PRÁCTICA 9: LO QUE QUEDA DEL DÍA[1]

1. Introducción

Toda definición de Ingeniería que se precie incluye la idea de que los sistemas que se construyen deben servir para satisfacer las necesidades de una o más personas. Los modelos de dominio ayudan a capturar la complejidad de un determinado ámbito de la vida real, pero por sí solos no sirven satisfacen las necesidades de ningún usuario. Por tanto, para que estos modelos sean útiles, necesitamos conectarlos con una *capa de presentación* que permita realizar diferentes acciones destinadas a satisfacer las necesidades de algún usuario.

Estas acciones requerirán obtener, introducir o modificar datos del modelo de dominio, y deben, por tanto, realizarse teniendo en cuenta las reglas del negocio. Además, durante el transcurso de estas acciones, puede ser necesario obtener objetos del almacén persistente o salvar datos en él.

Tanto para facilitar la evolución del modelo de dominio como para evitar que la capa de presentación tenga que lidiar con las peculiaridades de dicho modelo, la práctica habitual es crear una fachada entre la capa de presentación y el modelo de dominio que actúe como *capa o fachada de servicio*. Dicha capa de servicio tiene como responsabilidad: (1) recoger las peticiones de la capa de presentación; (2) recuperar objetos en la capa de persistencia que fuese necesario; (3) redirigir las peticiones recibidas al objeto y objetos del modelo de dominio que correspondan; (4) almacenar los objetos que se hayan modificado y; (5) finalmente, devolver los resultados que correspondan a la capa de presentación.

Además de las responsabilidades anteriormente descritas, la capa de servicio debe enfrentarse a un problema adicional: la transferencia de los datos entre negocio y presentación. Los modelos de dominio constituyen una red de elementos fuertemente relacionados entre sí. Por lo tanto, para enviar un objeto a la capa de presentación, si queremos preservar las relaciones que dicho objeto posee con otros objetos, necesitaríamos mandar esos objetos relacionados. A continuación, nos encontraríamos con la necesidad de enviar los objetos relacionados con los objetos relacionados, y así sucesivamente. Como resultado, acabaríamos transfiriendo el modelo de dominio entero a la capa de persistencia. Si finalmente la capa de persistencia sólo utilizase una pequeña fracción de dicho modelo de dominio, estaríamos trabajando de manera ineficiente.

Para evitar este problema suelen utilizarse diferentes estrategias, siendo una de las más populares la utilización de *Data Transfer Objects*. Un *Data Transfer Object* es un objeto o conjunto de objetos que contienen la información, procedente de otro objeto o conjunto de objetos más complejo, justa y necesaria para la realización de una tarea determinada. En el caso de la capa de presentación, dicha tarea suele ser realizar una determinada visualización.

El objetivo general de esta práctica es aprender a desarrollar capas de servicio, aplicando para ello los patrones de diseño pertinentes.

[1] Por *Lo que queda del día (James Ivory, 1993)*, película donde el personal de servicio es protagonista.

2. Objetivos

Los objetivos concretos que se persiguen con la realización de esta práctica son:

1. Aprender a especificar la interfaz de la capa de negocio de una aplicación empresarial.
2. Aprender a crear *Data Transfers Objects* que permitan transferir objetos de negocio a una capa de presentación.
3. Aprender a implementar capas de servicio mediante *Transaction Scripts*.
4. Aprender a implementar capas de servicio mediante la utilización de un *Domain Model*.
5. Comprender cómo se integran las diferentes capas de una aplicación empresarial.

3. Actividades

El alumno, para alcanzar los objetivos perseguidos, deberá completar con éxito las siguientes actividades sobre el sistema *Polaflix*, descrito en la práctica anterior:

1. Crear un objeto DTO (*Data Transfer Object*) para la interfaces de las Figuras 1 a 4, proporcionadas en el Apéndice A de la práctica anterior.
2. Desarrollar un *Assembler* para el DTO asociado a la interfaz de la Figura 2.
3. Especificar las operaciones que se considere necesario implementar en la capa de servicio que de soporte al sistema *Polaflix*.
4. Implementar la operación de negocio que agrega una serie al espacio personal de un usuario mediante el patrón *Transaction Script*.
5. Implementar la operación de negocio que agrega un cargo a la factura mensual del usuario mediante la utilización del patrón *Domain Model*.

Para las actividades 4 y 5, en caso de que necesiten utilizarse *DataMappers* que no se encuentren actualmente implementados, éstos podrán simularse mediante la creación de *mocks*.

4. Criterios de Evaluación y Aclaraciones

La práctica se entregará a través de la plataforma *moodle* siguiendo las instrucciones en ella proporcionadas. La práctica se realizará en Java.

La práctica se calificará atendiendo a los siguientes criterios de evaluación:

(1) Corrección de los DTOs creados (2 puntos).
(2) Corrección del *assembler* desarrollado (1 punto).
(3) Corrección de la especificación de la capa de negocio (2 puntos).
(4) Corrección del *Transaction Script* implementado (2 puntos).
(5) Corrección de la operación de negocio basada el *Domain Model* (2 puntos).
(6) Cumplimiento de la *Guía de Buenas Prácticas en Programación* (1 puntos).

Pablo Sánchez Barreiro