1. System Architecture

1.1. Technologies

Language: Scala

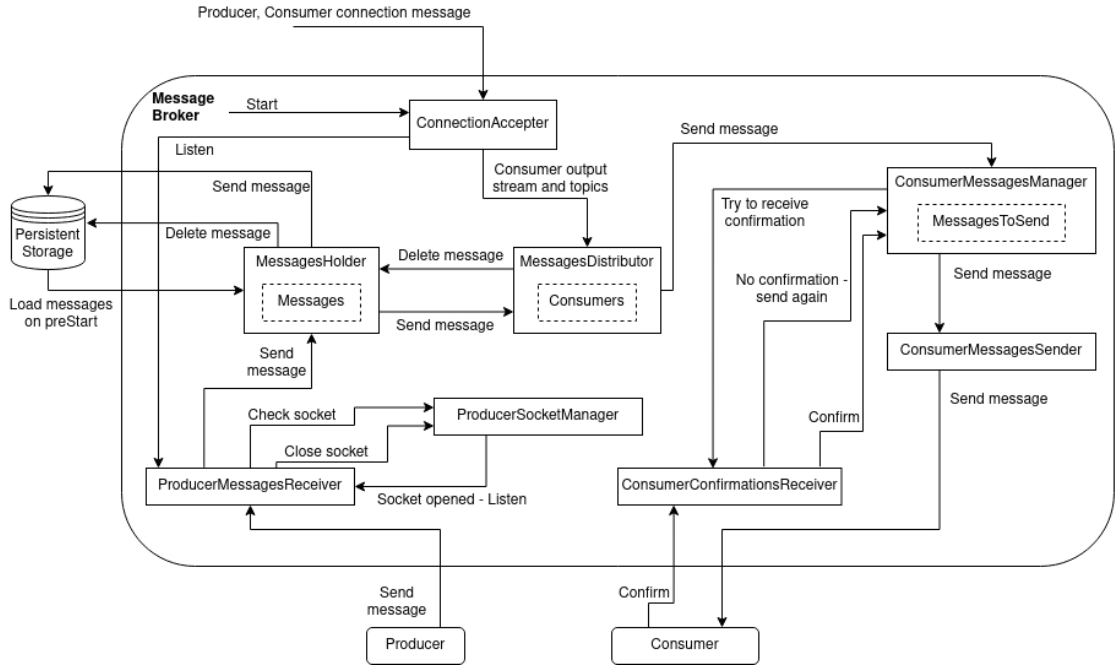
Libraries:
a) Akka - for creating and managing actors;
b) ServerSocket - to run the Message Broker as TCP server;
c) Socket - for Producers and Consumers to connect to MB;
d) ConcurrentLinkedQueue - to store the lists of connected clients and messages;
e) Serializable - to serialize classes for sending through TCP;
f) UUID - to generate IDs for the messages;
g) Source, Properties - to load configuration files;
h) LogBack - for printing logs;
i) ReactiveMongo - for MongoDB connection.

Framework: none.

1.2. General architecture

1.3. Message Broker architecture

1.3.1 Actors communication



The MessagesHolder actor has a Queue, for storing the received messages from Producers. The Queue entries are synced with the persistent storage. On preStart, the queus is populated with the messages from the DB.

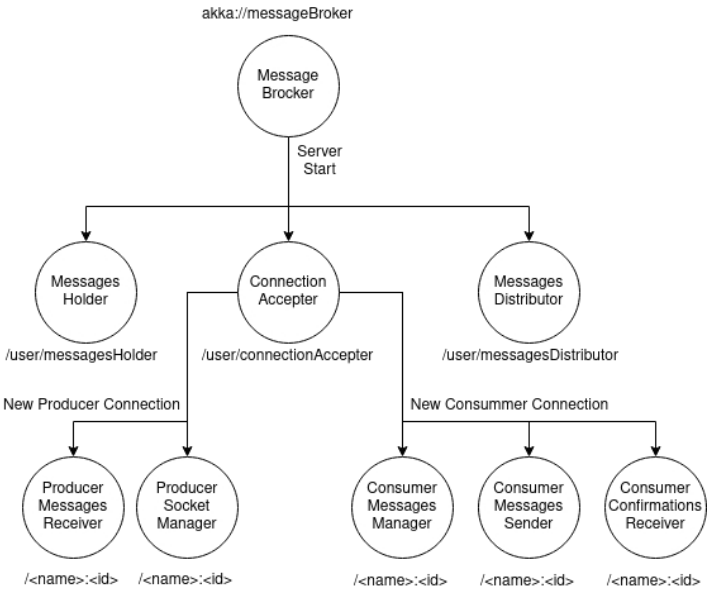
The MessagesDistributor actor has a list of references to the ConsumerMessagesManager actors, each associated with their topics list. It's goal is to distribute the messages from the MessageHolder to each Consumer, based on the topics it subscribed to.

The ConsumerMessagesManager has a list of messages to send. A message is stored for re-sending. A message is deleted when a confirmation of Consumer's reception is obtained from the ConsumerConfirmationsRecceiver.

The ProducerSocketManager holds the socket object. As long as the socket is open, a Listen command is sent to ProducerMessagesReceiver. This command is requested by the receiver actor itself, in a closed loop, until the receiver asks to close the socket. After this, both actors consume a PoisonPill.

The client connection actors are stopping by consuming PoisonPill from the ConsumerConfirmationsReceiver, when a connection-close command is received.

1.3.2 Actors hierarchy



The MessagesHolder and MessagesDistributor are created when the application starts. At this time, MessagesHolder restores the persistant messages.

The producer and consumer related actors are created on the connection establishment, and named by their name and the index which increments on every connection.