Práctica 2

Reingeniería software con *código heredado* (*legacy code)*

2.1. Programación y objetivos

Esta práctica constará de dos sesiones. Tendrá una puntuación en la nota final de prácticas de 2 puntos sobre 10. Como documentación, se debe entregar un pdf con:

1. al menos un diagrama UML que muestre la arquitectura del sistema o diseño de alto nivel (diagrama de componentes, diagrama de paquetes, diagrama de estados, diagrama de clases ...),
2. un diagrama de clases que muestre las clases de diseño,
3. una descripción en lenguaje natural de los nuevos requisitos funcionales del software, y
4. una descripción de las tareas de refactorización llevadas a cabo.

2.1.1. Objetivos generales de la práctica

1. Adquirir destreza en la reingeniería software y el mantenimiento adaptativo de código
2. Adquirir destreza en la refactorización de código
3. Profundizar en el uso de estilos arquitectónicos y patrones de diseño para mejorar la reusabilidad del código

4. Incorporar software externo

2.1.2. Planificación y tareas específicas

Se espera del estudiante que en la primera sesión de esta práctica:

1. Entienda bien lo que se pide.
2. Especifique el sistema a implementar usando diagramas UML junto con su compañero de prácticas y preguntando al profesor y a otros compañeros sobre posibles decisiones a tomar.
3. Empiece a implementar

En la segunda sesión de la práctica debe continuar la codificación, la cual se terminará en tiempo de trabajo fuera de la sesión y antes del inicio de la práctica 3.

Sesión	**Semana**	**Tareas**
S1	9-14 abril	Selección de patrones a utilizar, diseño del sistema con diagramas Inicio de la codificación (modelo)
S2	16-21 abril	Desarrollo de las páginas o pantallas de la vista de usuario Depuración de código

2.2. Criterios de evaluación

Para poder corregir la práctica será obligatorio cumplir con todos y cada uno de los siguientes requisitos:

- Capacidad demostrada de trabajo en equipo (reparto equitativo de tareas)
- Implementación completa y verificabilidad (sin errores de ejecución)
- Implementación en flutter
- Usar como software heredado el desarrollado en la sesión 3ª de la práctica 1

Para la evaluación se tendrán en cuenta los siguientes criterios:

- Nivel de modificación del software alcanzado (debe tratarse de un producto completamente remodelado y con mucha más funcionalidad)
- Uso de nuevos estilos arquitectónicos y patrones de diseño
- Tareas de refactorización realizadas
- Software externo incorporado, con indicación de licencia y autoría

Se valorarán además otros criterios de evaluación:

- Utilidad de la app
- Calidad de la GUI

2.3. Plazos de entrega y presentación de la práctica

Esta práctica será subida a PRADO en una tarea que terminará justo antes del inicio de la práctica 3 (a las 15:30 horas del día de la primera sesión de la misma). La práctica completa será presentada mediante una entrevista con el profesor de prácticas.

2.4. Descripción general de los requisitos de la práctica

Se debe desarrollar una aplicación que incorpore la funcionalidad del ejercicio de la sesión 3ª de la práctica 1, añadiendo suficiente funcionalidad para que el mantenimiento adaptativo no sea posible, sino que se trate de un producto significativamente distinto (reingeniería software) con código heredado, que use nuevos estilos arquitectónicos y patrones de diseño[1]. Esta aplicación será codificada en flutter.

Como sugerencia para aumentar significativamente la funcionalidad, se puede considerar la posibilidad de programar simuladores.

2.5. Ejemplo: Simulador de control automático para la conducción de un vehículo (SCACV)

Se incluye este ejemplo de especificación de un sistema que cumple con lo pedido en cuanto a nivel de reingeniería software. Como ejemplo del resultado puede descargarse

[1] Los criterios de selección y forma de aplicación de los mismos serán tenidos en cuenta para la evaluación.

un applet que implementa un problema similar[2]

2.5.1. Requisitos funcionales

Subsistema de control de velocidad

- Se pretende desarrollar un sistema de control automático de la velocidad para automóviles controlable mediante una palanca de cuatro posiciones y el pedal del freno, existiendo también el pedal del acelerador. Inicialmente el vehículo funcionará en modo manual, y el conductor usará los pedales del freno y del acelerador. Es importante que la velocidad de "crucero" del vehículo se mantenga, una vez alcanzada, ya que una velocidad superior a la necesaria implica un desperdicio de potencia consumida por el motor (y actualmente una retirada de puntos del carnet), por lo cual ha de mantenerse a toda costa. Para ello, el conductor pondrá la palanca del SCACV en la posición acelerar a la vez que suelta el pedal de aceleración y una vez alcanza la velocidad deseada, cambiará la palanca del SCACV a la posición de modo automático (estado "manteniendo"), cuando el vehículo alcance la velocidad deseada. Así, el vehículo mantendrá la velocidad de crucero. Para ello debe tenerse en cuenta el autómata de la Figura 2.1.

- Añadir funcionalidad para que cuando el conductor accione un mando del vehículo (palanca del SCACV, pedales, llave de contacto) la respuesta del sistema sea automática (síncrona) y no se espere a que, de forma asíncrona mediante una hebra que observe el estado de los mandos, se atienda a la petición del conductor. Los cambios en los mandos ocasionan transiciones internas en el control del sistema que hacen cambiar su estado. En todo caso, para mantener una petición del conductor en el tiempo, que se traduce en un estado del vehículo (por ejemplo, acelerar, bien usando la palanca correspondiente del SCACV o bien pisando el acelerador) hasta que el conductor no de una nueva orden, el subsistema de control de velocidad debe detectar de forma periódica (hebra) el estado en el que se encuentra el vehículo y actuar en consecuencia.

Subsistema de monitorización En el ejemplo de ejercicio 3 de la práctica 1, se monitorizaban: velocidad angular (RPM), velocidad lineal (km/h), distancia recorrida desde el inicio de la aplicación (km), distancia parcial recorrida (km, desde la última vez que se arrancó el motor). Ahora deberá ser ampliado para añadir otro requisito funcional:

[2]Para ejecutar el applet puede usarse el programa appletviewer que forma parte del Java Development Kit (JDK).

Monitorización del consumo. Para ello será necesario guardar cada vez que se actualiza la monitorización: (1) el instante en el que se realiza y (2) las vueltas (revoluciones) producidas por el eje desde la última actualización (a partir del instante anterior y la velocidad anterior). Para cada consumible deberán también almacenarse el tiempo y las rotaciones totales del eje desde el último cambio/recarga, y además:

- *Consumo del combustible promedio*: debe almacenarse el nivel de combustible alcanzado cuando se reposta o se inicia la aplicación (asignado de forma aleatoria dentro de un rango). El consumo debe calcularse en función del número de revoluciones del eje desde la última vez que se actualizó (por ejemplo, puede considerarse que el consumo en litros es de $rot \times rot \times 5 \times 10^{-10}$, siendo rot el número de revoluciones (vueltas) del eje desde la última vez que se actualizó.
- *Consumo aceite, pastillas de freno y revisión general*: El sistema proporciona notificaciones de mantenimiento a sus usuarios: cada 5×10^6 rotaciones del eje aparecerá un mensaje en la pantalla que le avisará de la necesidad de cambiar el aceite del motor; cada 10^8 rotaciones para cambio del pastillas y cada 10^9 rotaciones para efectuar una revisión general del sistema. Además se incluirán 3 botones para hacer el cambio de aceite, de pastillas de freno y revisión general respectivamente. Esos botones requerirán para estar activos que el motor del vehículo esté apagado y el vehículo parado, de forma que sea accesible para su mantenimiento. Las acciones que deben realizarse al ser pulsados serán las de actualizar, respectivamente (1) número de rotaciones acumuladas en la fecha del ultimo engrase, (2) número de rotaciones en la fecha del cambio pastillas de freno y (3) número de rotaciones en la última fecha de revisión general.

2.5.2. Requisitos no funcionales

- Se desarrollará una app programada en flutter
- Se usarán componentes lo más parecidos posible a los elementos a los que representan. Por ejemplo, el velocímetro debería aparecer como un círculo o semicírculo con una aguja señalando la velocidad actual.

2.5.3. Descripción del SCACV y resto de dispositivos de control

El subsistema de control automático de velocidad del vehículo es controlado inicialmente por el conductor que puede poner la palanca en alguna de sus 4 posiciones conmutables: *acelerar*, *apagado*, *reiniciar* y *mantener*.

- *Acelerar*: manteniendo la palanca en esta posición la velocidad del motor se incrementa continuamente.
- *Mantener* o *modo automático*: la velocidad actual del vehículo es memorizada por el sistema y el vehículo mantiene esta velocidad de forma constante. Hay que tener en cuenta que sólo si se apaga y se vuelve a encender el motor se cancela la última velocidad memorizada.
- *Reiniciar*: el vehículo recupera la última velocidad de crucero almacenada (la que tenía la última vez que estuvo en *modo automático*). Una vez alcanzada, se modificará automáticamente la palanca del SCACV a la posición de *Mantener*.
- *Apagado*: se vuelve al modo de control manual de la velocidad del vehículo. El conductor ha de poder seleccionar esta posición si el SCACV estaba activado (cuando la palanca estaba en posición *Modo automático*) o estaba en posición *Acelerar*. También se pone automáticamente en esta posición si la palanca estaba en posición *Modo automático* o en posición *Reiniciar* y pisa el freno.

Además de este subsistema, el conductor también dispone de los siguientes controles relacionados con la marcha del vehículo:

- *Encendido/apagado del motor*: Para arrancar-apagar el motor al inicio-fin de la ejecución del programa respectivamente.
- *Frenar/soltar freno*: Frena o deja de frenar el vehículo respectivamente. El frenado del vehículo, en el caso de estar activado el *Modo automático* o el de *Reiniciar* del SCACV, provocará automáticamente el cambio de la palanca del SCACV a la posición de *Apagado* (modo de control manual de la velocidad del vehículo).
- *Acelerar/soltar acelerador (pedal)*: Acelera o deja de acelerar el vehículo respectivamente.

2.5.4. Autómata que reproduce los estados del vehículo

En la Figura 2.1 se muestra un autómata con los 5 estados (nodos) del vehículo:

- motor **apagado** o **no arrancado** (nodo transparente, vehículo parado o en desaceleración con motor apagado)
- **motor arrancado** (nodos coloreados)

- **encendido** (vehículo parado o en desaceleración por rozamiento con motor arrancado)
- **frenando**
- **acelerando**
- **manteniendo**: este es el único estado que no se alcanza con la conducción manual y por tanto que no se representó en el ejercicio 3ª de la práctica 1

2.5.5. Descripción de los estados del diagrama y su relación con los estados del SCACV

Se puede observar en la figura que los mismos estados se repiten con distintos colores, lo que significa que el estado del SCACV (posición de la palanca y si la velocidad de crucero ha sido almacenada desde la última vez que se arrancó el vehículo) es distinto según la Tabla 2.1:

		Velocidad almacenada	
		No	Sí
Posición de la palanca	Apagado		
	Reiniciar	-	
	Acelerar		
	Mantener	-	

Tabla 2.1: Colores usados en el autómata para representar los distintos estados del SCACV.

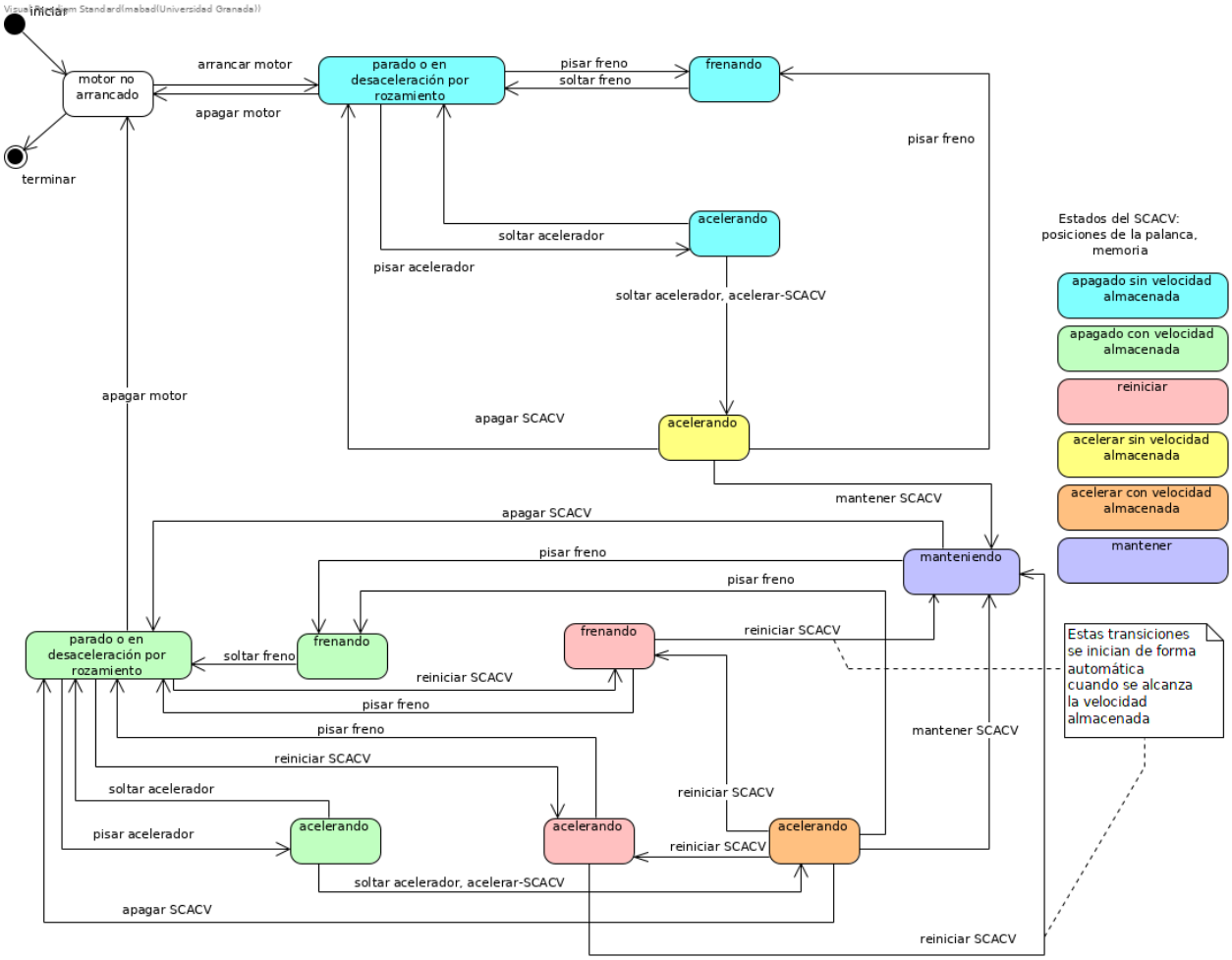

Figura 2.1: Autómata finito que representa los estados del vehículo y el SCACV y las transiciones entre ellos.