Distributed Systems Project

Reliable Broadcast Library

authors:
Luigi Fusco
Arianna Galzerano
Lorenzo Mainetti

Project Scope

- **Reliable Broadcast Library**
  - Reliable broadcast communication among faulty processes
  - Virtual synchrony
  - FIFO ordering
- **Application** to test the library
- **Java** implementation
- **Docker** deployment

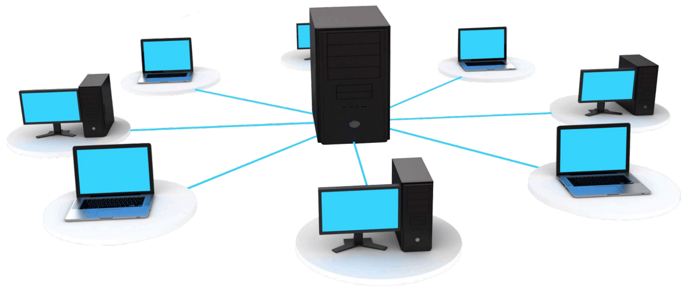
Virtual Synchrony

- Messages are delivered in order
- Crashed processes are removed from the view
- New processes have to be added to the view
- A message is received by either all or none of the processes in the view

Assumptions

Following the project specifications, the assumptions taken into consideration were:

- LAN scenario
- No joins or disconnections during the view change process
- No process fails during the time required for previous failures to be recovered
- No partial drops of ACK messages



Use Cases

Functionalities:

- Process joins
- Process leaves
- Process sends message
- Process receives message in FIFO order

Corner Cases:

- Process drops an incoming message (network failure)
- Process sends unordered messages
- Process fails after some processes received its messages, others did not
- Process fails after receiving a message before acknowledging it

Design Choices - Supervisor

There is a process which has the role of a single **supervisor** (leader).

Its tasks in the library consist of:

- Single authority on the composition of the view
- Orchestration of the process of view change
- Detection of failures through the use of PING messages and timeouts

Design Choices - Sending Functionality

Regarding the network protocol, **UDP** broadcast is used to send the messages.

Sent messages are marked as pending. Specifically:

- Pending messages can be resent when a *NACK* is received
- Pending messages are deleted only after they are *acknowledged* by all clients in the view

Design Choices - Receiving Functionality

While the FIFO order for the receivement of the messages is ensured with a sequence number and ACK/NACK mechanism. In particular:

- Incoming messages are marked as received
- If a message is received out of order, a NACK is sent for each missing message
- Messages are acknowledged only when received in the correct order

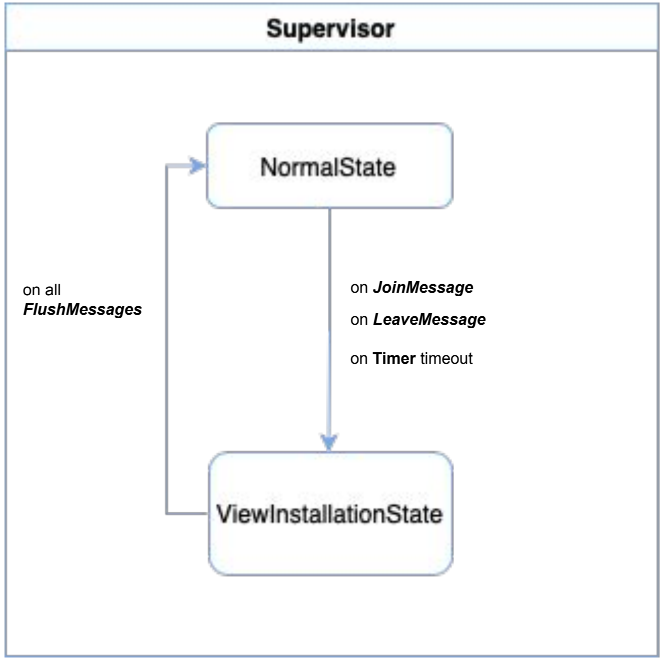
Design Choices - Receiving Functionality

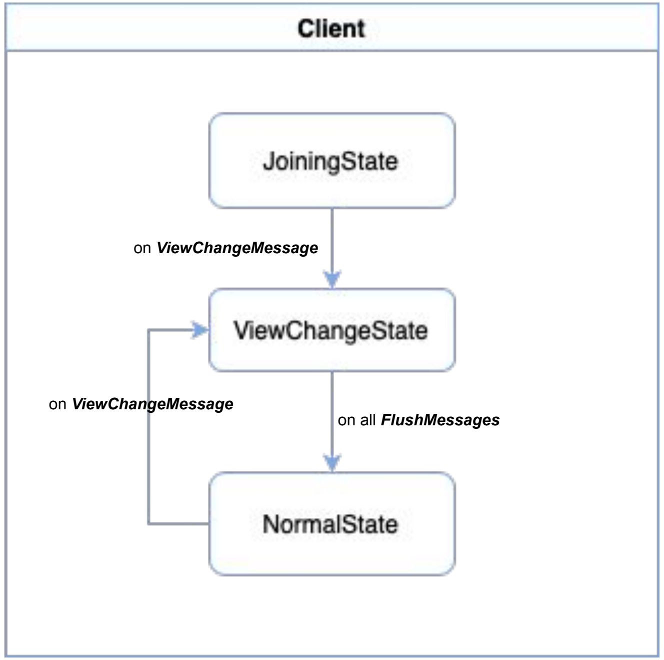
Pending received messages are marked as delivered only when an acknowledgement is received by all other processes of the view (sender excluded). This is to make sure that either all or none of the processes of the view deliver the message in case of partial delivery followed by a crash.

In general we prefer **false negatives** instead of **false positives.**

(Following the virtual synchrony, Messages from a failing process are processed either by all correct members or by none)

Closer Look - Architecture





Thank you for the attention!

DEMO in …