```
module TimeServer
  op get_time() returns int;  # retrieve time of day
  op delay(int interval);     # delay interval ticks
body
  int tod = 0;          # the time of day
  sem m = 1;            # mutual exclusion semaphore
  sem d[n] = ([n] 0);   # private delay semaphores
  queue of (int waketime, int process_id) napQ;
  ## when m == 1, tod < waketime for delayed processes

  proc get_time() returns time {
    time = tod;
  }

  proc delay(interval) {    # assume interval > 0
    int waketime = tod + interval;
    P(m);
    insert (waketime, myid) at appropriate place on napQ;
    V(m);
    P(d[myid]);   # wait to be awakened
  }

  process Clock {
    start hardware timer;
    while (true) {
      wait for interrupt, then restart hardware timer;
      tod = tod+1;
      P(m);
      while (tod >= smallest waketime on napQ) {
        remove (waketime, id) from napQ;
        V(d[id]);  # awaken process id
      }
      V(m);
    }
  }
end TimeServer
```

Figure 8.1 A time server module.