**Pub-Sub Model**

Subscriber 1

B

R

O

K

E

R

Publisher

Subscriber 2

o/p channel

i/p channel

Subscriber n

n

Subscriber 3

Suppose, you have an e-commerce website,

Where a user can trigger events like “placing an order”,  
the publisher might end following data through i/p channel:

{

Status: orderPlaced,

orderId: 876,

};

This data is received by broker, this broker adds some additional information which is needed by different subscribers like:

{

Status: orderPlaced,

orderId: 876,

buyerEmail: [abc@gmail.com](mailto:abc@gmail.com),

buyerPhoneNumber: 1234567,

};

Job of the broker is to enrich the published data, for different subscribers according to their needs.

Now this “placing Order” event might be subscribed by different services like:

* Email Service – this needs buyer email
* SMS Service – this needs buyer phone Number
* Logging Service – this logs the published event
* Order Processor Service – this requires status and orderId

All the services will not be subscribed to events else, it will consume lot of bandwidth.

Example:

* “Adding an item to cart”, this event need not be subscribed by Email or SMS Service or Order Processing Service, as we don’t email or SMS a user, whenever we add an item to cart.
* “Subscribing to newsletter”, this event will not be subscribed by SMS and order Processing service.

In this pattern, scaling of publishers and subscribers is decoupled.

We can increase number of subscribers/publishers, create multiple channels.

Factors to consider in Pub-Sub Model:

* **Message Ordering and consumption:**

In Pub-Sub pattern, order of messages being consumed is not guaranteed like in one to one communication.

If you want ordering, you can attach priority via priorityQueue.

Otherwise, take care of ordering in application.

* **Poison Message:**

You might be communicating with 3rd party software and it might send malformed messages, in that case, you must take care of security or discarding such messages.

* **Duplicate Message:**

Must take care of duplicate messages, if order placing event gets triggered twice, it must not happen that user’s money gets debited twice for one item.