Lab 3 : RabbitMQ
Official documentation: https://www.rabbitmq.com

1. Goals

In this lab you will work with the RabbitMQ middleware which provides different mechanisms for indirect communication. Contrary to Java RMI, it does provide failure transparency.

This lab is graded, the core work is on questions 4 and 5.

2. RabbitMQ Concepts (**this is NOT optional**)

Start by reading and understanding https://www.rabbitmq.com/tutorials/amqp-concepts.html.

3. From the tutorial (**this is NOT optional**)

Go at http://www.rabbitmq.com/getstarted.html and follow at least
the « HelloWorld », « Work Queues » and « Publish /Subscribe » tutorials.

For each tutorial:

- you need to be able to run it
- You need to pay attention to the queues, exchanges and bindings i.e. you need to understand how the interaction goes : who sends where and who receives where/how?

4. Chat with RabbitMQ

You have just implemented a chat application with Java RMI. Do the same (implement an app with the same specification and the same properties) with RabbitMQ. Compare the two solutions using the criteria of facility of development (without considering that learning RabbitMQ is hard :)) and lines of code.

5. Ring with RabbitMQ

4.1 Write a program that creates a unidirectional ring containing three nodes one of which is the initiator. The initiator sends a message that goes through the ring. The program stops when the message has done its tour.

For the creation of the different nodes you can use Java Threads. For the communication between nodes you can use RabbitMQ communication channels.

4.1 Generalize the previous program so as to create a ring of N nodes.

BONUS

6. Election on a Ring

Modify the previous program so as to implement an election algorithm (the node with the smaller ID is to be elected).

7. Tree with RabbitMQ

Write a program that takes as an input a matrix that defines a tree topology and instantiate it using RabbitMQ.

8. Diffusion on a tree

Using your RabbitMQ tree, implement the diffusion algorithm (one node sends a message to all nodes). You can first consider the simple version (without acknowledgements) and then consider the second version (with acknowledgements).