INFORME PRÁCTICA 4. PSCD

Nombre: Diego Marco Beisty
NIP: 755232

Decisiones de diseño:
El monitor PuntoAlquiler consta de dos operaciones.
Para sincronizar la primera operación de solicitud de bicicleta, he optado por una variable condición “hayBicis” sobre la que se bloquean los clientes cuando no hay ninguna bici disponible y otra variable condición “hayCompagnero” sobre la que se bloquea un cliente cuando hay una bici tandem disponible pero no tiene compañero para poder alquilarla.

Para sincronizar la segunda operación de devolución de bicicleta he optado por un array de variables condición “poderDevolver[i]”, una por cada cliente i, i=1..20. Cuando un cliente quiere devolver una bici tandem y su compañero todavía no se encuentra en el punto de alquiler para devolverla, se bloquea sobre su variable condición. Cuando llega su compañero, lo despierta y ambos devuelven la bici tandem.

Principales dificultades:
Mi diseño original asumía erróneamente una política de reanudación E < S < W. Por lo tanto se generaba una historia no deseada cuando había un cliente solicitando una bici tandem bloqueado sobre “hayCompagnero”, llegaba el segundo cliente que solicitaba la misma bici tandem desbloqueando al primer cliente, pero antes de que el primer cliente se desbloquease e indicase que se había alquilado la bici tandem, entraba otro cliente al monitor y veía que había una bici tandem falsamente disponible.
Esto lo he arreglado añadiendo la condición de que un cliente también se bloquea sobre la variable condición “hayBicis” si hay bicis tandem disponibles y 2 clientes solicitando una bici tandem, es decir, todavía no la han alquilado.

Comportamiento observado:
Inicialmente hay 5 bicis individuales y 3 bicis tandem disponibles.
<u>Solicitando una bicicleta:</u>
Cuando un cliente logra alquilar una bicicleta individual bicisDisp se decrementa en 1.
Cuando no hay bicis individuales disponibles y pero sí bicis tándem y además no hay nadie solicitando la bici tandem, el cliente se bloquea sobre “hayCompagnero” esperando un compañero.
Cuando no hay bicis individuales, hay bicis tándem y un cliente solicitando una bici tandem, se desbloquea a ese primer cliente solicitando la bici tándem. Como no se cumple que E < S < W, antes de que el proceso desbloqueado pueda volver a ejecutarse, entran nuevos procesos al monitor, que se bloquean sobre la primera variable condición “hayBicis”. Finalmente el proceso desbloqueado sale del monitor desbloqueando a los demás procesos. tandemsDisp se decrementa en 1.

Se observa que si antes de que llegue un segundo cliente solicitando la bici tándem, se devuelve una bici individual, el primer cliente bloqueado solicitando la bici tándem se desbloquea y alquila la bici individual. bicisDisp se decrementa en 1.

Devolviendo una bicicleta:

Cuando un cliente devuelve una bici individual, bicisDisp se incrementa en 1 y se desbloquea, primero un cliente que esté esperando una bici tandem, si lo hay, y después los procesos bloqueados sobre la variable condición “hayBicis”.

Cuando un cliente devuelve una bici tandem y su compañero todavía no ha llegado, se bloquea sobre la variable condición “poderDevolver[id - 1]”, siendo id el identificador del cliente que ha llegado.

Cuando un cliente devuelve una bici tandem y su compañero ya ha llegado, desbloquea a su compañero que lo estaba esperando y se incrementa tandemsDisp en 1, desbloqueando todos los clientes sobre la variable condición “hayBicis”.

Finalmente todos los clientes han devuelto las bicis alquiladas y hay 5 bicis individuales y 3 bicis tandem.

Ficheros que implementan la solución:

main_p4.cpp
Makefile_p4
PuntoAlquiler.cpp
PuntoAlquiler.hpp