ACTIVIDADES SEMINARIO 2 : JOSE SANTOS SALVADOR

Actividad 1

Cuestion 1:
Al quitar el unlock el cerrojo se queda completamente cerrado siempre, ya que no se desbloquea nunca y por tanto el programa no puede continuar.

Cuestion 2:
(1) La ultima hebra en entrar siempre es la primera en salir ya que es la que desbloquea al resto de hebras

(2) El orden de salida y de entrada no coinciden ya que cada hebra compite con las otras hebras tanto por entrar como por salir ya que en eso consiste el notify y no se puede asegurar la salida de una hebra concreta por su parte.

(3) No necesita ejecutarse el constructor en exclusión mutua porque no lo llamamos de forma recurrente en el main, unicamente se llama de esta forma
MBarreraSC monitor(num_hebras);
y después ya se llaman a las hebras.

(4) Al hacer notify_all() en vez del for con notify_one(), la hebra señaladora continua la ejecución y las hebras señaladas esperan en la cola del monitor y con notify_one() tiene el cerrojo hasta que sale del monitor(terminando el metodo que está ejecutando). En una opción se desbloquea una por una y se da salida para una hebra aleatoria a competir unicamente y con all() se abre para todas las hebras compitiendo entre ellas.

Actividad 2

Para poder realizar las nuevas propiedades de barrera parcial es necesario disponer de 3 colas distintas, la primera de ella es la cola asociada a la variable condicion del monitor(la primera entrada al monitor) y esta tras estar bloqueada se necesitaria otra nueva cola prioritaria asociada a la salida del monitor por parte de las hebras siendo esta una cola prioritaria regida por el orden de llegada al monitor donde la primera hebra en entrar es la primera en salir y llama a la siguiente hebra que entró para salir despues de ella. Por lo tanto cada hebra llama a la siguiente en salir y en la cola prioritaria de salida del monitor quedarian ordenadas para salir en orden de llegada. Y también dispondriamos de una ultima cola asociada a la entrada del monitor, que carece de prioridad y solo podrian entrar al monitor tras la salida completa de la cola prioritaria de salida. Se puede observar el diseño final con la implementación de colas y bloques de hebras distintas.

Actividad 3

Para modificar la implementación de ProdConsSC versión FIFO voy a escribir las cosas que he añadido y/o modificado:

```
class ProdCons1SC
{
 private:
 static const int           // constantes:
   num_celdas_total = 10;   //  núm. de entradas del buffer
 int                        // variables permanentes
   buffer[num_celdas_total],//  buffer de tamaño fijo, con los datos
   primera_libre ,          //  indice de celda de la próxima inserción
   primera_ocupada ;
```

```
ProdCons1SC::ProdCons1SC(  )
{
   primera_libre = 0 ;
   primera_ocupada = 0 ;
}


int ProdCons1SC::leer(  )
{
   // ganar la exclusión mutua del monitor con una guarda
   unique_lock<mutex> guarda( cerrojo_monitor );

   // esperar bloqueado hasta que 0 < num_celdas_ocupadas
   if ( primera_ocupada == 0 )
      ocupadas.wait( guarda );

   // hacer la operación de lectura, actualizando estado del monitor
   assert( 0 < primera_ocupada  );
   const int valor = buffer[primera_ocupada] ;
   primera_ocupada = (primera_ocupada + 1) % num_items ;


   // señalar al productor que hay un hueco libre, por si está esperando
   libres.notify_one();

   // devolver valor
   return valor ;
}
```