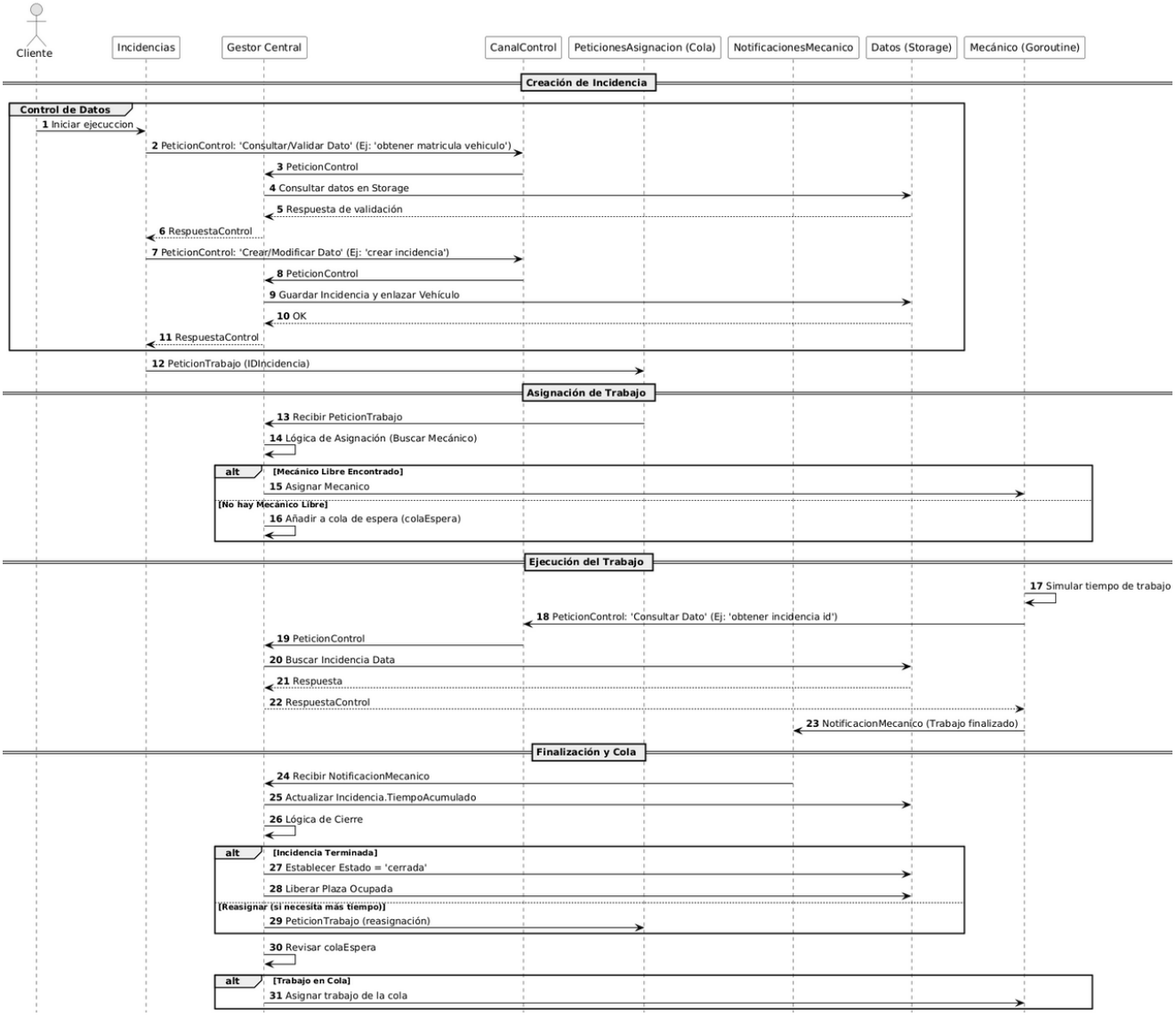
SSOO

Práctica 2 – Taller de coches en Go (Goroutines y Channels)

Alejandro Tejero de la Morena

1 - Explicación del sistema (diagrama de secuencia).



- El control del programa se divide en tres fases principales:

1. **Control de Datos**: Las operaciones de gestión de inventario (Crear/Validar Dato) son manejadas por el Canal Control, donde el módulo de interfaz (Incidencias en este caso) envía una petición y espera la Respuesta Control del Gestor Central antes de continuar.
2. **Asignación de Trabajo**: Al crearse una incidencia, el sistema utiliza el canal Peticiones Asignación para solicitar el trabajo. El Gestor Central aplica la lógica de asignación y, si no encuentra un Mecánico disponible, el trabajo se dirige a una cola de espera.
3. **Ejecución y Finalización**: La goroutine del Mecánico toma el trabajo y ejecuta la tarea. Al terminar, envía el resultado a través del canal Notificaciones Mecánico, permitiendo al

Gestor Central actualizar el estado de la incidencia en Datos (Storage), liberar la plaza de taller, y revisar la cola de espera para asignar el siguiente trabajo disponible.

2 - Resultados y análisis de los tests realizados.

2.1.1 - Metodología:

- Utilizo el paquete testing de Go, que se encarga de ejecutar las funciones Benchmark para simular la carga real del taller.
- Simulo un inicio antes de cada test donde se limpia el estado del taller y se pone todo vacío, perfecto para cada la ejecuccion correcta de cada test.
- Para los test uno y dos comparto una variable global donde se asigna el número de coches del taller, y para el test tres tengo otra para lo mismo, pero que se distribuira segun el tipo de incidencias de cada coche.

2.1.2 - Resultados:

- El resultado de cada test sale con la misma estructura, primero unos logs que indican la parte critica del proceso, para en caso de fallo saber que ocurre, y luego el nombre del test, el número de veces que se ejecutó, y el tiempo que tardo el test en nanosegundos / operación.

2.2 - Test 1: Comparativa cuando la cantidad máxima de coches con una sola incidencia se duplica.

```
BenchmarkCarga_Coches_Incidencias
    taller_test.go:89: Taller iniciado con 3 mecánicos (total plazas: 6)
    taller_test.go:168: Carga de 5 coches (mecanica) lanzada
    taller_test.go:181: Completados: 5/5
    taller_test.go:189: Todos los 5 trabajos completados
BenchmarkCarga_Coches_Incidencias-8                    1        31121545229 ns/op
BenchmarkCarga_DobleCoches_Incidencias
    taller_test.go:89: Taller iniciado con 3 mecánicos (total plazas: 6)
    taller_test.go:168: Carga de 10 coches (mecanica) lanzada
    taller_test.go:181: Completados: 5/10
    taller_test.go:181: Completados: 10/10
    taller_test.go:189: Todos los 10 trabajos completados
BenchmarkCarga_DobleCoches_Incidencias-8               1        58517419962 ns/op
```

TABLA RESULTADOS:

Escenario	Núm. Coches	Incidencias	Mecánicos (M-E-C)	Tiempo (s)	Incremento
Normal	5	Mecánica	1-1-1	31,12	-
Doble	10	Mecánica	1-1-1	58,51	1,88

EXPLICACIÓN DE RESULTADOS:

1. Cuando duplicamos la carga, el tiempo total aumenta casi el doble, exactamente en un factor de 1.88. Esto era de esperar ya que el número de coches aumento, sin embargo, lo más normal hubiese sido que al duplicar los coches, también se duplicara el tiempo. Al no suceder esto podemos decir que el **sistema funciona de forma eficiente**.
2. Solo tenemos un único mecánico de la especialidad Mecánica, que es el único que puede atender estas reparaciones. Cuando llegan los 10 coches, solo 1 puede ser atendido inmediatamente, aunque cada mecánico genere dos plazas. Esto genera un **cuello de botella**, lo cual con un gran número de coches se ralentizaría mucho el tiempo, esta es la causa principal del aumento de tiempo.
3. **Funcionamiento de la cola de espera:** Los 9 coches restantes se envían a la cola de espera del Gestor Central. El hecho de que el tiempo aumente en menos de 2x demuestra que la gestión de esta cola (a través de channels y del gestorDeTaller) es muy ligera. El sistema no pierde tiempo ni genera una sobrecarga significativa al tener que gestionarlo.
4. **Flujo del Programa:** El Gestor Central actúa como un monitor. Está siempre escuchando en el canal de notificaciones y, en cuanto el único mecánico de Mecánica libera su goroutine, el Gestor revisa inmediatamente la cola y le asigna el siguiente trabajo, manteniendo el recurso ocupado casi al 100% sin introducir latencia innecesaria.

2.3 - Test 2: **Comparativa cuando duplicamos la plantilla de 3 mecanicos a 6**.

TABLA RESULTADOS:

```
BenchmarkPlantilla_3Mecanicos
    taller_test.go:89: Taller iniciado con 3 mecánicos (total plazas: 6)
    taller_test.go:168: Carga de 3 coches (electrica) lanzada
    taller_test.go:168: Carga de 3 coches (mecanica) lanzada
    taller_test.go:168: Carga de 3 coches (carroceria) lanzada
    taller_test.go:181: Completados: 5/9
    taller_test.go:189: Todos los 9 trabajos completados
BenchmarkPlantilla_3Mecanicos-8                     1        35624400792 ns/op
BenchmarkPlantilla_6Mecanicos
    taller_test.go:89: Taller iniciado con 6 mecánicos (total plazas: 12)
    taller_test.go:168: Carga de 3 coches (electrica) lanzada
    taller_test.go:168: Carga de 3 coches (mecanica) lanzada
    taller_test.go:168: Carga de 3 coches (carroceria) lanzada
    taller_test.go:181: Completados: 5/9
    taller_test.go:189: Todos los 9 trabajos completados
BenchmarkPlantilla_6Mecanicos-8                     1        23155643020 ns/op
```

Escenario	Num. coches	Incidencias	Mecánicos (M-E-C)	Tiempo (s)	Reducción
Normal	9	3 cada tipo	1-1-1	35,63	-
Doble	9	3 cada tipo	2-2-2	23,15	35,05%

EXPLICACIÓN DE RESULTADOS:

1. Al duplicar la plantilla de 3 a 6 mecánicos, el **tiempo total se redujo** en aproximadamente un 35%, teóricamente nos deberíamos haber acercado a una reducción del 50%
2. **Apertura de Concurrencia:** En el escenario base (1:1:1), el sistema se congestiona rápidamente porque hay 3 trabajos de cada tipo compitiendo por 1 solo recurso. Por ejemplo, hay 3 trabajos de carrocería (11s de atención) luchando por 1 mecánico. Esto crea una cola de espera muy larga que domina el tiempo total.
3. **Paralelismo de Goroutines:** Al pasar a 2 mecánicos por especialidad (2:2:2), el Gestor Central puede asignar hasta 2 trabajos de cada tipo simultáneamente.
4. **Flujo del Programa:** La capacidad del sistema para abrir más goroutines de mecánicos y asignar trabajo en paralelo es lo que logra la reducción. Se eliminan los cuellos de botella en cada especialidad, y el tiempo se acerca al tiempo necesario para completar la reparación más larga, que son los 3 trabajos de carrocería con 2 mecánicos, más el tiempo de gestión del programa.

2.4 - Test 3: **Comparativa cuando tenemos tres mecanicos de especialidad mecanica por cada uno de carroceria y uno de electrica. Y cuando tenemos uno mecánicos de especialidad mecánica por cada tres de eléctrica y tres de carrocería.**

```
BenchmarkPlantilla_3M_1E_1C
    taller_test.go:89: Taller iniciado con 5 mecánicos (total plazas: 10)
    taller_test.go:168: Carga de 4 coches (mecanica) lanzada
    taller_test.go:168: Carga de 2 coches (electrica) lanzada
    taller_test.go:168: Carga de 2 coches (carroceria) lanzada
    taller_test.go:181: Completados: 5/8
    taller_test.go:189: Todos los 8 trabajos completados
BenchmarkPlantilla_3M_1E_1C-8                  1        22227014893 ns/op
BenchmarkPlantilla_1M_3E_3C
    taller_test.go:89: Taller iniciado con 7 mecánicos (total plazas: 14)
    taller_test.go:168: Carga de 4 coches (mecanica) lanzada
    taller_test.go:168: Carga de 2 coches (electrica) lanzada
    taller_test.go:168: Carga de 2 coches (carroceria) lanzada
    taller_test.go:181: Completados: 5/8
    taller_test.go:189: Todos los 8 trabajos completados
BenchmarkPlantilla_1M_3E_3C-8                  1        30061137086 ns/op
```

TABLA RESULTADOS:

Escenario (Num. Mecánicos)	Num. Coches	Incidencias (M-E-C)	Mecánicos (M-E-C)	Tiempo (s)

+Mecánica	8	4-2-2	3-1-1	22,22
+Eléctrica +Carrocería	8	4-2-2	1-3-3	30,06

EXPLICACIÓN DE RESULTADOS:

1. En este caso uso otra variable global de 8 coches para poder repartir las incidencias de la siguiente forma: Cuatro para mecánica, dos para eléctrica y otras dos para carrocería. **Intentando simular un escenario real** donde las incidencias más rápidas (mecánicas) son más habituales que las complejas.
2. El escenario que contiene más mecánicos de la incidencia más frecuente (3M:1E:1C) fue el más rápido (22.23s), superando al escenario con más mecánicos en total (7 mecánicos) y duración de 30,06 segundos.
3. Estrategia Eficiente: El sistema funciona mejor cuando las **tareas más frecuentes son atendidas lo más rápido posible**. Al dedicar 3 goroutines de Mecánica, el Gestor puede vaciar las peticiones de Mecánica casi instantáneamente, eliminando el volumen de la cola.

1. **Flujo del Programa:** Esto minimiza la cantidad de mensajes que el Gestor Central tiene que procesar en la cola y reduce el tiempo que las tareas Mecánicas más cortas pasan esperando, optimizando el throughput total del taller. El tiempo que las reparaciones largas esperan es compensado por la velocidad con la que se procesa el gran volumen de tareas más cortas.

2.5 - Test unitarios

```
RUN   TestCreacionMecanico
taller_test.go:80: Taller iniciado con 1 mecánicos (total plazas: 2)
taller_test.go:351: Test pasado: 1 mecánico creado correctamente
PASS: TestCreacionMecanico (0.20s)
RUN   TestPlazasPorMecanico
taller_test.go:80: Taller iniciado con 3 mecánicos (total plazas: 6)
taller_test.go:381: Test pasado: 3 mecánicos generan 6 plazas (2 por mecánico)
PASS: TestPlazasPorMecanico (0.50s)
RUN   TestEscalamiento15Segundos
taller_test.go:80: Taller iniciado con 1 mecánicos (total plazas: 2)
taller_test.go:178: Todos los 1 trabajos completados
taller_test.go:428: Test pasado: Incidencia cerrada tras 32.08s (prioridad: escalada)
PASS: TestEscalamiento15Segundos (32.30s)
RUN   TestColaDeEspera
taller_test.go:80: Taller iniciado con 1 mecánicos (total plazas: 2)
taller_test.go:170: Completados: 5/5
taller_test.go:178: Todos los 5 trabajos completados
taller_test.go:463: Test pasado: 5 incidencias procesadas con 1 solo mecánico (cola funcionó)
PASS: TestColaDeEspera (33.78s)
```

Estos tests se hicieron antes de los Benchmarks para comprobar el funcionamiento de los apartados más complejos. Pequeña descripción del propósito de cada test.

1. **TestCreacionMecanico**: Confirma que el proceso de inicio de un mecánico es exitoso. Verifica que el GestorCentral recibe la petición de creación, lanza la nueva goroutine del mecánico y registra correctamente la capacidad del Taller.
2. **TestPlazasPorMecanico**: Confirma que el cálculo de la capacidad del Taller es correcto. Asegura que la lógica de conteo dentro del GestorCentral asigna las 2 plazas por cada mecánico activo, necesario para ver si hay hueco disponible.
3. **TestEscalamiento15Segundos**: Confirma la implementación de la Regla de Prioridad una vez superado el tiempo total de 15 segundos. Verifica que el GestorCentral detecta esta condición (prioridad escalada en la imagen) y activa la lógica de reasignación forzada o contratación de emergencia para cerrar la incidencia, cumpliendo con la regla de prioridad.
4. **TestColaDeEspera**: Confirma la funcionalidad de la cola ilimitada. Se lanzan 5 trabajos de Mecánica cuando solo hay 1 Mecánico de esa especialidad. El test demuestra que los 4 trabajos excedentes se añaden a la cola y son procesados secuencialmente por el único recurso, probando que el sistema no colapsa.

3 - Código fuente del sistema (Repositorio del proyecto).

Link: https://github.com/atejero2021/DistribuidosP2