UMEÅ UNIVERSITY, DEPARTMENT OF COMPUTING SCIENCE

Written Assignment 1 (40 points)

Distributed Systems - 5DV147

You need to complete and submit this assignment individually. Collaboration is permitted only as described in the Guidelines for compulsory assignments and the Honor code. You need to submit this assignment by September 19 at 13:00 (hard deadline). You can either email your solution to us (5dv147-staff(at)cs.umu.se) or drop off your solution in the box outside the department.

1 LOGICAL TIME (12 POINTS)

Give a distributed algorithm to maintain vector clocks for a group that has a dynamic number of processes. Assume that there are the following events in the life of any process: *start-process*, *send*, *receive*, *terminate-process*. Also assume that a process can *join a group*, l*eave a group temporally*, and *leave a group permanently*.

2 MUTUAL EXCLUSION (14 POINTS)

1. In a certain system, each process typically uses a critical section many times before another process requires it. Explain why Ricart and Agrawala's multicast-based mutual exclusion algorithm is inefficient for this case, and describe how to improve its performance. Does your adaptation satisfy the liveness condition for mutual exclusion? (7 points)

2. Modify the Ricart and Agrawala algorithm to permit up to K simultaneous entries into the critical section. Does your algorithm satisfy the requirements for mutual exclusion (safety, liveness, and $\rightarrow$ ordering)?(7 points)

3 ELECTIONS (6 POINTS)

Suggest how to adapt the Bully algorithm to deal with temporary network partitions (slow communication) and slow processes. Remember, we studied that algorithm in class.

4 GROUP COMMUNICATION (8 POINTS)

Consider the following algorithms for *multicast* and *deliver* of messages in FIFO order.

FO-multicast

Send $S(p_i,g)$ with message

    B-multicast message

    $S(p_i,g) = S(p_i,g) + 1$

FO-deliver

If $S=R(p_j, g) + 1$

    FO-deliver and set $R(p_j, g) = S$

If $S>R(p_j, g) + 1$

    Place in hold-back queue until

    $S = R(p_j, g) + 1$

Show that the algorithms do not work for overlapping groups, by considering two messages sent from the same source to two overlapping groups, and considering a process in the intersection of those groups. Adapt the protocol to work for this case. *Hint*: processes should include with their message the latest sequence numbers of messages sent to all groups.