ED07_1 - Diagramas de comportamiento

En esta práctica se describe un caso práctico en el que tendrás que definir diferentes diagramas de comportamiento que permitan desarrollar una aplicación para la gestión de clases deportivas en un gimnasio. Se espera que seas capaz de identificar los actores, casos de uso y las interacciones entre ellos, así como los estados de los objetos en el sistema.

1. Objetivos

Diseñar y analizar el comportamiento de un sistema informático destinado a gestionar las **reservas de clases deportivas en un gimnasio**. Se trabajarán tres tipos de diagramas UML:

- **Diagrama de casos de uso** del sistema completo
    - **Descripción detallada** de uno de los casos de uso principales
- **Diagrama de secuencia** asociado a alguno de los casos de uso
- **Diagrama de estados** de uno de los elementos clave del sistema

2. Enunciado

El gimnasio "**TeisSportLife**" quiere implantar un sistema online que permita a sus socios reservar plaza en clases dirigidas (como spinning, yoga, etc.). Este sistema debe permitir:

- Consultar el **calendario de clases** por parte de los socios y entrenadores
- **Reservar una clase** a un socio si hay plazas disponibles
- **Cancelar una reserva**
- Controlar la **asistencia** a los entrenamientos por parte de los entrenadores
- Los entrenadores deben poder **crear nuevas clases** y **consultar asistentes**
- Los administradores del sistema podrán añadir nuevos tipos de clases así como el aforo máximo de cada una de ellas.

El sistema tiene tres tipos de usuarios:

- **Socio**: puede consultar el calendario, reservar clases y cancelar reservas.
- **Entrenador**: puede crear clases y consultar quién se ha apuntado.
- **Administrador**: gestiona automáticamente el aforo y los tipos de clases.

3. Actividades solicitadas

3.1. Diagrama de casos de uso

Realiza un **diagrama de casos de uso** del sistema que incluya, al menos, los casos de uso que consideres en base a la descripción del sistema. Incluye a los actores y relaciones entre casos de uso si procede (`<<include>>`, `<<extend>>`).

Utiliza draw.io para crear el diagrama de casos de uso y exporta el resultado como imagen.

3.2. Descripción detallada de un caso de uso

Redacta la **descripción textual** del caso de uso **"Reservar clase"**, incluyendo los siguientes apartados:

- Nombre del caso de uso
- Código
- Actor principal
- Breve descripción
- Precondiciones
- Flujo principal de eventos (paso a paso)
- Flujos alternativos o excepciones (por ejemplo: si no hay plazas)
- Postcondiciones

Puedes utilizar la tabla proporcionada en los apuntes como plantilla.

Elemento	**Descripción**
Nombre del caso de uso	
Identificador	
Descripción	
Actor(es)	
Precondiciones	
Curso normal	
Curso alternativo 1	(si procede)
Postcondiciones	

3.3. Diagrama de secuencia

Dibuja un **diagrama de secuencia** que represente el flujo de interacción entre los objetos/participantes durante la ejecución del caso de uso **"Reservar clase"**.

Para desarrollar correctamente el **diagrama de secuencia**, se debe tener en cuenta la interacción entre los siguientes **actores y componentes del sistema**:

Actor principal:

- **Socio**: es el usuario final que desea realizar la reserva. Interactúa a través de una interfaz del sistema, como una app móvil o una página web.

Módulos y objetos participantes:

1. **IU (Interfaz de Usuario)**
   Módulo que presenta la información al usuario y recoge sus acciones. Encargado de iniciar las solicitudes hacia el backend.
   Ejemplo: pantalla de calendario, formulario de reserva.

2. **GestorReservas (Controlador o Servicio de Dominio)**
   Componente central que gestiona la lógica de negocio de las reservas. Valida las peticiones, consulta la disponibilidad y ejecuta la reserva si es posible.

3. **BD Clases (Base de Datos o Repositorio)**
   Fuente de datos donde se almacenan las clases disponibles, el número de plazas, las reservas realizadas, etc.
   Puede modelarse como una base de datos o como un objeto `RepositorioClases`.

Flujo básico del escenario:

1. El **socio** accede a la aplicación y selecciona una clase desde el calendario.
2. La **IU** lanza una solicitud al **Gestor de Reservas** para intentar reservar.
3. El **Gestor** consulta la base de datos para verificar si hay plazas disponibles.
4. Si las hay, el **Gestor** registra la reserva y actualiza el número de plazas.
5. Finalmente, se notifica el resultado al usuario (reserva confirmada o error si no hay plazas).

Consejos para modelar:

- Representar cada **objeto** como una línea de vida en el diagrama.
- Incluir mensajes sincrónicos (`->>`) entre módulos.
- Mostrar claramente condiciones o bifurcaciones (por ejemplo, si hay plazas o no).
- Se puede incluir el **loop de selección de clase** si se quiere mostrar que el usuario puede explorar varias opciones antes de reservar.

Puedes utilizar **draw.io** o **mermaid** para crear el diagrama de secuencia y exportar el resultado como imagen.

3.4. Diagrama de estados

Una reserva de clase en el sistema puede pasar por distintas situaciones a lo largo de su ciclo de vida. Inicialmente, cuando un usuario realiza una reserva, esta queda **registrada** en el sistema. A partir de ese momento, la reserva puede seguir distintas trayectorias según las acciones del usuario o los eventos del sistema:

- Si el socio decide no asistir y **cancela** con antelación, la **reserva se anula** y no se tiene en cuenta para el control de asistencia.
- Si el socio mantiene la reserva, y llega el momento de la clase, el sistema puede **registrar la asistencia automáticamente** (por ejemplo, escaneando un código QR) o bien **marcar la ausencia** si no se presenta.
- En caso de que la clase se imparta con normalidad y el socio asista, la reserva se considera **completada** con éxito.
- Por otro lado, si por algún motivo la clase es **suspendida** (por el entrenador o el sistema), la reserva puede **considerarse invalidada** o cancelada por fuerza mayor.
- Una vez finalizada la clase o cancelada la reserva, esta ya no puede modificarse, aunque puede **conservarse** como histórico.

Este comportamiento puede representarse mediante un **diagrama de estados**, donde se muestran las distintas situaciones posibles por las que puede pasar una reserva y los eventos que provocan el paso de una situación a otra (como asistir, cancelar, modificar, etc.). Crea un **diagrama de estados** que muestre los distintos estados posibles de una **reserva de clase**, así como los eventos que provocan la transición entre ellos.

Puedes usar draw.io o mermaid para crear el diagrama de estados y exportar el resultado como imagen.

4. Entregable

Deberás entregar un archivo comprimido (zip) con el nombre **apellidos_nombre_ed07_1.zip** que contenga:

- El diagrama de casos de uso en formato imagen (png o jpg).
- El diagrama de secuencia en formato imagen (png o jpg).
- El diagrama de estados en formato imagen (png o jpg).
- Un documento en formato .pdf con el nombre **apellidos_nombre_ED07_1.pdf** que contenga toda la información que consideres relevante para la correcta comprensión de los diagramas y el caso de uso "Reservar clase". En caso de que utilices **mermaid** para la realización de los diagramas de secuencia y estados, deberás incluir el código fuente de los mismos en el documento .pdf. Puedes usar el editor de mermaid en mermaid.live para generar el código y luego exportar el diagrama como imagen.
  Asegúrate de que el código fuente esté correctamente formateado y sea legible. En caso de que utilices **draw.io**, deberás incluir el archivo .drawio de cada diagrama creado.

5. Criterios de evaluación

- **Diagrama de casos de uso**: 2 puntos
  - Correcta identificación de actores y casos de uso.
  - Uso adecuado de relaciones entre casos de uso (incluyendo `<<include>>` y `<<extend>>`).
  - Claridad y legibilidad del diagrama.
- **Descripción del caso de uso "Reservar clase"**: 2 puntos
  - Correcta redacción de la descripción textual del caso de uso.
  - Inclusión de todos los apartados solicitados (nombre, código, actor principal, descripción, precondiciones, flujo principal, flujos alternativos y postcondiciones).
  - Claridad y coherencia en la redacción.
- **Diagrama de secuencia**: 3 puntos
  - Correcta representación de la interacción entre los objetos/participantes.
  - Uso adecuado de mensajes y activaciones.
  - Claridad y legibilidad del diagrama.
- **Diagrama de estados**: 3 puntos
  - Correcta representación de los estados y transiciones.
  - Uso adecuado de eventos y condiciones.
  - Claridad y legibilidad del diagrama.