Distributed System 1 - lab 1

Francesco Penasa

March 30, 2020

`davide.vecchia@unitn.it` `timofei.istomin@unitn.it` We will implement some distributed algorithms seen during the course.

akka `akka.io` framework for reactive concurrent and distributed applications in Java. Actors interacts by sending messages to each other, the state change w.r.t. the message. Messages put in a queue and a dispatcher decide how to handle them.

Actor model

1. variable
2. queue
3. dispatcher
4. handlers
5. messages

Akka guarantees

1. Actor may run in parallel
2. Message queues are FIFO
3. Messages might be lost
4. No actor is running in more than one thread at a time.

Rules

1. encapsulate! Make sure no object is accessible from multiple actors instances.
2. Don't send recerences to mutable objects, send copies instead.
3. Don't use non-final static variable in the actor class nor other classes accessible from the actor
4. Don't use threads that may access internals of an actor.
5.

Akka message Serializable. String is inmutable, we don't need final to be sure that the variable is not tampered. Serializability: turn an object into a series of bytes.

akka messagges should be: immutable, final and implement serializable

1 Exercise

1.1 Causal Multicast

1. We'll create a toy group chat application
2. There will be a group of actors that send chat messages to the whole group (multicast)
3. For simplicity: all the actors will run locally
4. The chat system should guarentee the property of causal delivery.