Assignment CDS 2016/2017

1. Implement a synchronization mechanism similar to the mechanism provided by Java within the java.util.concurrent packages (explicit locks and condition variables) but whose behaviour is in accordance with the semantic "signal-and-urgent". For the implementation of this mechanism you can use only the built-in synchronization constructs provided by Java (i.e. synchronized blocks or synchronized methods) and the methods wait(), notify() and notifyAll() provided by the class object). In particular:
   - 1.1. Implement the class FairLock that provides the two methods lock() and unlock(), to be used to explicitly guarantee the mutual exclusion of critical sections. Your implementation must guarantee that threads waiting to acquire a FairLock are awakened in a FIFO order.
   - 1.2. Implement also the class Condition that provides the two methods await() and signal() that are used, respectively, to block a thread on a condition variable and to awake the first thread (if any) blocked on the condition variable. In other words, condition variables must be implemented as FIFO queues. The semantics of the signal operation must be "signal and urgent". Remember that every instance of the class condition must be intrinsically bound to a lock (instance of the class FairLock). For this reason, the class FairLock provides, in addition to methods lock() and unlock(), also the method newCondition() that returns a new condition instance that is bound to this FairLock instance.
2.
   - 2.1. As a simple example of the use of the previous mechanism, implement a manager of a single resource that dynamically allocates the resource to three client threads: clienta1, clienta2 and clientb. If the resource is in use by clienta1 or by clienta2, when it is released and both clientb the other clienta are waiting for the resource, clientb must be privileged.
   - 2.2. Provide also the implementation of the same manager but now by using the analogous mechanism provided by Java (lock and condition variables) whose behaviour is in accordance with the semantics signal-and-continue and point out the differences, if any, between this implementation and the previous one.
3.
   - 3.1. By using the language PSF, provide the design model of the problem described at point 2.1.
   - 3.2. From the design model described at point 3.1., derive the corresponding Java program implemented by using the lock and condition variables provided by Java and whose behaviour is in accordance with the semantics signal-and-continue.
   - 3.3. By modelling this implementation with the FSP language, verify that it satisfies the problem's specification.