TCP/IP FLOWCHART

Ⓐ Basic Client-Server Model

Server		Client

```
Server                          Client

socket()                        socket()
   ↓                               |
bind()                             |
   ↓                               |
listen()                           |
   ↓         Synchronisation       ↓
accept() ← — Point — → connect()
   ↓                               ↓
recv() ←————————————→  send()
   ↓                               ↓
send() ————————————→  recv()
   ↓                               ↓
close()                         close()
```
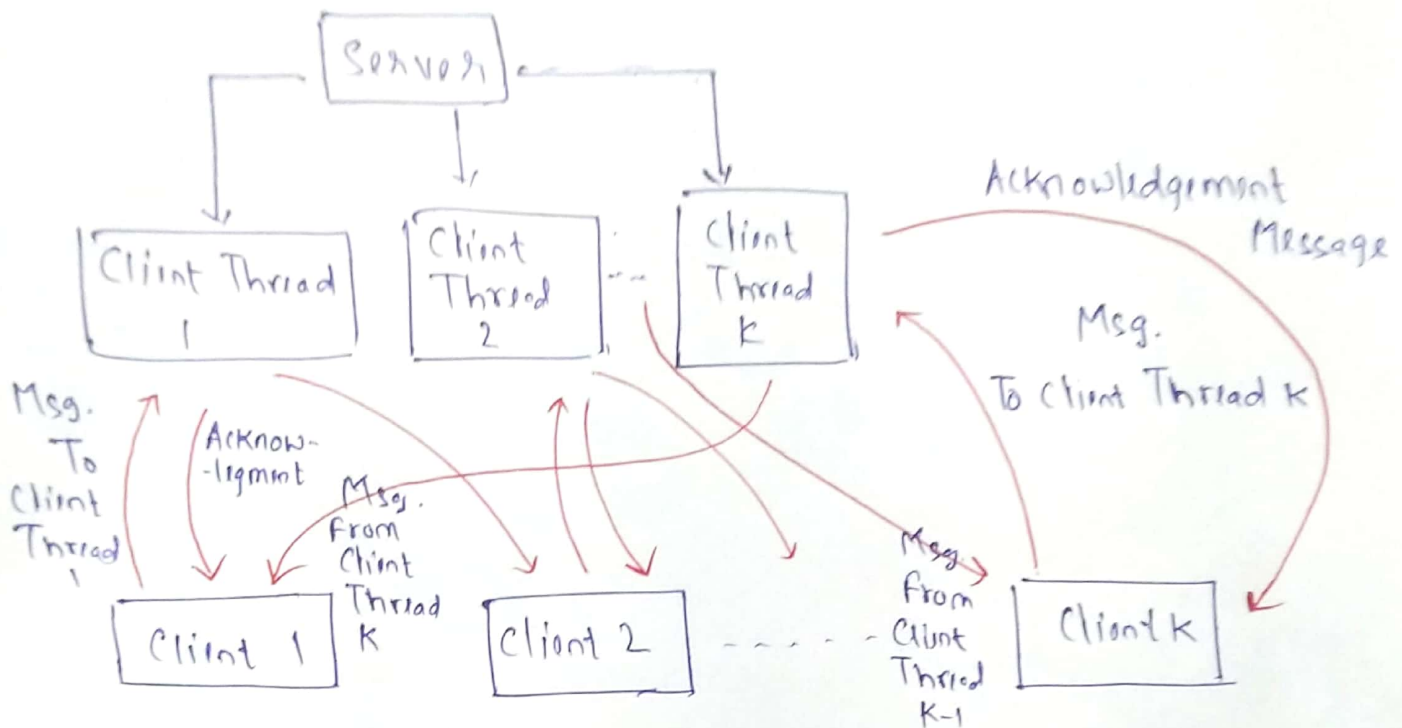
Ⓑ Handing Multiple Clients by using Threading

```
                 ┌──────────────────┐  ───────→  ┌──────────┐
                 │ Client Thread 1  │ ←───────   │ Client 1 │
                 │  in server       │            └──────────┘
                 └──────────────────┘

                 ┌──────────────────┐  ───────→  ┌──────────┐
                 │ Client Thread 2  │ ←───────   │ Client 2 │
┌─────────┐      │  in server       │            └──────────┘
│ SERVER  │ ─ ─ └──────────────────┘        Information
└─────────┘              ┆                    exchange
                         ┆
                 ┌──────────────────┐  ───────→  ┌──────────┐
                 │ Client Thread K  │ ←───────   │ Client   │
                 │  in server       │            │   K      │
                 └──────────────────┘            └──────────┘
```
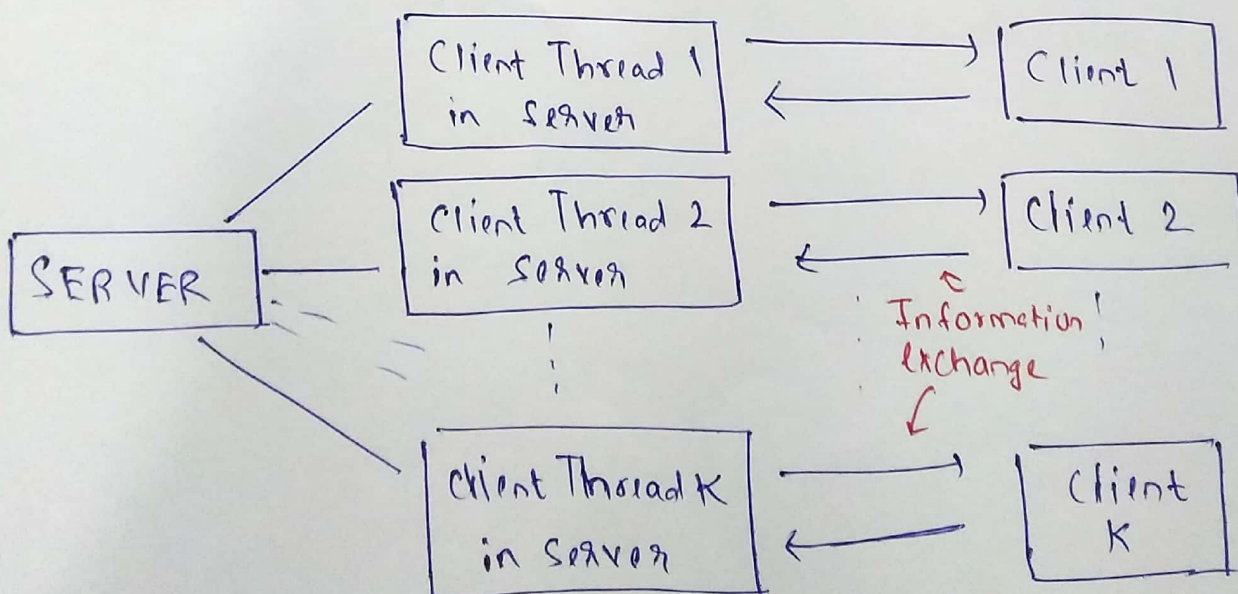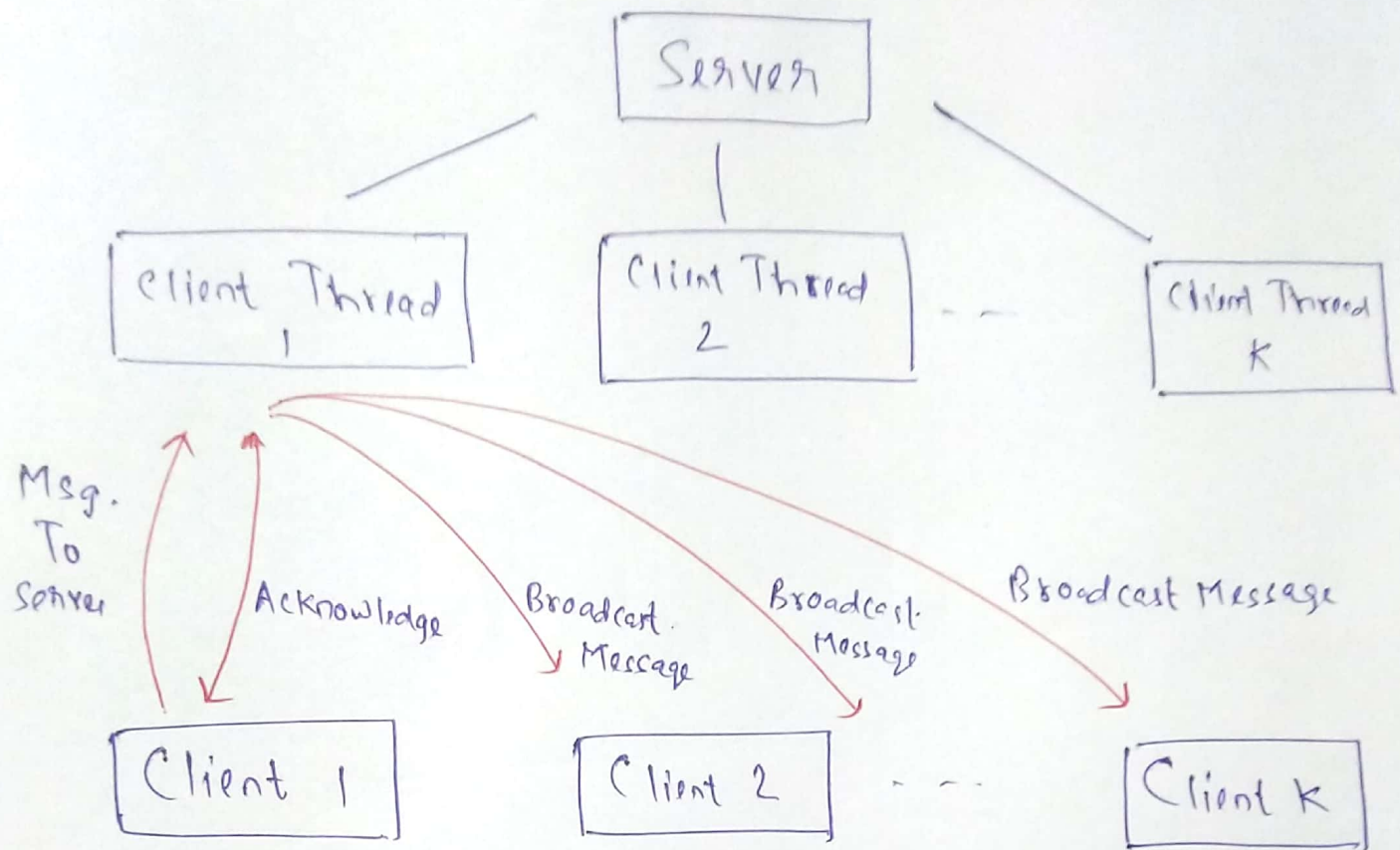
Ⓒ One to One Model

• Message from i^{th} client goes to $(i \% K + 1)$ client

• Client $i \rightarrow$ Client Thread $i \rightarrow$ Client $(i \% K + 1)^{th}$ client

Acknowledgement Message

Ⓓ Broadcast Model

• Message from i^{th} Client goes to all the other Active Clients (i.e K-1 clients).

• i^{th} Client receives Acknowledgement from Client Thread i in Server on sending a message.
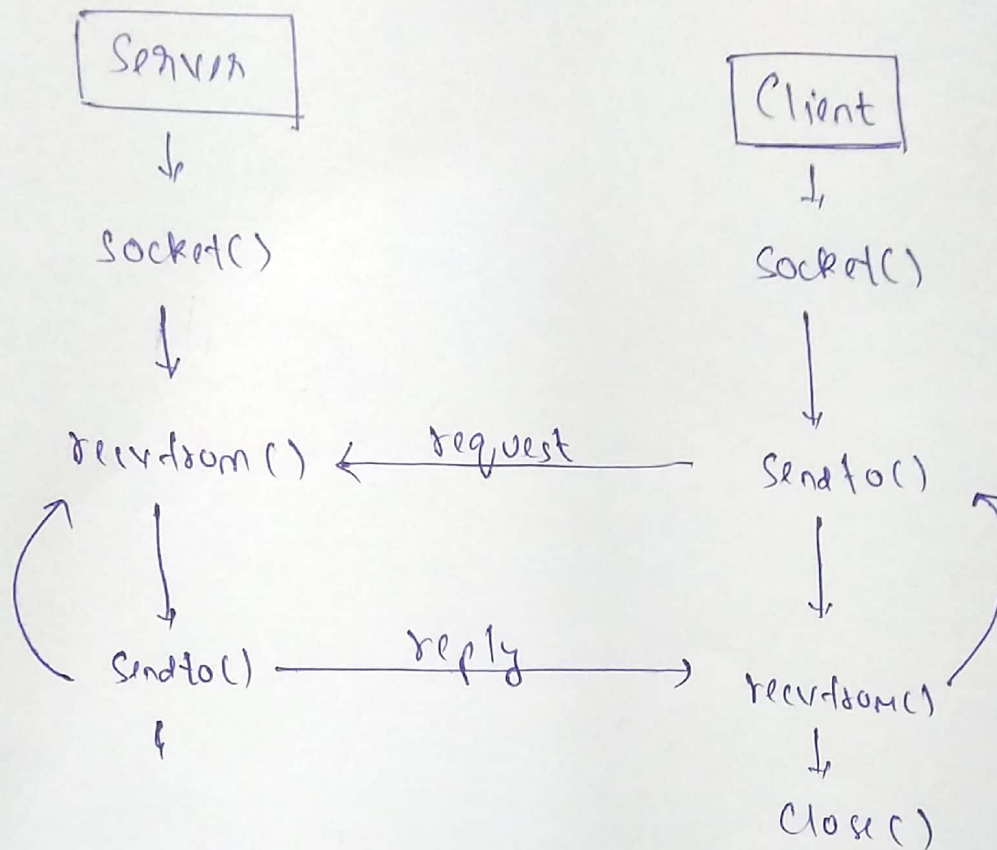
The above Model depicts a Case only where Client 1 sends a message in Broadcast Mode. If other clients also send Message then, a similar Analogy will be continued.

- The complete model (message from each client) hasn't been shown to improve clarity of the block diagram.
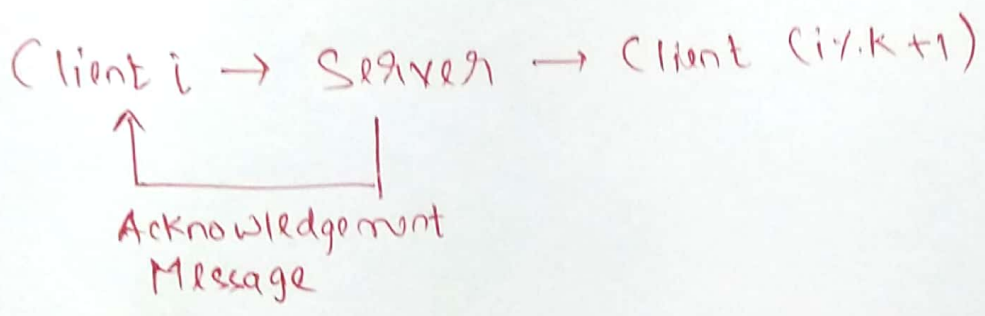
UDP FLOWCHART

Ⓐ

Server		Client

```
Server                              Client
  ↓                                   ↓
socket()                           socket()
  ↓                                   ↓
recvfrom() ←——— request ———        sendto()
  ↑  ↓                                ↓
  |  ↓                                ↓
sendto() ——————— reply ————→       recvfrom()
  ↓                                   ↓
                                    close()
```
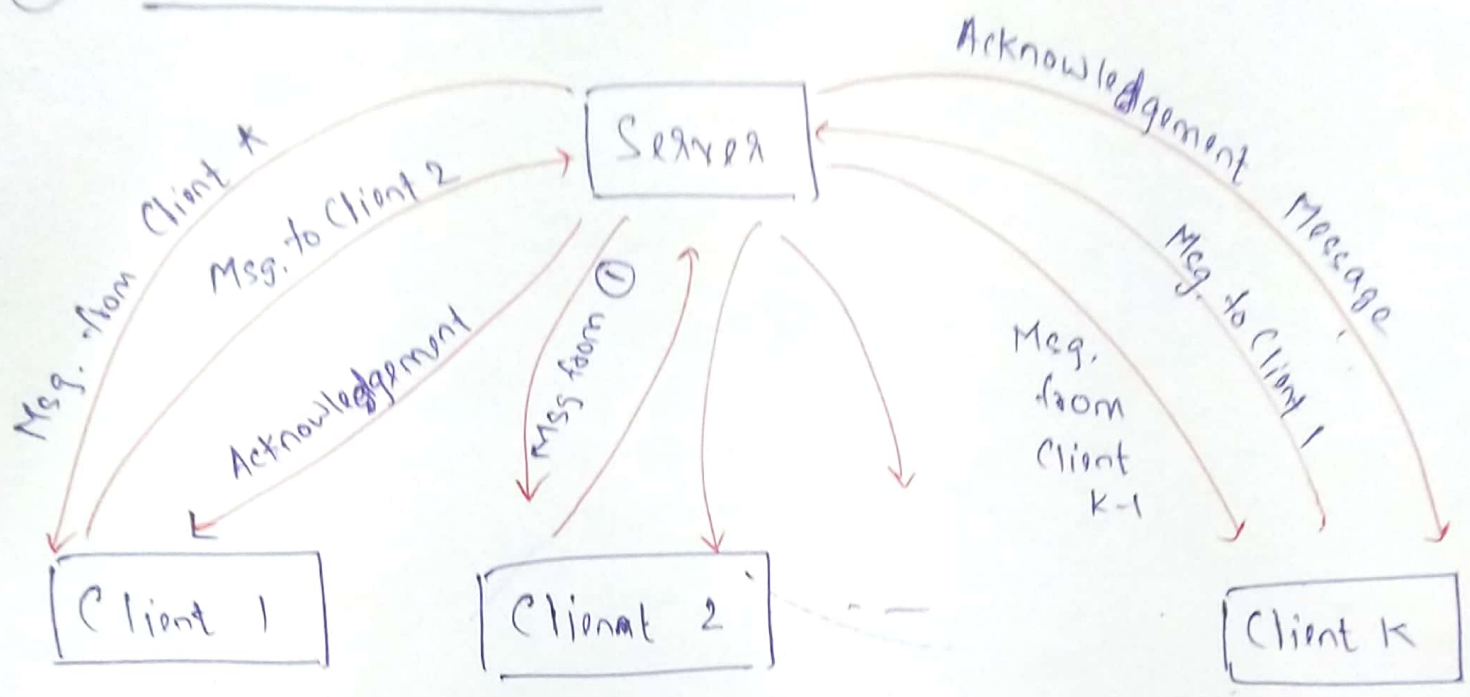
Basic Client-Server Model

- Information of Client is stored in an Array in the Server through which we can implement One-One and Broadcast Modes.
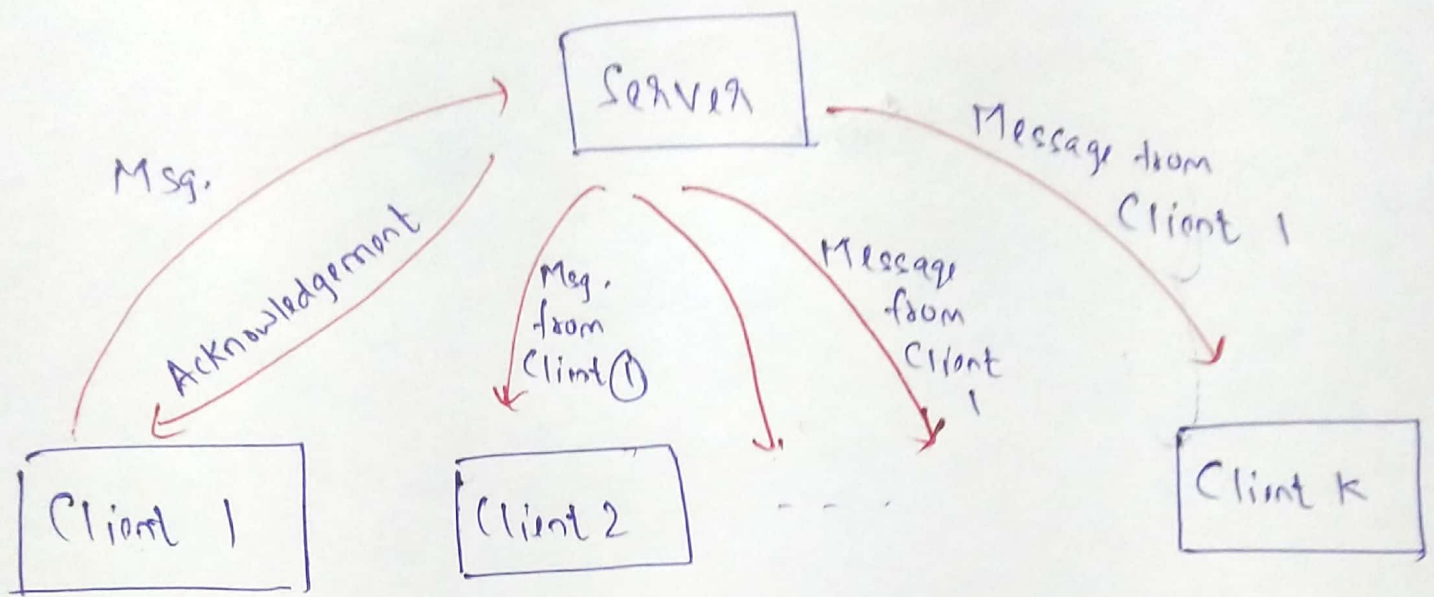
Ⓑ One to One Model

Client i → Server → Client (i%K+1)

Acknowledgement Message

Ⓒ Broadcast Model

- Message from ith Client goes to all other K-1 Active Clients

- ith Client Receives acknowledgement from ~~Client~~ ~~thread~~ Server on sending a Message.

- Assume only Client 1 sends a Message

- Similarly if any Client i, sends a message, the model Can be drawn

- Only 1 Client sending a message has been depicted to improve clarity of Diagram.