Exclusión mutua

El objetivo es garantizar que no haya vehículos transitando el túnel a la vez en direcciones opuestas. En otras palabras, el número de vehículos en una dirección y el número en la otra no pueden ser simultáneamente estrictamente positivos. Además son números naturales. El invariante es:

$$coches\_norte \geq 0 \wedge coches\_sur \geq 0 \wedge \neg(coches\_norte > 0 \wedge coches\_sur > 0)$$

Descripción del programa

La propiedad que caracteriza a cada vehículo (proceso) es su dirección. Los vehículos que no comparten esta propiedad no deben poder interactuar con el monitor al mismo tiempo. El monitor refleja este hecho manejando ambos tipos de procesos por separado. Cada tipo (dirección) tiene su propio contador y variable condición.

Cuando un proceso llama al método *wants_enter* y accede al monitor, se incrementa en 1 el conteo de vehículos en esa dirección. Tras un tiempo, llama a *leaves_tunnel* y se decrementa en 1. Como todos los vehículos deben terminar saliendo del túnel, el efecto sobre su valor es nulo.

Los contadores se emplean para el correcto funcionamiento de las dos variables condición que se usan. Para regular el acceso al monitor, cada proceso debe verificar si se da la condición de entrada, es decir, que el correspondiente *coches_{direccion opuesta}* sea exactamente 0. En la definición de *wants_enter* se ejecuta justamente el método *wait_for* de la variable condición, que bloqueará al proceso si no es posible continuar.

Finalmente, los procesos que salen (han ejecutado un *leaves_tunnel*) del túnel mandan un aviso a ambas variables condición mediante un *notify_all*, para que los procesos dormidos tengan oportunidad de usar el monitor.

Algunas consideraciones adicionales

Los dos métodos compartidos del monitor se sincronizan con un *lock*. El cerrojo es liberado al llamar a *wait_for*. Esto puede derivar en un *deadlock*: si se incrementa el contador antes de ejecutar el *wait_for*, podría ocurrir que ambos contadores fuese estrictamente positivos y ningún vehículo entraría. Por tanto, el contador se incrementa después de *wait_for* y no antes.