Unit 1

1. Definition of IoT. *
    - A dynamic global network infrastructure with self-configuring capabilities based on standard communication protocols where physical and virtual things have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network, often used to communicate data associated with users and their environments.
2. Characteristics of IoT. *
    - **Dynamic and self-adapting**: IoT devices and systems may have the capability to dynamically adapt with the changing contexts and take actions based on their operating conditions, user's context, or sensed environment.
    - **Self-configuring**: IoT devices may have self-configuring capability, allowing a large number of devices to work together to provide certain functionality.
    - **Interoperable communication protocols**: IoT devices may support a number of interoperable communication protocols and can communicate with other devices.
    - **Unique identity**: Each IoT device has a unique identity and a unique identifier such as an IP address.
    - **Integrated into information network**: IoT devices are usually integrated into the information network that allows them to communicate and exchange data with other devices and systems. IoT devices can be dynamically discovered in the network, by other devices and/or the network, and have the capability to describe themselves to other devices or user applications.
3. Things in IoT. *
    - The "things" in IoT usually refers to IoT devices which have unique identities and can perform remote sensing, actuating and monitoring capabilities. IoT devices can exchange data with other connected devices and applications, or collect data from other devices and process the data either locally or send it on cloud-based applications.
    - IoT devices may consist of several interfaces for connections to other devices, both wired and wireless. These include:
        a. Input output interfaces for sensors.
        b. Interfaces for internet connectivity
        c. Memory and storage interfaces
        d. Audio/video interfaces
    - IoT devices can also be of varied types, for instance, wearable sensors, smart watches, LED lights, automobiles and industrial machines.
4. IoT protocols.
    - Link layer: Link layer protocols determine how the data is physically sent over the network's physical layer or medium.
    - The scope of the link layer is the local network connection 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 the packets are coded and signaled by the hardware device over the medium to which the host is attached.

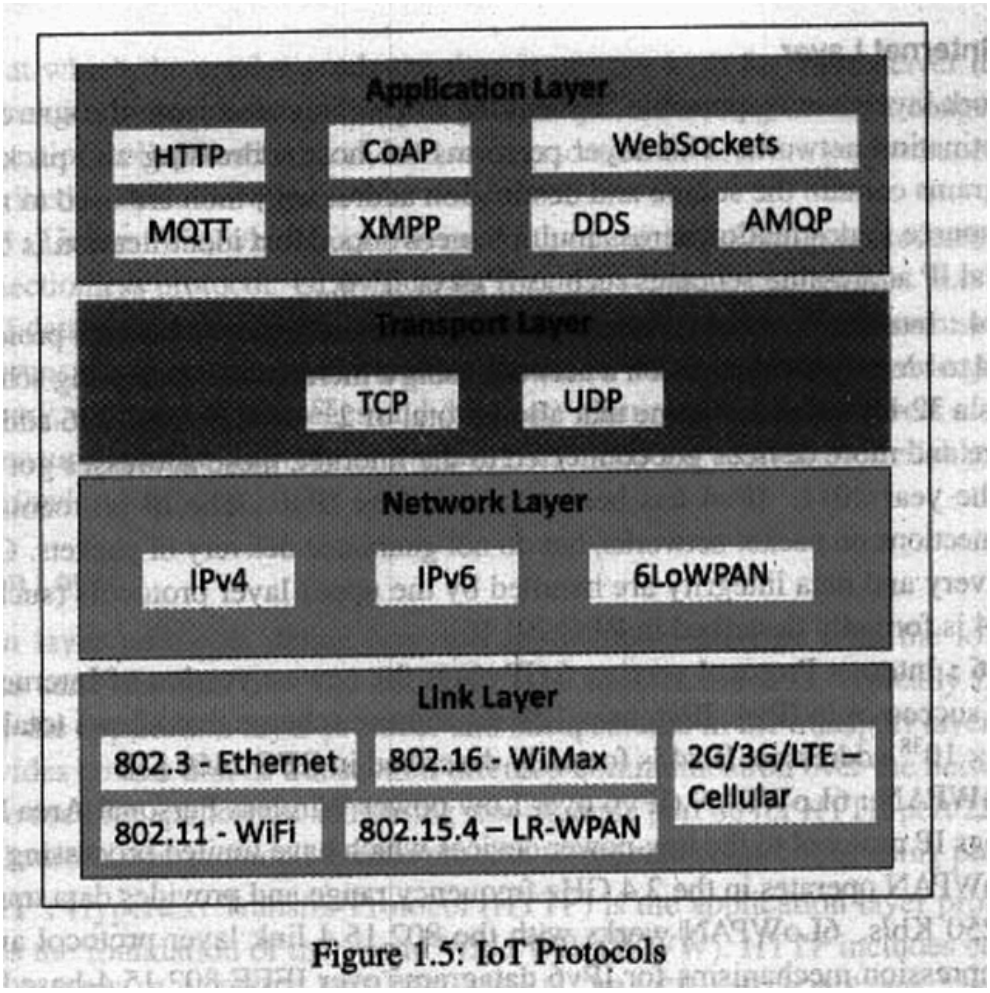

Figure 1.5: IoT Protocols

-

5. Transport layer. *
    - **TCP**: Transmission control protocol is the most widely used transport layer protocol that is used by web browsers, email programs and file transfer. TCP is a connection oriented and stateful protocol. While IP protocol deals with sending packets, TCP ensures reliable transmission of packets in order. TCP also provides error detection capability so that duplicate packets can be discarded and lost packets are retransmitted. TCP helps in avoiding network congestion and congestion collapse which can lead to degradation of network performance.
    - **UDP (User Datagram Protocol):** Unlike TCP, which requires carrying out an initial setup procedure, UDP is a connectionless protocol. UDP is useful for time-sensitive applications that have very small data units to exchange and do not want the overhead of connection setup. UDP is a transaction oriented and stateless protocol. UDP does not provide guaranteed delivery, ordering of messages and duplicate elimination.
6. Application layer. *

Application layer protocols define how the applications interface with the lower layer protocols to send the data over the network.

- **HTTP:** HyperText Transfer Protocol (HTTP) is the application layer protocol that forms the foundation of the world wide web. HTTP includes commands such as GET, PUT, POST, DELETE, HEAD etc. The protocol follows a request-response model where a client sends requests to a server using the HTTP commands. HTTP is a stateless protocol and each HTTP request is independent of the other requests.
- **CoAP**: Constrained Application Protocol is an application layer protocol for machine to machine applications, meant for constrained environments with constrained devices and constrained networks. Like HTTP, CoAP is a web transfer protocol and uses a request-response model, however it runs on top of UDP instead of TCP. CoAP uses a client-server architecture where clients communicate with servers using connectionless datagrams.
- **WebSocket:** WebSocket protocol allows full-duplex communications over a single socket connection for sending messages between client and server. WebSocket is based on TCP and allows streams of messages to be sent back and forth between the client and server while keeping the TCP connection open.
- **MQTT:** Message Queue Telemetry Transport, is a lightweight messaging protocol based on the publish-subscribe model. MQTT uses a client-server architecture where the client connects to the server and publishes messages to topics on the server. The broker forwards the messages to the clients subscribed to topics. MQTT is well suited for constrained environments where the devices have limited processing and memory resources and the network bandwidth is low.
- **XMPP:** Extensible Messaging and Presence Protocol is a protocol for real time communication and streaming XML data between network entities. XMPP powers a wide range of applications including messaging, presence, data syndication, gaming, multi-party chat and voice/video calls. XMPP allows sending small chunks of XML data from one network entity to another in near real-time. XMPP supports both client to server and server-to-server communication paths.
- **DDS:** Data Distribution Service is a data centric middleware standard for device to device or machine to machine communication. DDS uses a publish-subscribe model where publishers create topics to which subscribers can subscribe. Publisher is an object responsible for data distribution and the subscriber is responsible for receiving published data.
- **AMQP:** Advanced Message Queuing Protocol (AMQP) is an open application layer protocol for business messaging. AMQP supports both point to point and publisher/subscriber models. AMQP brokers receive messages from publishers and route them over connections to consumers.

7. Logical design of IoT. *
    - Logical design of an IoT system refers to an abstract representation of the entities and processes without going into the low-level specifications of the implementation.
8. IoT functional blocks. *

- **Device**: An IoT system comprises devices that provide sensing, actuation, monitoring and control functions.
- **Communication**: The communication block handles the communication for the IoT system.
- **Services**: An IoT system uses various types of IoT services such as services for device monitoring, device control services, data publishing services and services for device discovery.
- **Management**: Management functional block provides various functions to govern the IoT system.
- **Security**: Security functional block secures the IoT system by providing functions such as authentication, authorization, message and content integrity, and data security.
- **Application**: IoT applications provide an interface that the users can use to control and monitor various aspects of the IoT system. Applications also allow users to view the system status and view or analyze the processed data.

9. IoT communication models. *
    - **Request-response**: Request-response is a communication model in which the client sends requests to the server and the server responds to the requests. When the server receives a request, it decides how to respond, fetches the data, retrieves resource representations, prepares the response, and then sends the response to the client.
    - **Publish-subscribe**: Publish-subscribe is a communication model that involves publishers, brokers and consumers. Publishers are the source of data. Publishers send the data to the topics which are managed by the broker. Publishers are not aware of the consumers. Consumers subscribe to the topics which are managed by the broker. When the broker receives data for a topic from the publisher, it sends the data to all the subscribed consumers.

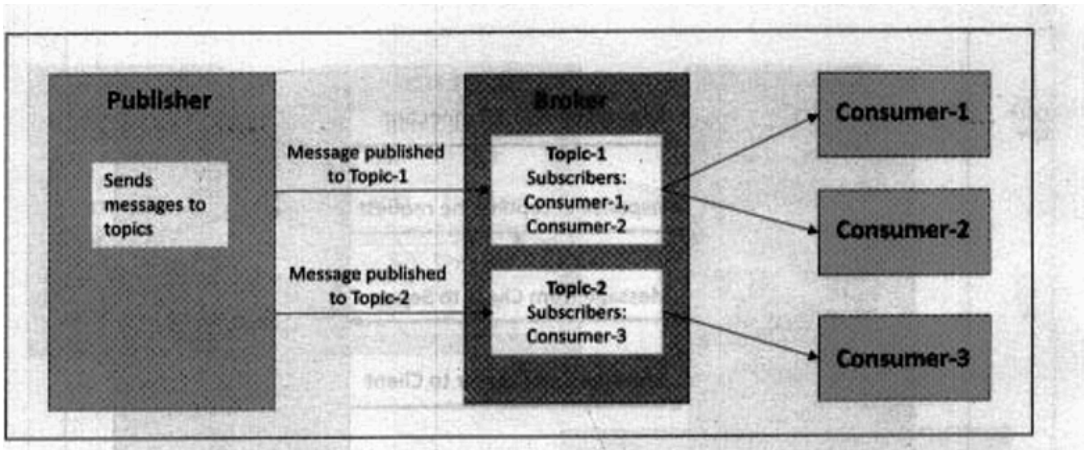

Figure 1.8: Publish-Subscribe communication model

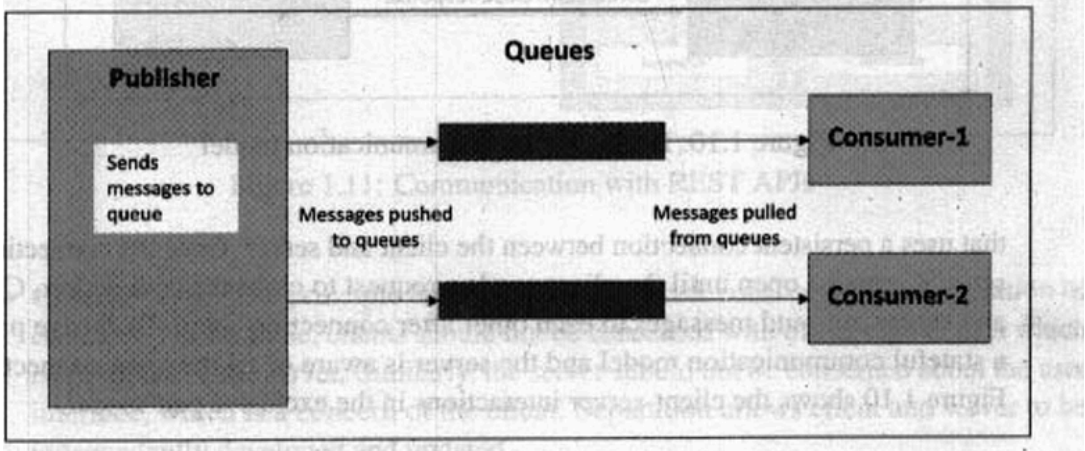

Figure 1.9: Push-Pull communication model

-
- **Push-pull**: Push-pull is a communication model in which the data producers push the data to queues and the consumers pull the data from the queues. Producers do not need to be aware of the consumers. Queues help in decoupling the messaging between the producers and consumers. Queues also act as a buffer which helps in situations when there is a mismatch between the rate at which the producers push data and the rate at which the consumers pull data.
- **Exclusive pair**: It is a bi-directional, fully duplex communication model that uses a persistent connection between the client and server. Once the connection is set up it remains open until the client sends a request to close the connection. Client and server can send messages to each other after connection setup. Exclusive pair is a stateful communication model and the server is aware of all the open connections.

10. IoT communication APIs. *
    - **REST-based Communication APIs**: Representational State Transfer is a set of architectural principles by which you can design web services and web APIs that focus on a system's resources and how resource states are addressed and transferred. REST APIs follow the request-response communication model. The

REST architectural constraints apply to the components, connectors and data elements, within a distributed hypermedia system.

- **WebSocket based communication APIs**: WebSocket APIs allow bi-directional, full duplex communication between clients and servers. WebSocket APIs follow the exclusive pair communication model. Unlike request-response APIs such as REST, the WebSocket APIs allow full duplex communication and do not require a new connection to be set up for each message to be sent. WebSocket communication begins with a connection set up request sent by the client to the server. This request is sent over HTTP and the server interprets it as an upgrade request.

11. IoT enabling technologies.*
    - **WSN**: A WSN comprises distributed devices with sensors which are used to monitor the environmental and physical conditions. A WSN consists of a number of end-nodes and routers and a coordinator. End nodes have several sensors attached to them. End nodes can also act as routers. Routers are responsible for routing the data packets from end-nodes to the coordinator. The coordinator collects the data from all the nodes. Coordinator also acts as a gateway that connects the WSN to the Internet.
    - **Cloud computing**: Cloud computing is a transformative computing paradigm that involves delivering applications and services over the Internet. Cloud computing involves provisioning of computing, networking and storage resources on demand and providing these resources. The process of provisioning resources is automated. Cloud computing resources can be accessed over the network using standard access mechanisms that provide platform-independent access through the use of heterogeneous client platforms such as workstations, laptops, tablets and smart-phones.
    - **Big data analytics**: Big data is defined as collections of data sets whose volume, velocity or variety is so large that it is difficult to store, manage, process and analyze the data using traditional databases and data processing tools. Big data analytics involves several steps starting from data cleansing, data munging, data processing and visualization. The underlying characteristics of big data include:
        (a) Volume: though there is no fixed threshold for the volume of data to be considered as big data, however typically the term big data is used for massive scale data that is difficult to store, manage and process using traditional databases and data processing architectures.
        (b) Velocity: Velocity of data refers to how fast the data is generated and how frequently it varies.
        (c) Variety: Variety refers to the forms of the data. Big data comes in different forms such as structured or unstructured data, including text data, image, audio, video and sensor data.
12. IoT system *

    An IoT system comprises of the following components:

    - **Device**: An IoT device allows identification, remote sensing, actuating and remote monitoring capabilities.

- **Resources**: Resources are software components on the IoT device for accessing, processing, and storing sensor information, or controlling actuators connected to the device. Resources also include the software components that enable network access for the device.
- **Controller service**: Controller service is a native service that runs on the device and interacts with the web services. Controller service sends data from the device to the web service and receives commands from the application for controlling the device.
- **Database**: Database can either be local or in the cloud and stores the data generated by the IoT device.
- **Web service**: Web services serve as a link between the IoT device, application, database and analysis components.

13. Differentiate between REST and WebSocket. *

Parameters	**REST**	**WebSocket**
Stateless/Stateful	Stateless	Stateful
Direction	Uni-directional	Bi-directional
Communication model	Request-response	Full duplex
TCP	Each request involves a new TCP.	Involves a single TCP connection.
Header overhead	Overhead	Not overhead
Scalability	Easy	Cumbersome

14. IoT level 1. (Explanation, diagram and example)
15. IoT level 2. (Explanation, diagram and example)
16. IoT level 3. (Explanation, diagram and example)
17. IoT level 4. (Explanation, diagram and example)
18. IoT level 5. (Explanation, diagram and example)
19. IoT level 6. (Explanation, diagram and example)
20. In the context of IoT platform design methodology, explain the following:
    (a) Purpose and requirement specification.
    (b) Process specification.
    (c) Domain model specification.
    (d) Information model specification.
    (e) Service specification.
    (f) IoT level specification.
    (g) Functional view specification.
    (h) Operational view specification.
    (i) Device and component integration.
    (j) Application development.