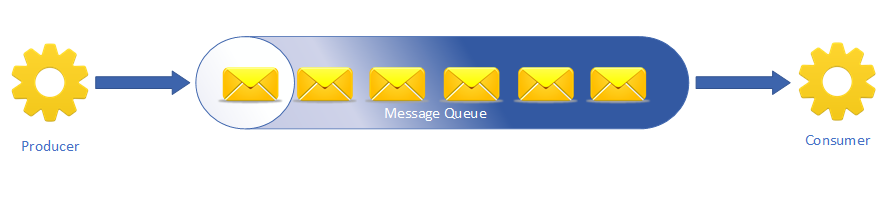
**Messaging queues**

Messaging queues are fundamental components in software architecture that enable asynchronous communication between different parts of a system.

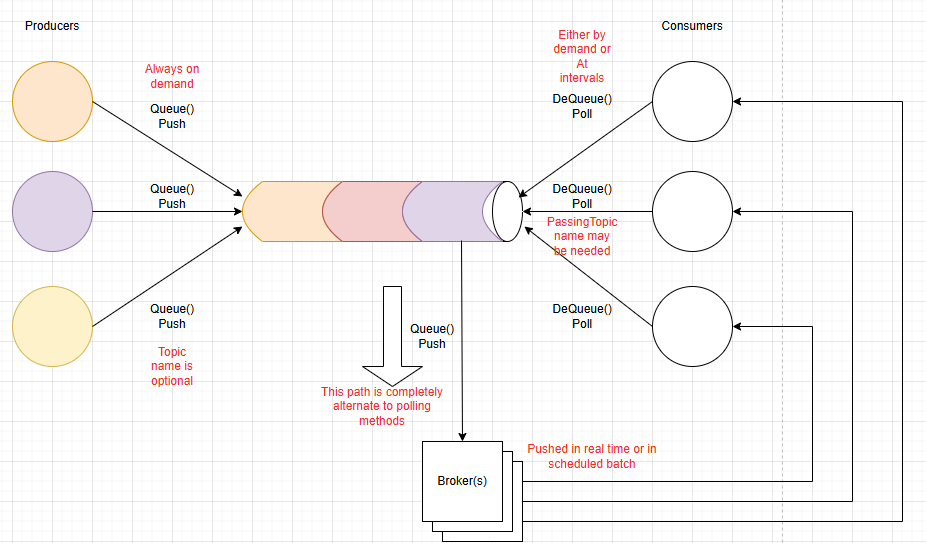


Fundamentals of Queues:

* Main need of queue is in two aspects:
  + When producing speed is different from consuming speed, it serves as a parking lot.
  + They decouple (or loosely couple) producer and consumer processes.
* Messages cannot be of infinite size. Max size of messages is configurable, so is the number of all messages in a queue.
* Usually **it is a FIFO (FIRST-IN-FIRST-OUT) and self evicting the “queue” space once Queued out.**
* **Parking the information in** the queue is always a “push” from producers.
* Either one of these are used:
  + Messages can be “pushed” out to consumers immediately or in a scheduled burst.
  + Messages can be “Polled” by consumers, at intervals or on demand.
* Retrials (for a specified period of time) are possible only in polling.

Architecture:

Note: Push/poll to consumers are mutually exclusive arrangements. Any one flavor at a time is operative. Push always is intended for a “guaranteed Delivery”. Poll focusses on “large throughput” and “flexibility”.



Scenario: Messages are of only one “Topic”

Queue() method lies at the Producer side, it simply pushes into the queue.

|  |  |  |
| --- | --- | --- |
|  | PUSH TO CONSUMER | PULL FROM CONSUMER |
| One producer + One Consumer  Or  Many producers + one consumer | Is avoided by default. Beats the very purpose of queues, and may clutter the consumer. | Dequeue() method lies at the Consumer, polls the queue either at regular intervals or as soon as processing the previous processing of message is complete or both. |
| One producer + many consumers  Or  Many producers + many consumers | Pushes to a parking lot of consumers instead of consumers themselves. These parking lots are called “Brokers”, may be implemented in “Clusters” | Dequeue() method lies at the Consumer, polls the broker. (either dedicated or shared. Persistence is for a duration configured) |

Scenario: Messages are of many “Topics”, all consumers may not need all topics

Queue() method lies at the Producer side, it pushes into the queue and along with a “Topic”.

|  |  |  |
| --- | --- | --- |
|  | PUSH TO CONSUMER | PULL FROM CONSUMER |
| One producer + One Consumer  Or  Many producers + one consumer | Is avoided by default. Beats the very purpose of queues, and may clutter the consumer. | Dequeue() method uses the Topic Name. |
| One producer + many consumers  Or  Many producers + many consumers | Brokers are involved. | Dequeue() method uses the Topic Name and polls from broker. |