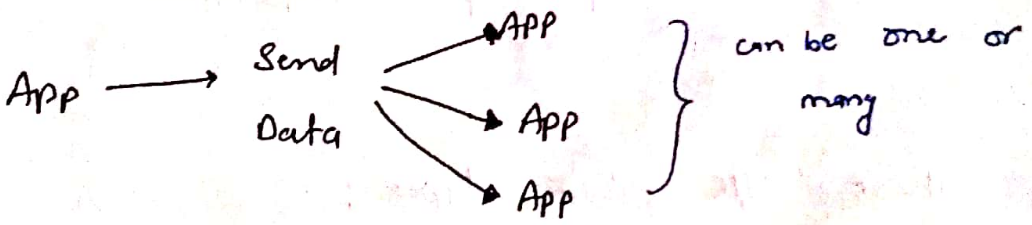
19-8-19

-:- Distributed Messaging :-

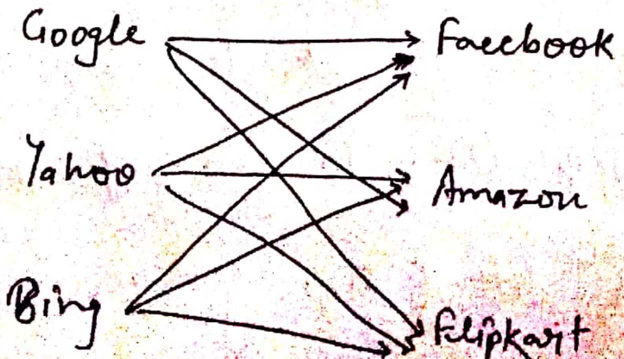
It's a process of "sending message from source application to one or more destination application", It's not a concept of request and response.

→ This is implemented using MQs (Message Queues)

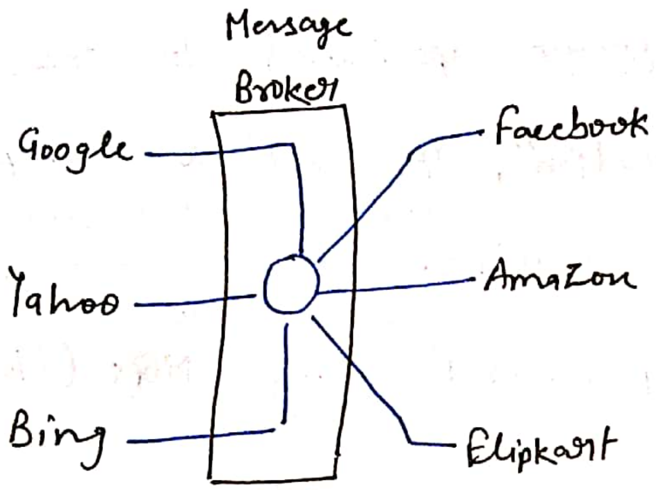


→ App Sending data to multiple applications.

→ Consider below Example, a design given without MQs looks like mesh design.



→ This is simplified by MQ which is given below with one mediator also called as Message Broker.



→ These are two types :

A. Basic Message Queuing Protocal (BMQP)

B. Advanced Message Queuing Protocal (AMQP)

A. Basic Message Queuing Protocal (BMQP) :- It is also called as language specific MQ. In case of java it is called as JMS (Java Message Service).

→ It is language dependent.

ex Broker Software :- Apache ActiveMQ, Attrims, RabbitMQ etc

-!- Spring Java Message Service :-

It is used to implement Basic messaging process b/w applications, to send or recieve data.

→ It uses one Broker software that is called as MOM (Message Oriented Middleware).

→ MOM contains one special memory (container) called as Destination that holds messages.

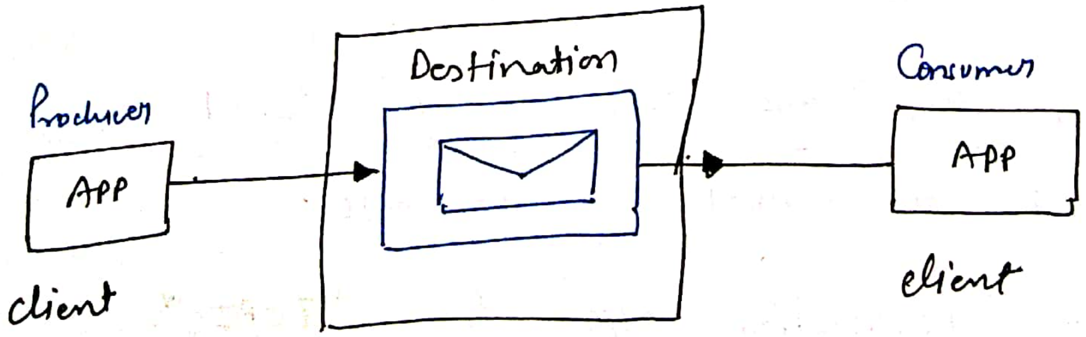
→ Every Application connected to MOM is called as Client.

→ Producer (Client) : Sends Message to MOM

Consumer (Client) : Read Message from MOM.

MOM

(Menge Oriented Middleware)



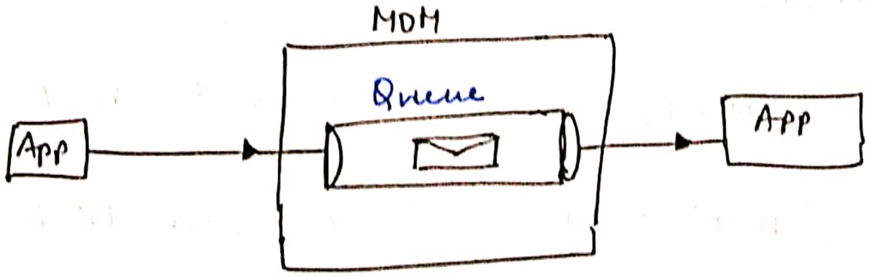
Types of Communications in JMS :-

There are two types of Communications supported by JMS.

1. **P2P (Peer-to-Peer) Communication** :- One message is sent to one consumer app only.

→ In this case Destination type is called as "Queue". [QUEUE]

P2P Communication: Destination : Queue

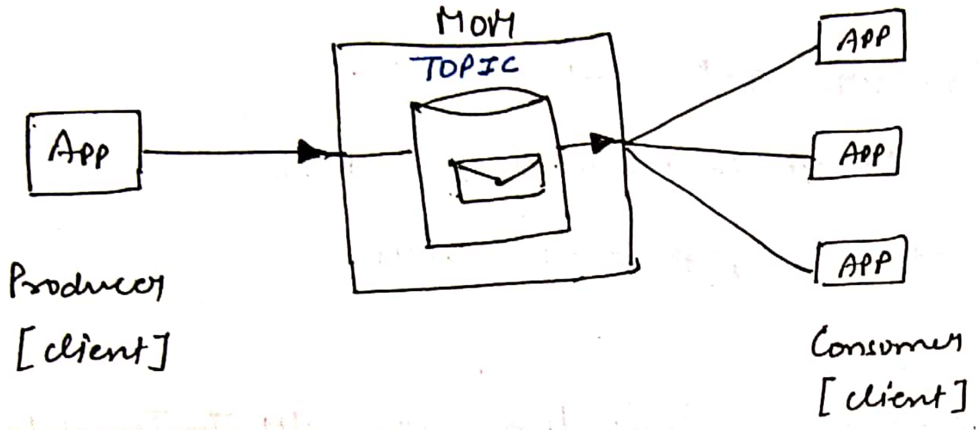


2. **Pub/Sub (Publish-and-subscribe)** :-

→ In this case one message cloned copy send to multiple consumers.

→ Here Destination type is "TOPIC".

Pub|Sub Communication! Destination: Topic



-:- Steps to implement JMS using ActiveMQ :-

#1- In client app. inside pom.xml provide below dependency.

```
<dependency>
    <g-id>  org.sf.boot  </g-id>
     <a-id>  spring-boot-starter-activemq </a-id>
</dependency>
```

#2. Get into dependency and find version.

e.g. <activeMq.version> 5.15.9 </activemq.version>

→ Download activemq same version from

https://activemq.apache.org/component/classic/download

→ Extract > open > bin > win64 > activemq . bat

→ Goto browser and enter Url -

http:// localhost : 8161 / admin

User: admin , password: admin

#3 At producer Application use ~~JT~~ JmsTemplate (c) to send message.

#4 At consumer application enable Jms using @EnableJms Annotation also define consumer code using @JmsListener.

Note - If we are using JmsTemplate (c) then internally it follows @EnableJms.

#5. In client application inside application.properties file, provide broker URL, un, pwd, pub/sub flag

```
spring.activemq.broker-url = tcp://localhost:61616
spring.activemq.user = admin
spring.activemq.password = admin
spring.jms.pub-sub-domain = true
```