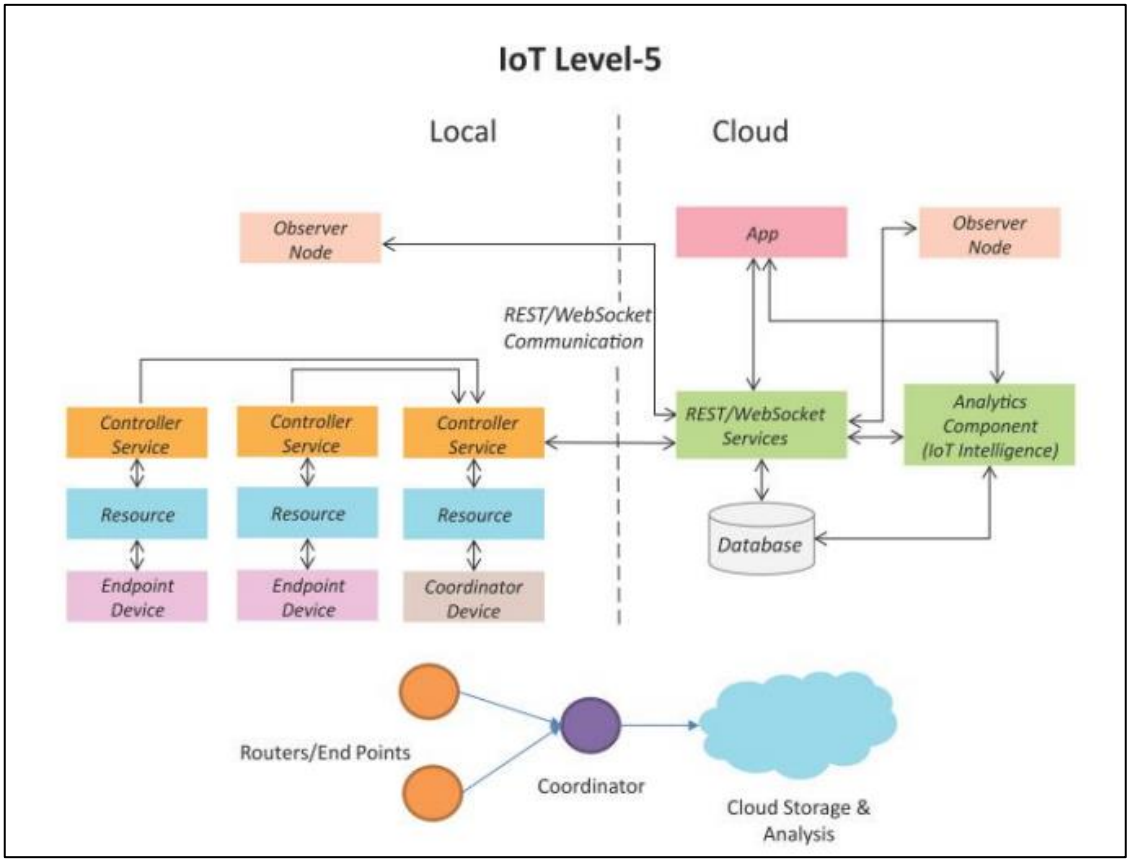
IoT Level-4

- A level-4 IoT system has multiple nodes that perform local analysis. Data is stored in the cloud and application is cloud-based.
- Level-4 contains local and cloud-based observer nodes which can subscribe to and receive information collected in the cloud from IoT devices.
- Level-4 IoT systems are suitable for solutions where multiple nodes are required, the data involved is big and the analysis requirements are computationally intensive.



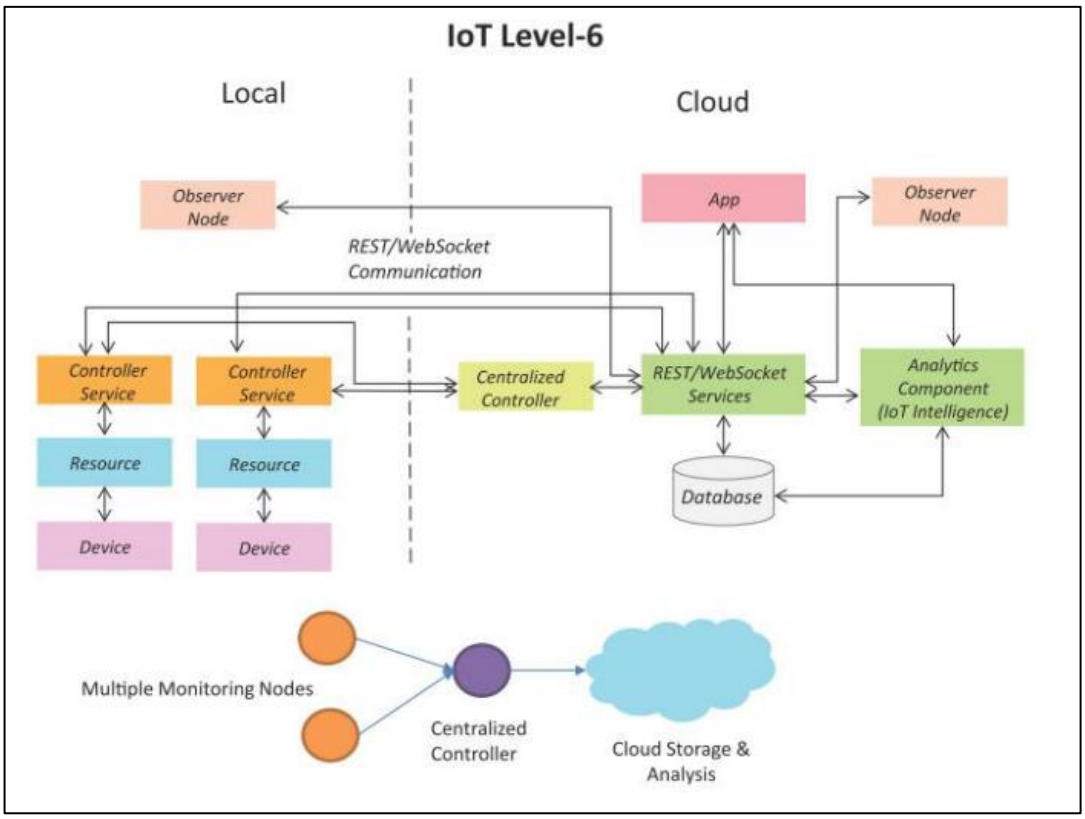
IoT Level-5

- A level-5 IoT system has multiple end nodes and one coordinator node.
- The end nodes that perform sensing and/or actuation.
- Coordinator node collects data from the end nodes and sends to the cloud.
- Data is stored and analyzed in the cloud and application is cloud-based.
- Level-5 IoT systems are suitable for solutions based on wireless sensor networks, in which the data involved is big and the analysis requirements are computationally intensive.



IoT Level-6

- A level-6 IoT system has multiple independent end nodes that perform sensing and/or actuation and send data to the cloud.
- Data is stored in the cloud and application is cloud-based.
- The analytics component analyzes the data and stores the results in the cloud database.
- The results are visualized with the cloud-based application.
- The centralized controller is aware of the status of all the end nodes and sends control commands to the nodes.



Challenges in IoT

- **Security challenges in IoT :**

  1. Lack of encryption –

  Although encryption is a great way to prevent hackers from accessing data, it is also one of the leading IoT security challenges.

  These drives like the storage and processing capabilities that would be found on a traditional computer.

  The result is an increase in attacks where hackers can easily manipulate the algorithms that were designed for protection.

  2. Insufficient testing and updating –

  With the increase in the number of IoT(internet of things) devices, IoT manufacturers are more eager to produce and deliver their device as fast as they can without giving security too much of although.

  Most of these devices and IoT products do not get enough testing and updates and are prone to hackers and other security issues.

  3. Brute forcing and the risk of default passwords –

  Weak credentials and login details leave nearly all IoT devices vulnerable to password hacking and brute force.

  Any company that uses factory default credentials on their devices is placing both their business and its assets and the customer and their valuable information at risk of being susceptible to a brute force attack.

  4. IoT Malware and ransomware –

  Increases with increase in devices.

  Ransomware uses encryption to effectively lock out users from various devices and

platforms and still use a user's valuable data and info.

Example –

A hacker can hijack a computer camera and take pictures.

By using malware access points, the hackers can demand ransom to unlock the device and return the data.

5. **IoT botnet aiming at cryptocurrency –**

IoT botnet workers can manipulate data privacy, which could be massive risks for an open Crypto market. The exact value and creation of cryptocurrencies code face danger from mal-intentioned hackers.

The blockchain companies are trying to boost security. Blockchain technology itself is not particularly vulnerable, but the app development process is.

Design challenge in IoT :

1. **Battery life is a limitation –**

Issues in packaging and integration of small-sized chip with low weight and less power consumption. If you've been following the mobile space, you've likely see how every yr it looks like there's no restriction in terms of display screen size. Take the upward thrust of 'phablets', for instance, which can be telephones nearly as huge as tablets. Although helpful, the bigger monitors aren't always only for convenience, rather, instead, display screen sizes are growing to accommodate larger batteries. Computers have getting slimmer, but battery energy stays the same.

2. **Increased cost and time to market –**

Embedded systems are lightly constrained by cost.

The need originates to drive better approaches when designing the IoT devices in order to handle the cost modelling or cost optimally with digital electronic components.

Designers also need to solve the design time problem and bring the embedded device at the right time to the market.

3. **Security of the system –**

Systems have to be designed and implemented to be robust and reliable and have to be secure with cryptographic algorithms and security procedures.

It involves different approaches to secure all the components of embedded systems from prototype to deployment.

Deployment challenges in IoT :

1. **Connectivity –**

It is the foremost concern while connecting devices, applications and cloud platforms.

Connected devices that provide useful front and information are extremely valuable. But poor connectivity becomes a challenge where IoT sensors are required to monitor process data and supply information.

2. **Cross platform capability –**

IoT applications must be developed, keeping in mind the technological changes of the future.

Its development requires a balance of hardware and software functions.

It is a challenge for IoT application developers to ensure that the device and IoT platform drivers the best performance despite heavy device rates and fixings.

3. **Data collection and processing –**

In IoT development, data plays an important role. What is more critical here is the

processing or usefulness of stored data.
Along with security and privacy, development teams need to ensure that they plan well for the way data is collected, stored or processed within an environment.

4. **Lack of skill set –**

All of the development challenges above can only be handled if there is a proper skilled resource working on the IoT application development.
The right talent will always get you past the major challenges and will be an important IoT application development asset.