Basic Nodal Capabilities

- ❑ Remote device generally needs to have a basic protocol stack
- ❑ Basic protocol supports as minimum local connectivity and networking connectivity.
- ❑ In addition some higher layer application support protocols are generally needed

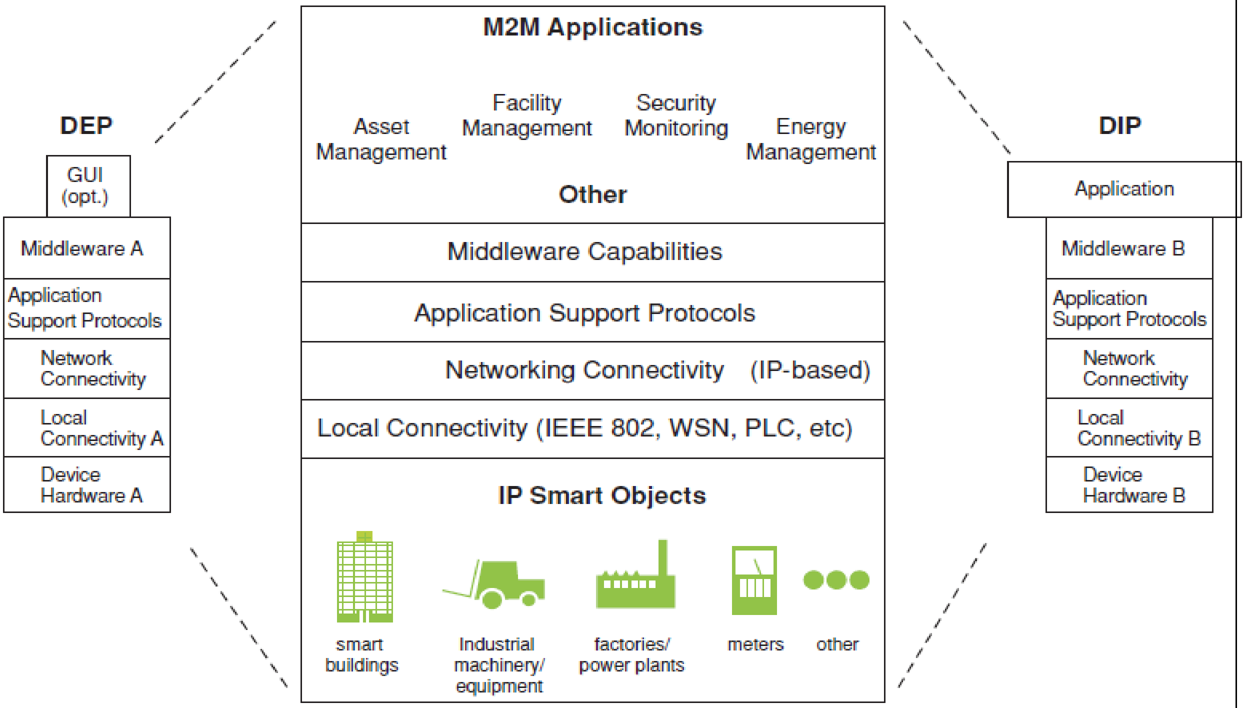
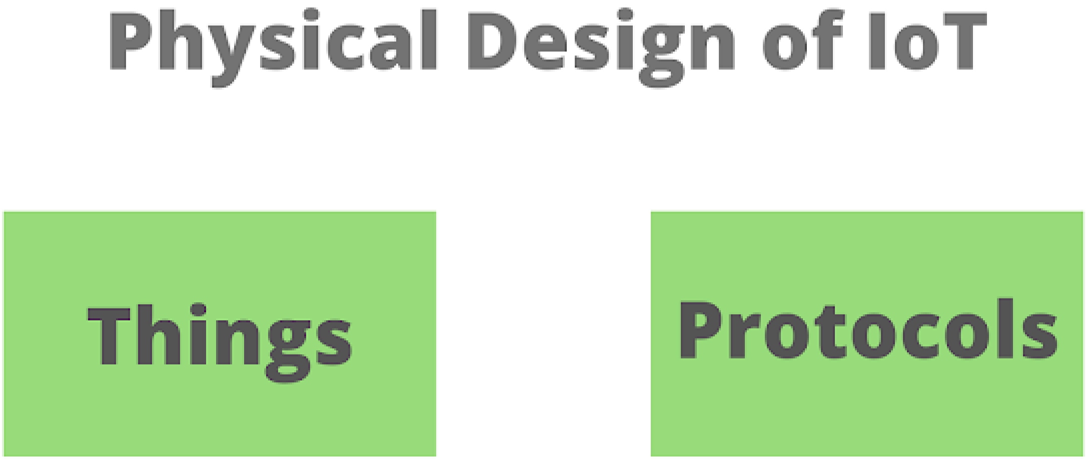


FIGURE 2.9 Protocol stack, general view.

Physical Design of IoT

The **physical design** of an **IoT** system is referred to as the **Things/Devices** and protocols that are used to build an IoT system. all these things/Devices are called Node Devices and every device has a unique identity that performs remote sensing, actuating and monitoring work. and the protocols that are used to establish communication between the Node devices and servers over the internet.

Physical Design of IoT



Physical Design of IoT

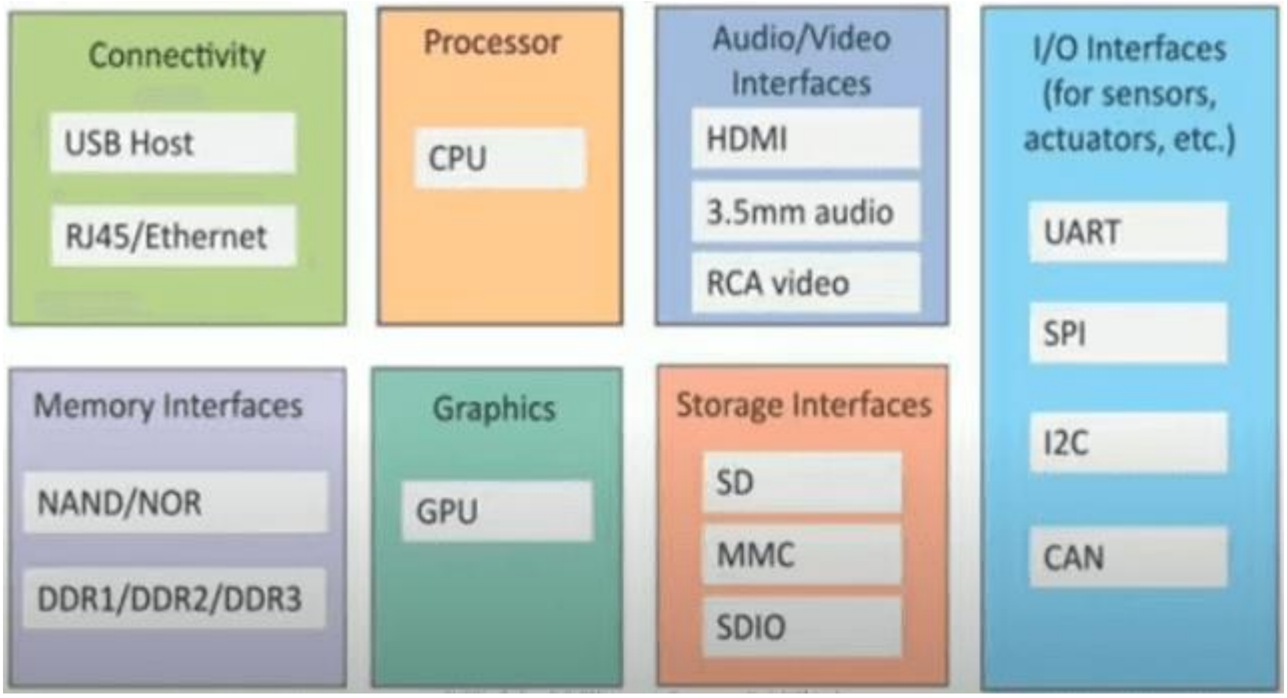
Things

Protocols

Things/Devices

Things/Devices are used to build a connection, process data, provide interfaces, provide storage, and provide graphics interfaces in an IoT system. all these generate data in a form that can be analyzed by an analytical system and program to perform operations and used to improve the system.

for example temperature sensor that is used to analyze the temperature generates the data from a location and is then determined by algorithms.



Connectivity

Devices like USB hosts and ETHERNET are used for connectivity between the devices and the server.

Processor

A processor like a CPU and other units are used to process the data. these data are further used to improve the decision quality of an IoT system.

Audio/Video Interfaces

An interface like HDMI and RCA devices is used to record audio and videos in a system.

Input/Output interface

To give input and output signals to sensors, and actuators we use things like UART, SPI, CAN, etc.

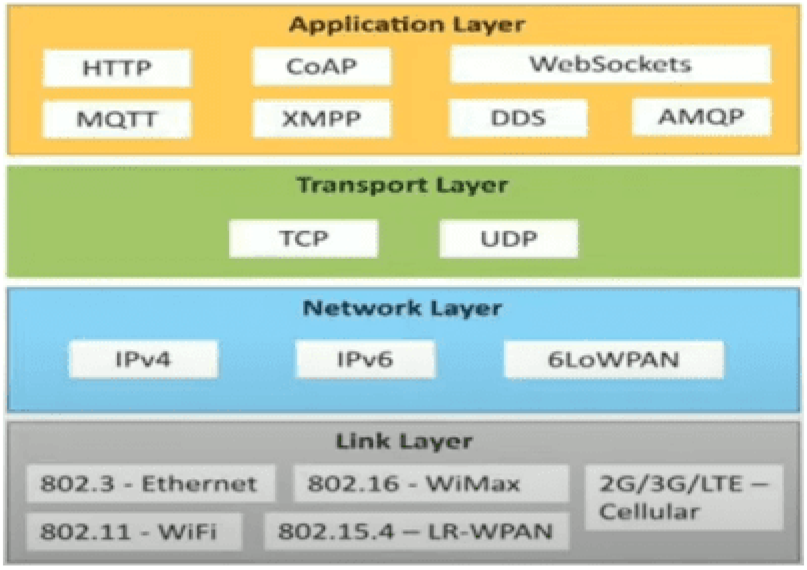
Storage Interfaces

Things like SD, MMC, and SDIO are used to store the data generated from an IoT device.

Other things like DDR and GPU are used to control the activity of an IoT system.

IoT Protocols

These protocols are used to establish communication between a node device and a server over the internet. it helps to send commands to an IoT device and receive data from an IoT device over the internet. we use different types of protocols that are present on both the server and client-side and these protocols are managed by network layers like application, transport, network, and link layer.



Application Layer protocol

In this layer, protocols define how the data can be sent over the network with the lower layer protocols using the application interface. these protocols include HTTP, WebSocket, XMPP, MQTT, DDS, and AMQP protocols.

HTTP

Hypertext transfer protocol is a protocol that presents in an application layer for transmitting media documents. it is used to communicate between web browsers and servers. it makes a request to a server and then waits till it receives a response and in between the request server does not keep any data between two requests.

WebSocket

This protocol enables two-way communication between a client and a host that can be run on an untrusted code in a controlled environment. this protocol is commonly used by web browsers.

MQTT

It is a machine-to-machine connectivity protocol that was designed as a publish/subscribe messaging transport. and it is used for remote locations where a small code footprint is required.

Transport Layer

This layer is used to control the flow of data segments and handle the error control. also, these layer protocols provide end-to-end message transfer capability independent of the underlying network.

TCP

The transmission control protocol is a protocol that defines how to establish and maintain a network that can exchange data in a proper manner using the internet protocol.

UDP

a user datagram protocol is a part of an internet protocol called the connectionless protocol. this protocol is not required to establish the connection to transfer data.

Network Layer

This layer is used to send datagrams from the source network to the destination network. we use IPv4 and IPv6 protocols as host identification that transfers data in packets.

IPv4

This is a protocol address that is a unique and numerical label assigned to each device connected to the network. an IP address performs two main functions host and location addressing. IPv4 is an IP address that is 32-bit long.

IPv6

It is a successor of IPv4 that uses 128 bits for an IP address. it is developed by the IETF task force to deal with long-anticipated problems.

Link Layer

Link-layer protocols are used to send data over the network's physical layer. it also determines how the packets are coded and signaled by the devices.

Ethernet

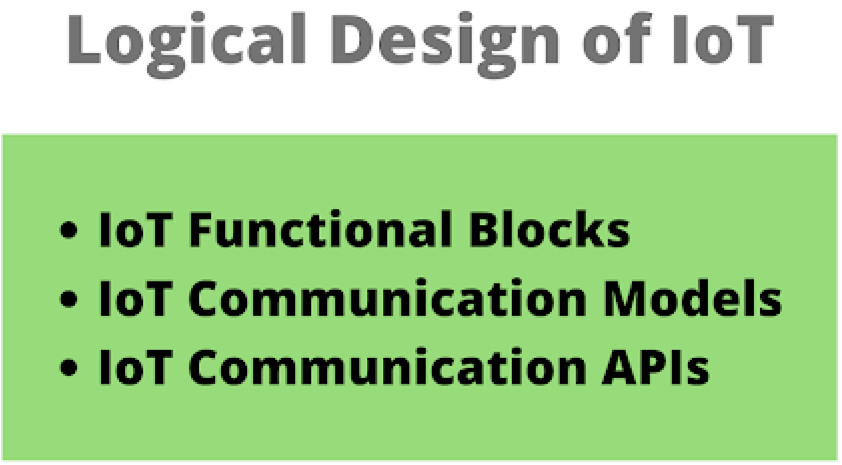
It is a set of technologies and protocols that are used primarily in LANs. it defines the physical layer and the medium access control for wired ethernet networks.

WiFi

It is a set of LAN protocols and specifies the set of media access control and physical layer protocols for implementing wireless local area networks.

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.

But before you learn about the logical design of IoT systems you need to know a little bit about the physical design of IoT.



Logical Design of IoT

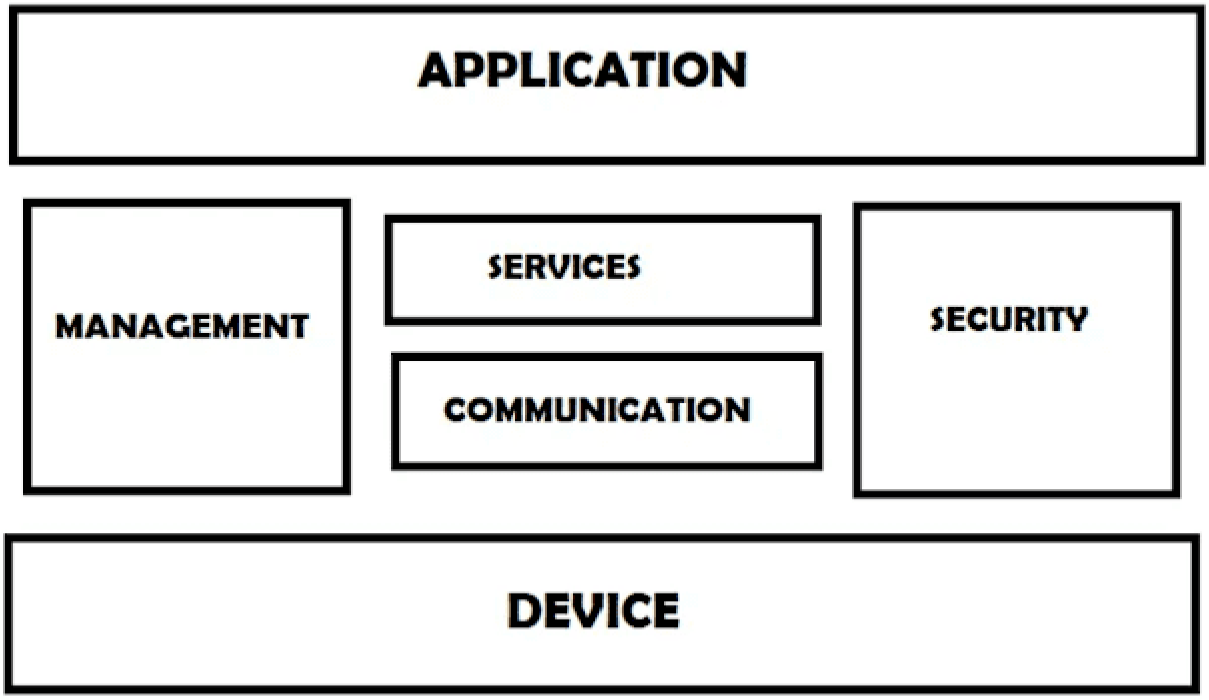
- **IoT Functional Blocks**
- **IoT Communication Models**
- **IoT Communication APIs**

Logical Design of Internet of Things(IoT)

1. IoT Functional Blocks
2. IoT Communication Models
3. IoT Communication APIs

IoT Functional blocks

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



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.

Application

It is an interface that provides a control system that use by users to view the status and analyze of system.

Management

This functional block provides various functions that are used to manage an IoT system.

Services

This functional block provides some services like monitoring and controlling a device and publishing and deleting the data and restoring the system.

Communication

This block handles the communication between the client and the cloud-based server and sends/receives the data using protocols.

Security

This block is used to secure an IoT system using some functions like authorization, data security, authentication, 2-step verification, etc.

Device

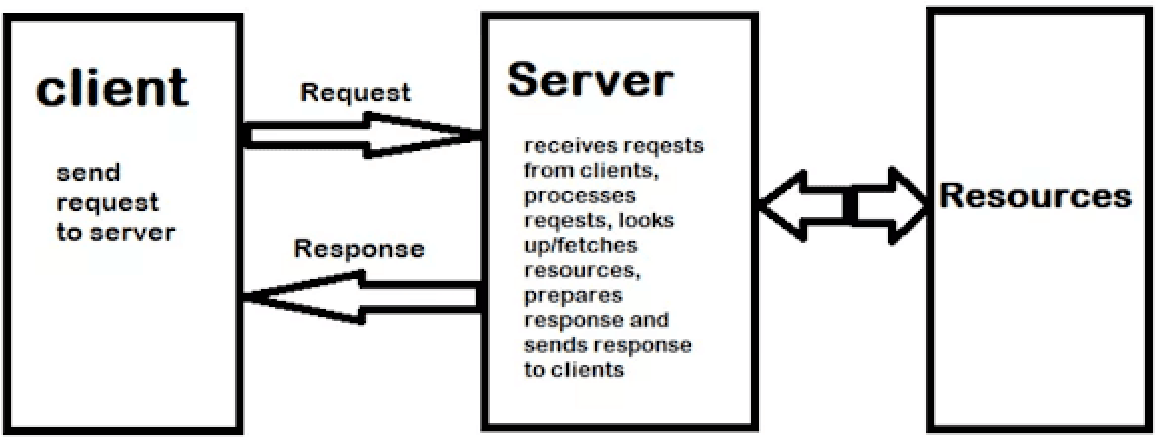
These devices are used to provide sensing and monitoring control functions that collect data from the outer environment.

IoT Communication Models

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


client
send request to server
Request
Response
Server
receives reqests from clients, processes reqests, looks up/fetches resources, prepares response and sends response to clients
Resources

Request-Response Communication Model

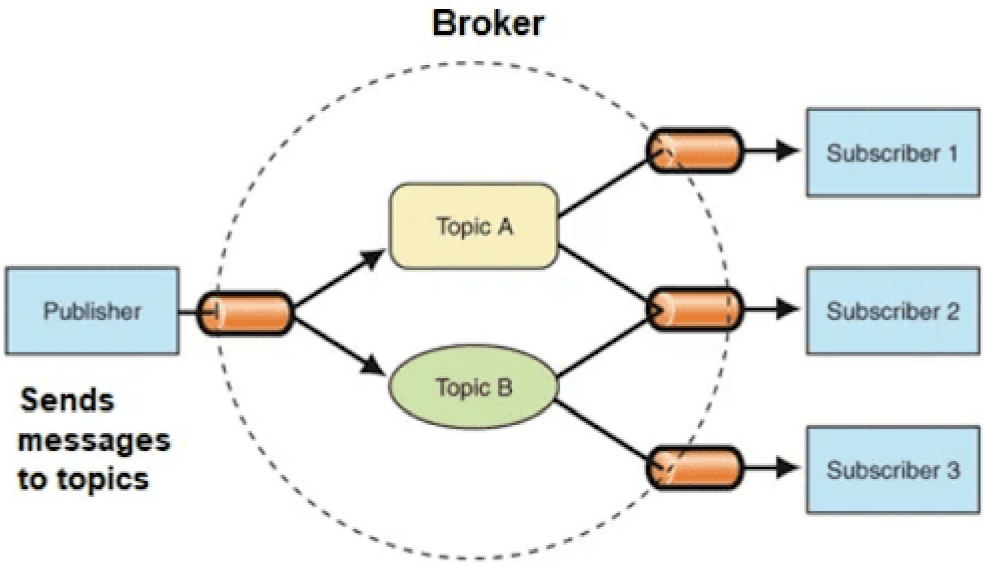
In simple terms, we can say that in the request-response model server send the response of equivalent to the request of the client. in this model, HTTP works as a request-response protocol between a client and server.

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).

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 is managed by the broker and when the broker receives data from the publisher it sends the data to all the subscribed consumers.

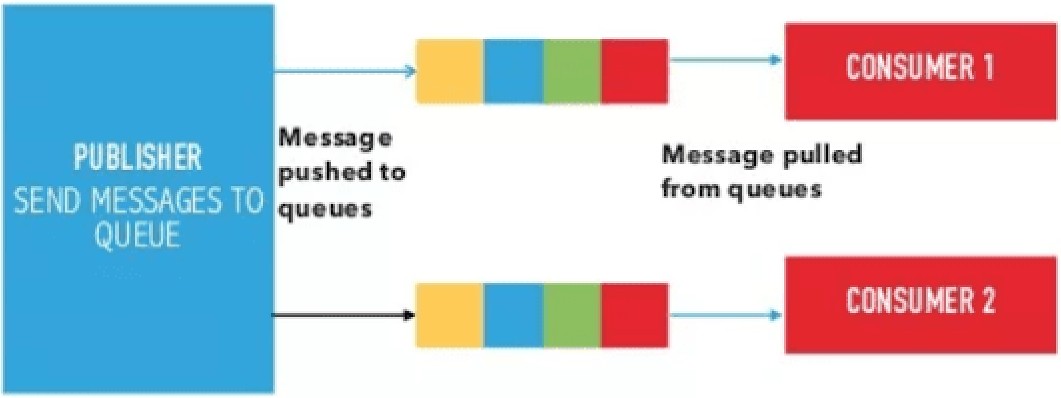


Example

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

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.



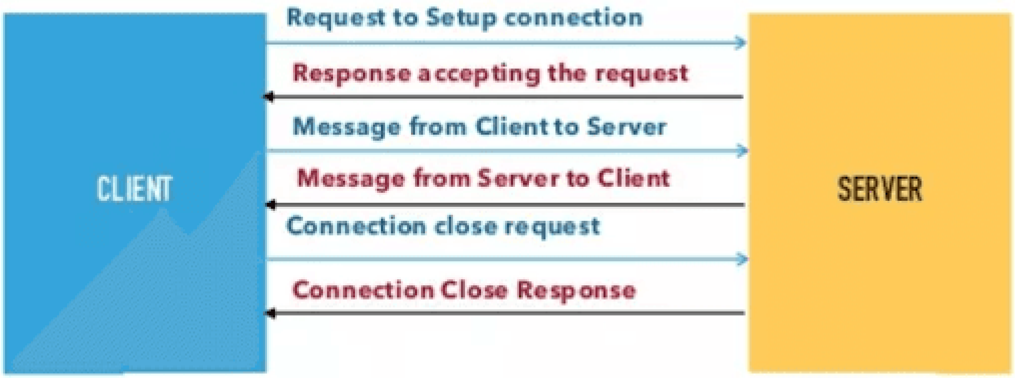
PUSH PULL MODEL

Example

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

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 remain open until the client sends a close connection request to the server.


Request to Setup connection
Response accepting the request
Message from Client to Server
Message from Server to Client
Connection close request
Connection Close Response
CLIENT
SERVER

EXCLUSIVE PAIR COMMUNICATION MODEL

IoT communication APIs

These APIs like REST and WebSocket are used to communicate between the server and system in IoT.

REST-based communication APIs

Representational state transfer(REST) API uses a set of architectural principles that used to design web services. these APIs focus on the systems' resources that how resource states are transferred using the request-response communication model. this API uses some architectural constraints.

Client-server

Here the client is not aware of the storage of data because it is concerned about the server and similarly the server should not be concerned about the user interface because it is a concern of the client. and this separation is needed for independent development and updating of server and client. no matter how the client is using the response of the server and no matter how the server is using the request of the client.

Stateless

It means each request from the client to the server must contain all the necessary information to understand by the server. because if the server can't understand the request of the client then it can't fetch the request data in a proper manner.

Cacheable

In response, if the cache constraints are given then a client can reuse that response in a later request. it improves the efficiency and scalability of the system without loading the extra data.

A RESTful web APIs is implemented using HTTP and REST principles.

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 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 after the setup of a connection server and the 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.