Trabajo Práctico 1 — Java

[7507/9502] Algoritmos y Programación III
Curso 1
Primer cuatrimestre de 2018

Alumno:	Taborda, Gastón
Número de padrón:	96491
Email:	gastonlucastaborda@gmail.com

Índice

1. Introducción

El presente informe reune la documentación de la solución del primer trabajo práctico de la materia Algoritmos y Programación III que consiste en desarrollar una aplicación de un sistema de una agencia de viajes en Java utilizando los conceptos del paradigma de la orientación a objetos vistos hasta ahora en el curso.

2. Supuestos

-Las fechas de los vuelos nunca se almacenan, ya que se valida en el momento que sea una fecha límite del viaje. En cambio, las fechas de las estadías se almacenan en la clase Hotel para que esta misma calcule la duración de la estadía y además el costo.

-El nombre de la ciudad se usa únicamente como key en un diccionario de ciudades para agilizar la búsqueda de las mismas, por lo que no se almacena dentro de la clase Ciudad.

3. Modelo de dominio

A continuación se realizará una breve descripción de las clases involucradas en el desarrollo del trabajo.

Algotrip: se encarga de almacenar diccionarios para agilizar la búsqueda de objetos. Existe uno para cada clase Viaje, Ciudad y Paquete.También se encarga de enviar mensajes a los objetos que contiene para que realicen sus responsabilidades.

Vuelo: se encarga de calcular su costo y alterarlo según los adicionales agregados. Posee una instancia de VueloNacional o VueloInternacional según corresponda. Estas clases implementan la interfaz TipoDeVuelo y su método obtenerCosto(), el cual varía en VueloInternacional según los impuestos que se aplican.

Viaje: guarda los vuelos, estadías y paquetes del viaje correspondiente en una lista de cada objeto. Sus responsabilidades son calcular su costo y su duración, además de delegar a los objetos que contiene para que realicen los mismos procedimientos y sumen adicionales.

Hotel: guarda el precio de cada noche que se hospeda en el hotel y se encarga de calcular la duración de la estadía y el costo de la misma.

Ciudad: guarda su nombre, país latitud y longitud. Sus responsabilidades son calcular la distancia hacia otra ciudad y validar si se encuentra en el mismo país que otra ciudad.

Paquete: es la clase hija de Viaje, por lo que hereda de ella. Comparte comportamiento con Viaje, pero difiere principalmente en el método para calcular el costo, ya que aplica un descuento a los vuelos y las estadías.

4. Diagramas de clase

A continuación se incluyen 3 diagramas de secuencia. El primero detalla la interacción general de los objetos, alrededor de la clase AlgoTrip. El siguiente detalla la relación entre Vuelo, la interfaz TipoDeVuelo y las clases VueloNacional y VueloInternacional. Por último, se detalla la relación entre Viaje y las listas que contiene de los objetos Vuelo, Hotel y Paquete.

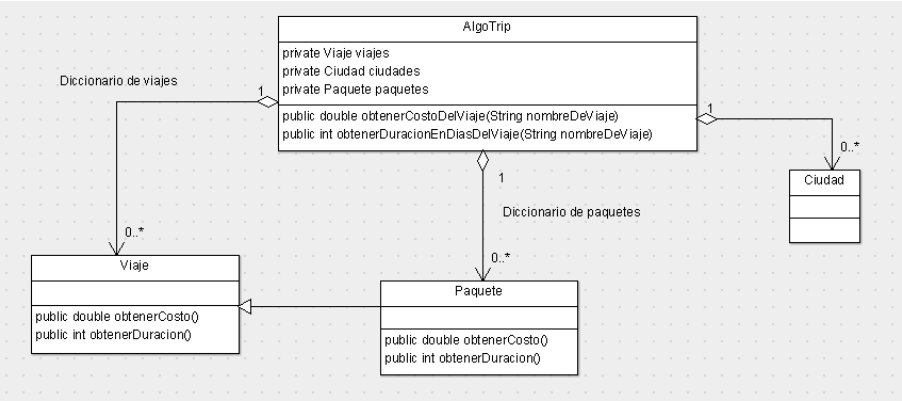

Figura 1: Diagrama General .

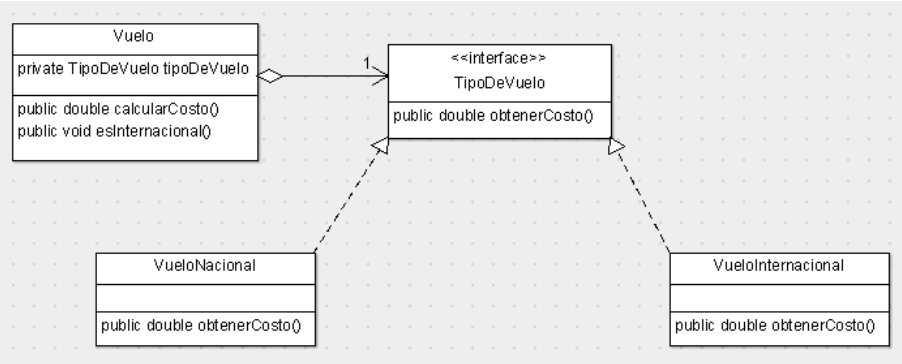

Figura 2: Diagrama Vuelo.

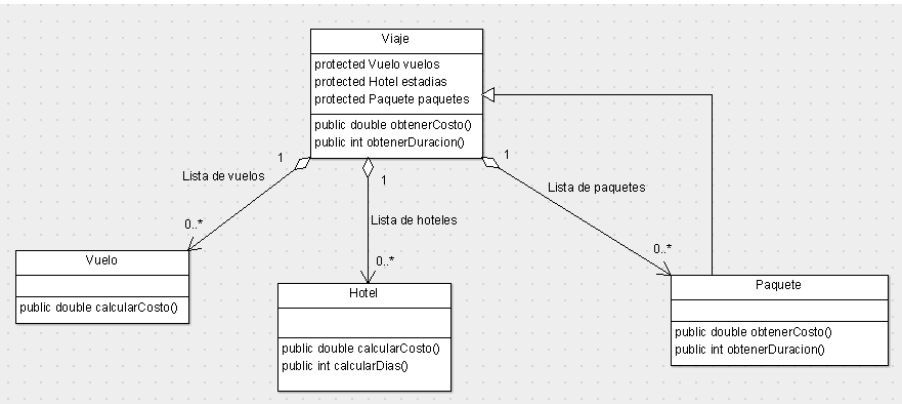

Figura 3: Diagrama de clase Viaje.PNG.

5. Detalles de implementación

5.1. Método Vuelo.agregarAsistenciaAlViajero

El método modifica el atributo asistenciaAlViajero de la clase Vuelo. Este atributo es un int que toma valor 0 inicialmente y, al agregarle la asistencia al viajero, se cambia a 1. De esta manera,

al realizar la cuenta se toman en cuenta los recargos que la asistencia al viajero aplica.

5.2. Validación de fechas

Al cargar las fechas, ya sea de un vuelo o una estadía, se valida con las fechas de comienzo y fin del viaje al que correspondan. De esta manera se mantienen los valores actualizados estos valores para agilizar el cálculo de la duración del mismo viaje.

Además, al manejar estas fechas, se utiliza una excepción que atrapa cuando alguna de estas fechas se encuentra sin inicializar, de manera que la duración sea correctamente cero.

6. Excepciones

NoHayVuelosError Arroja una excepción cuando no hay vuelos en la lista, por ejemplo para solicitar comida abordo del mismo vuelo.

PaqueteNoExisteError Arroja una excepción cuando el paquete buscado se encuentra en el diccionario de paquetes.

VueloEnMismaCiudadError Arroja una excepción cuando se intenta agregar un vuelo que sale de una ciudad y llega a la misma.

7. Diagramas de secuencia

El siguiente diagrama ilustra la interacción de los objetos a partir del método de AlgoTrip obtenerCostoDelViaje(nombreDeViaje).

El mismo busca el viaje correspondiente en su diccionario de viajes y luego se llevan a cabo las interacciones del diagrama.

Se observa que, al llevarse a cabo el método obtenerCosto() de Viaje, también se lleva a cabo el método en su clase hija Paquete que tiene su propia implementación. De esta manera, se recorren las listas de vuelos y hoteles en el viaje y en sus paquetes guardados.

Dentro de la clase Hotel se llevan a cabo los métodos calcularDias() y calcularCosto(), ya que el costo se calcula en base a los días de la estadía.

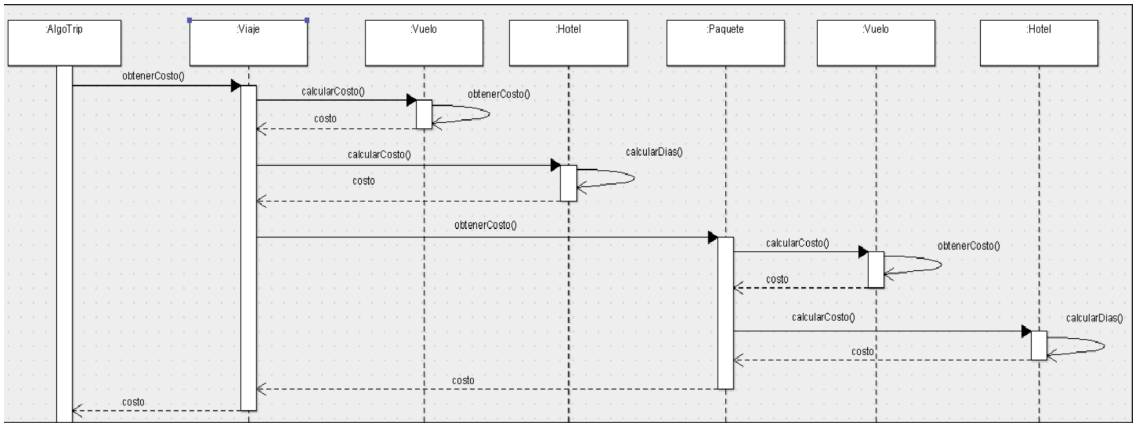

Figura 4: Diagrama de secuencia de obtenerCosto()

El siguiente diagrama ilustra la interacción de los objetos a partir del método de AlgoTrip obtenerDuracionEnDiasDelViaje(nombreDeViaje).

El mismo busca el viaje correspondiente en su diccionario de viajes y luego se llevan a cabo las interacciones del diagrama.

La idea detrás del método es muy similar al mñetodo obtenerCostodelViaje(nombreDeViaje), pero se observa un diagrama mucho más simple. Esto se debe a que se actualizan las fechas límites del viaje, correspondientes a su inicio y su fin, cada vez que se agrega un vuelo o una estadía. De esta manera se agiliza la interacción entre los objetos en este método y se redice considerablemente la complejidad.

También se puede observar que se lleva a cabo la implementación de la clase Paquete del método obtenerDuracion() para poder considerar tanto la duración de los vuelos y estadías dentro del paquete como fuera del mismo.

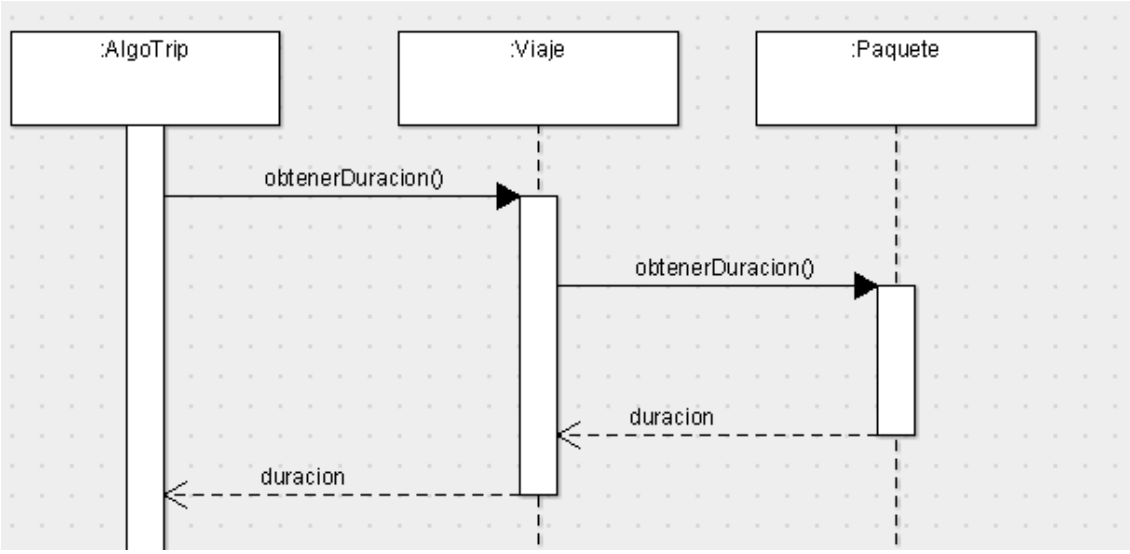

Figura 5: Diagrama de secuencia de obtenerDuracion().PNG.