PRÁCTICA 5 : *Paso de mensajes*

OBJETIVOS:

1. Comprender y saber resolver problemas de **comunicación y sincronización** de tareas concurrentes, mediante el uso el paso de mensajes en Ada.
2. Saber diferenciar la comunicación y sincronización de tareas mediante **paso de mensajes** y mediante **memoria compartida**.
3. Saber utilizar la sentencia de **transferencia asíncrona de control** (*select…then abort…*) para la captura de eventos asíncronos.

PERIODO RECOMENDADO PARA SU REALIZACIÓN: 2 semanas

ENUNCIADO: Control de averías de coches.

Como continuación de la práctica 4, vamos a empezar a añadir un sistema de comunicación y reparación de distintos tipos de averías de coches (los taxis no tendrán averías).

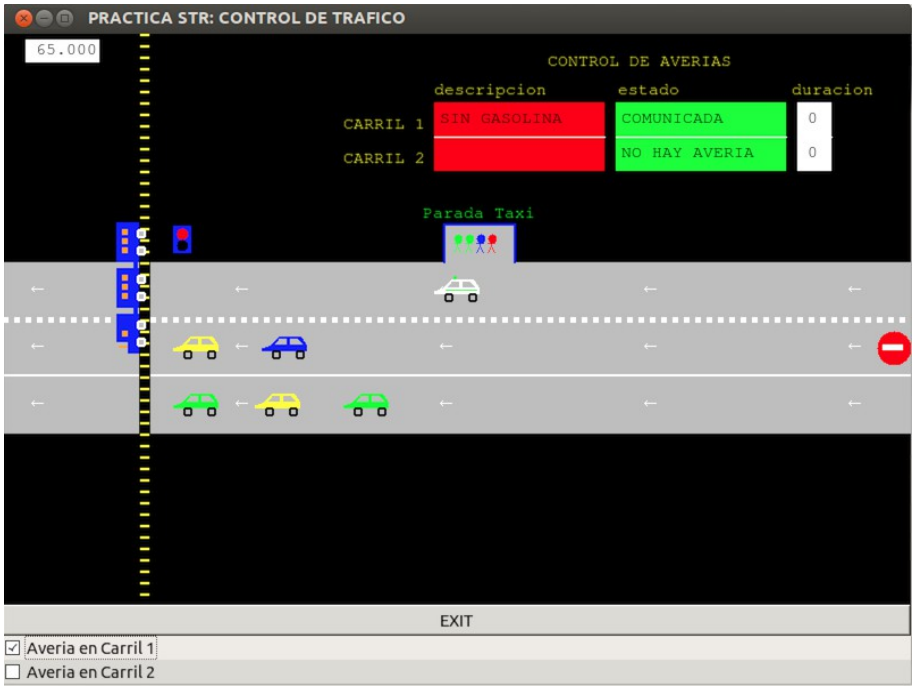

Figura 1. Nuevos elementos del sistema simulado:

1. Descripción, estado y tiempo de reparación de la avería de un coche en un carril (situados en la parte superior de la ventana).
2. Señal de prohibido el paso que aparecerá cuando hay una avería en un carril (al principio del mismo).
3. Botones para provocar averías (situados en la parte inferior de la ventana).

PASOS A SEGUIR:

PASO 1: Añadir y actualizar nuevas versiones de paquetes

En primer lugar debes sustituir los paquetes ***pkg_tipos*** y ***pkg_graficos*** por sus nuevas versiones proporcionadas, y añadir a tu proyecto el nuevo paquete proporcionado ***pkg_averias***.

Haz un build de tu proyecto y ejecútalo. Observa los nuevos elementos que aparecen en el sistema y comprueba que si pulsas cualquiera de los botones para provocar una avería en el coche de un carril, se visualiza "COMUNICADA" como nuevo estado de avería en ese carril y aparece siempre como descripción de la avería generada "SIN GASOLINA". De momento, el sistema sólo responde a la primera pulsación en cada botón, e ignora posteriores pulsaciones sobre los mismos.

PASO 2: Interrumpir la aparición de coches en un carril

El paquete ***pkg_averias*** contiene el array ***OP_Indicador_Averia*** de objetos protegidos de tipo ***T_OP_Indicador_Averia***. Este array de tamaño 2, tiene dos objetos protegidos, uno por cada carril, para que puedas acceder al mismo por el índice que identifica al carril. Los objetos protegidos contienen el valor actual del estado de la avería de un coche en el carril correspondiente. Cuando se pulsa sobre un botón para provocar una avería, además de cambiar visualmente el rótulo correspondiente (como se ha visto en el paso anterior) también se actualiza el estado del objeto protegido correspondiente (esta funcionalidad ya está implementada).

En este paso debes modificar tu tarea que define el comportamiento de generar coches en un carril. En concreto, debes hacer que se interrumpa la creación de coches en el carril cuando el indicador correspondiente de avería de dicho carril indique que se ha comunicado una avería (valor ***pkg_tipos.COMUNICADA_AVERIA*** del tipo enumerado ***pkg_tipos.T_Estado_Averia***). En caso de que ocurra esto, además habrá que dibujar la señal de prohibido el paso usando el procedimiento ***pkg_graficos.Dibujar_Carril_Cerrado***.

También deberás modificar convenientemente la implementación del tipo de objeto protegido ***T_OP_Indicador_Averia.***

Al finalizar el paso 2, se deberá haber añadido la siguiente funcionalidad a tu proyecto: la circulación de coches se interrumpe en un carril cuando se pulsa el botón correspondiente de ese carril para generar una avería. De momento, la circulación de coches quedará interrumpida indefinidamente en dicho carril. En el paso 4 veremos cuándo hay que reanudarla.

PASO 3: Provocar averías en los coches y comunicarlas a un teleoperador

En este paso debes modificar tu tarea que define el comportamiento de un coche. En concreto, al inicio de su bucle de control debes añadir lo siguiente:

1. Comprobar si el coche está averiado con la función ***pkg_graficos.Tiene_Averia***. **Aclaración**: Esta función sólo devolverá *true* para el último coche que está circulando en ese momento en un carril en el que hemos provocado una avería. No puede haber simultáneamente dos coches averiados en un mismo carril.
2. Si no está averiado, continúa con el resto de su ciclo de control ya implementado.
3. Si está averiado, hay que hacer lo siguiente:
   3.1. Parar el coche, actualizando su atributo velocidad a cero e invocar al procedimiento *pkg_graficos.Actualiza_Movimiento.*
   3.2. Generar un tipo de avería de forma aleatoria (usando ***pkg_tipos.T_Tipo_Averia***)
   3.3. Enviar un mensaje a una nueva tarea con la información del tipo de avería y del carril en el que se ha producido. Esta nueva tarea representará a un **teleoperador** que se encarga de recibir mensajes de coches averiados. Cada vez que reciba un mensaje, esta tarea debe visualizar la descripción de la avería (invocando al procedimiento ***pkg_graficos.Actualiza_Tipo_Averia_Carril***) y actualizar el estado de la avería en el correspondiente objeto protegido (valor ***pkg_tipos.REPARANDO_AVERIA*** del tipo enumerado ***pkg_tipos.T_Estado_Averia***). Añade esta nueva tarea en un nuevo paquete.
   3.4. Suspender la tarea coche durante 5 segundos. Esto lo hacemos de forma provisional, y será la simulación de que la reparación la hace el propio conductor. En la siguiente práctica, necesitaremos implementar nuevas tareas que representarán la asistencia técnica de mecánicos.
   3.5. Reanudar la marcha del coche (con el valor ***pkg_tipos.VELOCIDAD_COCHE***) una ve transcurridos el tiempo de reparación
   3.6. Continuar con el resto del ciclo de control que ya teníamos implementado

Al finalizar el paso 3, se deberá haber añadido la siguiente funcionalidad a tu proyecto: la visualización de un tipo de avería provocada cuando pulsamos el botón de avería en un carril, y parar durante 5 segundos el último coche de ese carril. Después de ese tiempo, el coche reanuda su marcha. **¿Por qué se ralentiza ahora la circulación del coche después de la reparación?**

PASO 4: Reanudar la aparición de coches en un carril

En este paso debes añadir lo siguiente:

1. Después de que un coche sea reparado, y antes de reanudar su marcha, debe enviar otro mensaje al teleoperador para indicarle en qué carril ha terminado su reparación.
2. El teleoperador cuando reciba el mensaje de fin de una reparación, debe actualizar el estado del objeto protegido correspondiente con el valor ***pkg_tipos.NO_AVERIA*** **.**
3. La tarea que genera coches en un carril, una vez que el estado del carril vuelve a ser de "NO AVERÍA", debe volver a reanudar la aparición de coches en su carril. Para visualizar esta nueva situación en nuestro sistema, utiliza el procedimiento ***pkg_graficos.Dibujar_Carril_Abierto*** para eliminar la señal de prohibido el paso en un carril.

Al finalizar el paso 4, se deberá haber añadido la siguiente funcionalidad a tu proyecto: que vuelvan a aparecer coches en un carril después de una reparación y que la circulación de los coches reparados no se vea ralentizada como ocurría en el paso 3.