Práctica 4

Sistemas Distribuidos

Grado en Ingeniería Telemática

Concurrencia en GO - Servidores

En esta práctica lo que se busca es definir la capacidad de desarrollo cuando la concurrencia sea avanzada, para la comunicación entre los diversos métodos en `GO` y un servidor básico.

Problema: Modernidad en El Taller del pueblo

Una cantidad `N` de coches requieren ser reparados en el taller del pueblo. El taller dispone de `NumPlazas` para los coches que esperan a ser reparados, y `NumMecanicos` mecánicos que se encargan de reparar los coches, cada mecánico puede hacer cualquiera de los 3 tipos de reparación (mecánica, eléctrica o de carrocería) y cada coche tendrá una única incidencia a resolver.

El procedimiento para reparar un coche costa de 4 fases secuenciales definidas de la siguiente manera:

1. El coche llega al taller y espera a que haya una plaza libre para entrar. Se tiene en cuenta la prioridad para asignar las plazas de espera. Mientras se prepara la documentación del coche, se espera el tiempo asignado por el tipo de prioridad.
2. Una vez dentro, el coche espera a que un mecánico esté libre para ser atendido. Se tiene en cuenta la prioridad para asignar los mecánicos libres. Mientras el mecánico revisa el coche, se espera el tiempo asignado por el tipo de prioridad.
3. Al terminar la reparación, el coche se limpia para que marche del taller. Se tiene encuentra la prioridad para el orden de limpieza. Mientras se limpia el coche, se espera el tiempo asignado por el tipo de prioridad.
4. Para entregar el coche al cliente, se realiza una revisión final. Se tiene en cuenta la prioridad para el orden de entrega. Mientras se revisa el coche, se espera el tiempo asignado por el tipo de prioridad.

A medida que van siendo reparados, los coches se van marchando del taller. Cada tipo de incidencia tendrá una prioridad diferente, de forma que los coches con prioridad alta serán atendidos antes que los de prioridad media, y estos antes que los de prioridad baja.

Debido a la escasez de mano de obra cualificada, se deberá clasificar a los coches que llegan al taller en 3 categorías diferentes. Cada categoría tendrá una importancia relativa mayor con respecto a la otra en función de la siguiente tabla:

- **Categoria A**: Los coches con incidencia de mecánica tendrán prioridad alta. Tiempo en cada fase de 5 segundos.
- **Categoria B**: Los coches con incidencia de eléctrica tendrán prioridad normal o media. Tiempo en cada fase de 3 segundos.
- **Categoria C**: Los coches con incidencia de carrocería tendrán prioridad baja. Tiempo en cada fase de 1 segundos.

El taller para mejorar su clientela ha decidido darle cobertura a los coches de una mutua, por lo que se ha designado un canal de información que es gestionado por un servidor que estará enviando información en una serie de números. Cada número tendrá un valor que designará el tipo de operación que deba hacer el taller. Los valores que entregara este servidor pueden ser los siguientes:

Numero	Estado	Descripción
0	Taller Inactivo	No hay atención
1	Solo Categoría A	Solo podrán acceder los coches de dicha categoría
2	Solo Categoría B	Solo podrán acceder los coches de dicha categoría
3	Solo Categoría C	Solo podrán acceder los coches de dicha categoría
4	Prioridad Categoría A	La prioridad de la atención lo tienen los coches de dicha categoría
5	Prioridad Categoría B	La prioridad de la atención lo tienen los coches de dicha categoría
6	Prioridad Categoría C	La prioridad de la atención lo tienen los coches de dicha categoría
7	No definido	Se mantiene el estado anterior
8	No definido	Se mantiene el estado anterior
9	Taller Cerrado	No hay atención

El hecho que un coche tenga prioridad elevada implica que al estamento con el cual contacte le atenderá de manera preferente frente a los demás que deberán esperar a ser atendidos.

Enunciado

Se requiere un programa en `GO` que modele el comportamiento del sistema anteriormente descrito, de forma que se pueda ejecutar dicha simulación una única vez y en función del programa denominado `mutua` , adjunto en aula virtual a esta práctica, gestione los cambios de estado que deba hacer el taller.

El programa debería cumplir las siguientes consideraciones:

- Cada fase de la reparación tiene un tiempo de uso/utilización que debe ser contemplado.
- Cada fase de la reparación tiene una variación de tiempo de uso/utilización que debe ser contemplado.
- Cada fase de la reparación debería tener una cantidad máxima de coches esperando a ser atendidos.
- El orden de entrada de los coches debe ser completamente aleatorio para cada categoría.
- Se debería visualizar en la ejecución el mensaje de `Tiempo {Tiempo_Ejecución_Programa} Coche {N} Incidencia {Tipo} Fase {Fase_Actual} Estado {Estado_Fase}` cada vez que un coche entre en una fase o salga de una fase.

Para su correcta implementación, en el aula virtual dispondrá de un conjunto de archivos cuya descripción es la siguiente:

- `servidor.go` : Programa que simula el comportamiento del servidor de información de la entidad aseguradora. Es el primero en ejecutarse, no visualiza información. **No se debe modificar**

- `taller.go` : Programa que simula el cliente que debe existir en el taller para obtener la información de `servidor.go` . Es el segundo en ejecutarse, y es el **único que se puede modificar**
- `mutua.go` : Programa que simula el comportamiento de la entidad aseguradora. Visualiza cada uno de los datos enviados al servidor.**No se debe modificar**

Además de las consideraciones anteriores, el programa debe tener una descripción técnica sobre su funcionamiento usando Diagramas de Secuencia y diagramas UML.

Tests

Para esta ocasión se deberá hacer una comparación entre los siguientes casos:

- Comparativas para cuando se atienden las siguientes cantidades de coches:

# Test	Categoría	Coches	Categoría	Coches	Categoría	Coches
1	A	10	B	10	C	10
2	A	20	B	5	C	5
3	A	5	B	5	C	20

Teniendo en cuenta la siguiente configuración de plazas y mecánicos, para cada test de cantidades de coches:

# Test	NumPlazas	NumMecanicos
1	6	3
2	4	4

En total, se deberían llevar a cabo 6 comparativas diferentes.

Para realizar estas comparativas es altamente recomendable que se use el paquete `testing` de `GO` .

Evaluación

Se tendrá en cuenta los siguientes items:

- Que realice las acciones requeridas para cada una de las peticiones de estado que realiza el software `mutua` . (60% de la práctica)
- Memoria técnica de la práctica, con código o repositorio, Diagramas de Secuencia, Diagramas UML. (30% de la práctica)
- El mínimo uso de estructuras de sincronización de procesos ofrecidas por el paquete *sync* de `GO` . (10% de la práctica)

> **NOTA:** Es **altamente recomendable** que se use un proyecto de referencia como el que se expone en el capitulo 8 del libro *The Go Programming Language*, proyecto que están disponibles en este Repositorio

Indicaciones adicionales

La descripción técnica, las métricas obtenidas en los tests y el código fuente del software (o link en repositorio si se prefiere), deberán estar registrados en un archivo de formato PDF, que deberá llamarse `Practica_4_nombre_SSOO_dist.pdf` , donde `nombre` sea tu nombre de usuario.

Luego este archivo se subirá a la actividad de aula virtual.