Universitetet i Oslo
Institutt for Informatikk

PMA

Olaf Owe, Martin Steffen

INF 4140: Models of Concurrency

Høst 2015

Series 8

10. 11. 2015

Topic: RPC and Rendezvouz

Issued: 10. 11. 2015

Exercise 1 (Time server RPC improvement) Solve [1, Exercise 8.1]: Consider the first implementation of the time server, the one based on RPC. The improvement should achieve that not every "tick" awakes the processes. Instead, the clock process should set the HW timer to go off at the *next interesting event.* Assume the timer is maintained in millisec's and that the timer can be set to any number of millisec's. Assume also that the processes can read how much time is left before the HW timer goes off. Finally assume that the timer can be reset any time.

Exercise 2 (Time server with rendezvouz.) Do [1, Exercise 8.5]: Rewrite the time server from [1, Figure 8.7] so that the `delay` operation works with *relative* time, i.e., it specifies an interval as in [1, Figure 8.1].

Exercise 3 (Minimum) Do [1, Exercise 8.8]: Given a set of integers, calculate the mimimum. Given is as an *array* of processes named `Min`, i.e., we are given `Min[1 : n]`, where each process initially contains one integer value. The processes repeatedly *interact* with each other, each one giving another processes the minimum value it has seen so far. After having given away its minimum value, a process terminates. Eventually, one process will be left, and this one contains the overall minimum value.

1. Solve that problem using RPCs.
2. Do the same with rendez-vouz.

References

[1] G. R. Andrews. *Foundations of Multithreaded, Parallel, and Distributed Programming.* Addison-Wesley, 2000.