

EXERCISE 2: DISTRIBUTED MUTUAL EXCLUSION

Learning Objectives

By the end of this exercise, you should be able to:

- Implemented clock based mutexes.
- Understand distributed system synchronization challenges.

Exercise statement

We need to design and implement a distributed application. This application must have two *heavyweight* processes **ProcessA** and **ProcessB**. ProcessA must invoke 3 lightweight processes **ProcessLWA1**, **ProcessLWA2** and **ProcessLWA3**. ProcessB, on the other hand, must invoke 3 processes **ProcessLWB1**, **ProcessLWB2**, **ProcessLWB3**. Each lightweight process must live in a infinity loop which will consist of showing your ID on the screen 10 times and while waiting 1 second.

Both heavyweight processes have to run on the same machine, so all lightweight processes will compete for the same shared resource: the screen. A token-based mutual exclusion policy will need to be implemented between the two heavyweight processes. Among the processes invoked by ProcessA, a *Lamport's policy* must be implemented for mutual exclusion. Among the processes invoked by ProcessB, *Ricart and Agrawala's policy* will need to be implemented for mutual exclusion.

So, when running our distributed application, an output like the following should appear:

```
I'm lightweight process A1
     I'm lightweight process A1
3
     I'm lightweight process A1
4
     I'm lightweight process A1
5
     I'm lightweight process A2
6
     I'm lightweight process A2
7
8
     I'm lightweight process A2
9
     I'm lightweight process A2
10
     I'm lightweight process A3
11
     I'm lightweight process A3
12
13
```



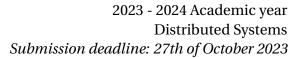
```
I'm lightweight process A3
      I'm lightweight process A3
15
      I'm lightweight process B1
16
      I'm lightweight process B1
17
18
      I'm lightweight process B1
19
      I'm lightweight process B1
20
      I'm lightweight process B2
21
      I'm lightweight process B2
22
23
      I'm lightweight process B2
24
      I'm lightweight process B2
25
      I'm lightweight process B3
26
      I'm lightweight process B3
27
28
```

Although all *heavyweight* and *lightweight* processes must run on the same machine, these must be independent programs that can only communicate with each other via *sockets*.

Below is an example of the skeleton of each of the heavyweight processes:

```
1
     ...
     while(1){
2
        while(!token) listenHeavyweight();
3
        for (int i=0; i<NUM LIGHTWEIGHTS; i++) {
4
           sendActionToLightweight();
5
6
        while(answersfromLightweigth < NUM LIGHTWEIGHTS) {
7
           listenLightweight();
8
9
        token=0;
10
        sendTokenToHeavyweight();
11
     }
13
```

Below is an example of the skeleton of each of the lightweight processes:





```
while(1){
2
        waitHeavyWeight();
3
        requestCS();
4
        for (int i=0; i<10; i++){
5
           printf("I'm lightweight process %s\n", myID);
6
           waitSecond();
7
8
        releaseCS();
9
        notifyHeavyWeight();
10
     }
11
     ...
12
```