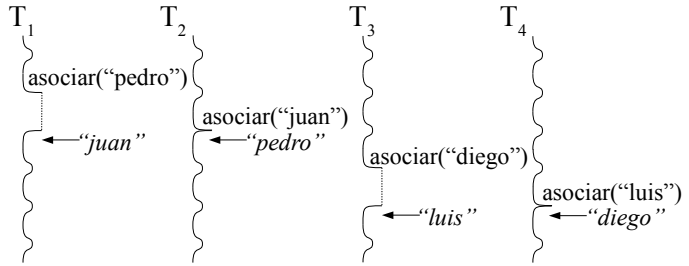
CC4302 Sistemas Operativos – Control 1 – Semestre Primavera 2016 – Prof.: Luis Mateu

Pregunta 1

La función *asociar* debe obtener el nombre de un socio único. Recibe como argumento el nombre de la persona que busca socio y debe retornar el nombre de su socio. Al llamarla pueden ocurrir solo 2 casos: (i) no hay un socio pendiente, en cuyo caso el llamador queda pendiente hasta que otro thread invoque *asociar*, o (ii) hay un socio pendiente, en cuyo caso el socio del llamador es el socio pendiente y viceversa, y no queda ningún socio pendiente. El siguiente diagrama muestra un ejemplo de ejecución:



La siguiente solución es incorrecta aunque funciona casi siempre:

```
nSem s; // = nMakeSem(0);   ¡0 tickets iniciales!
char **psocio= NULL;

char *asociar(char *nom) {
  if (psocio==NULL) {
    psocio= &nom;     // no hay un socio pendiente
    nWaitSem(s);      // nom queda pendiente a la espera de un socio
  }
  else {
    char *socio;      // sí hay un socio pendiente
    socio= *psocio;   // el nombre del socio pendiente
    *psocio= nom;     // cambia el parámetro nom en el thread pendiente
    nom= socio;       // el nombre del socio de este thread
    psocio= NULL;     // ahora no hay ningún socio pendiente
    nSignalSem(s);    // reanuda al socio
  }
  return nom;         // entrega el nombre del socio
}
```

a.- Haga un diagrama de threads que muestre que *asociar("pedro")* podría retornar incorrectamente el mismo *"pedro"*.

b.- Corrija esta solución usando uno o más semáforos adicionales. No olvide indicar los tickets iniciales. No puede usar monitores. Ayuda: lo único incorrecto de la solución de más arriba es el aspecto sincronización.

Pregunta 2

Por cada iteración del siguiente código secuencial se producen 2 átomos de hidrógeno, un átomo de oxígeno, se forma una molécula de H_2O y luego se consume la molécula:

```
for(;;) {
  Oxygen *o= produceOxy();
  Hydrogen *h1= produceHydro();
  Hydrogen *h2= produceHydro();
  H2O *h2o= makeH2O(h1, h2, o); // fabrica molécula de H2O
  consume(h2o);
}
```

Se necesita paralelizar este código. Para lograrlo se han lanzado múltiples threads productores de hidrógeno (ejecutan la función *hydrogen*) y múltiples threads productores de oxígeno (ejecutan la función *oxygen*) que además consumen la molécula de H2O. Este es su código:

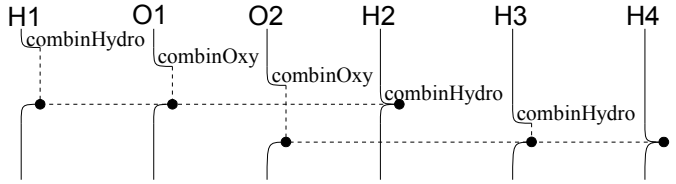
```
void oxygen() {
  for(;;) {
   Oxygen *o= produceOxy();
   H2O *h2o= combinOxy(o);
   consume(h2o);
} }
```

```
void hydrogen() {
  for(;;) {
   Hydro *h= produceHydro();
   combinHydro(h);
} }
```

Programe las funciones *combinOxy* y *combinHydro* con encabezados:

```
H2O *combinOxy(Oxygen *o);
void combinHydro(Hydrogen *h);
```

La función *combinOxy* debe esperar por 2 átomos de los productores de hidrógeno y formar la molécula llamando a *makeH2O* (vea el código secuencial). La función *combinHydro* solo puede retornar una vez que su átomo formó una molécula. Para la sincronización Ud. debe usar un solo monitor de nSystem. El siguiente diagrama muestra un ejemplo de la sincronización requerida:



Ud. solo debe programar las funciones *combinOxy* y *combinHydro*. El resto de las funciones, incluyendo *oxygen* y *hydrogen*, así como las estructuras de datos son dadas y por lo tanto no puede modificarlas.