POLITECNICO
MILANO 1863

Actor Model – Akka Lab

Luca Mottola

luca.mottola@polimi.it
http://mottola.faculty.polimi.it

Exercise 1

- Modify the CounterActor code to accept different messages that either increment or decrement the counter
  - Try both with different classes of messages and with message selectors that predicate on the content of messages
  - For testing, you may simply duplicate the line **exec.submit()** to increment or decrement the counter within the **for** loop

Exercise 2

- Now modify either solution to Exercise 1 to make sure that the state of the Counter actor is never negative
  - If a message is received that would try and decrement the counter when the state is zero, the message shall be processed later
- To use stash(), extend from **AbstractActorWithStash**

- For testing, verify a sequence of
  - decrement
  - decrement
  - increment
  - decrement
  - increment
  - increment
- The actor state should evolve as 1, 0, 1, 0, 1, 0

Exercise 3

- Write a server actor that manages a contact list that associates **email addresses** to **names**

- The server actor processes two types of messages
  - **PutMsg** to add a new contact in the list
  - **GetMsg** to read the address of a contact given its name

- Write an actor client that sends messages to the server actor and, in the case of a **GetMsg**, expects a reply and displays the content on the console

Exercise 4

- Write a server actor that receives text messages and sends them back to the client actor
  - Upon receiving a **Sleep** message
    - It stores the subsequent messages
    - It stops sending messages back to the client
  - Upon receiving a **Wakeup** message
    - It sends the accumulated messages back to the client
      - In the same order in which they were received
    - It starts again to send new messages back to the client
- Hint: check who is the **sender()** when un-stashing messages!

Exercise 5

- Modify the solution to Exercise 3 so that the server actor fails every time a **PutMsg** is received with name “Fail!”
  - You may simply raise a generic Exception
- Create a supervisor actor that manages the fault
- Verify that using a **resume** strategy preserves the contact list, whereas a **restart** strategy effectively resets the contact list