Jakub Balcerzak

Inżynieria Oprogramowania

5. etap – sprawozdanie

Identyfikacja klas reprezentujących logikę biznesową projektowanego oprogramowania. Definicja atrybutów i operacji klas oraz związków między klasami na podstawie analizy scenariuszy przypadków użycia. Opracowanie diagramów klas i pakietów. Zastosowanie projektowych wzorców strukturalnych i wytwórczych.

1. Analiza scenariuszy przypadków użycia – identyfikacja klas:

Analiza przeprowadzana poniżej w punktach 1.2.1, 1.2.2, 1.2.3, 1.2.4 oparta jest na analizie scenariuszy przypadków użycia: PU Dodanie Nowego Zlecenia, PU Zarządzanie Usługami, PU Zarządzanie Zleceniem, PU Wstawianie rachunku oraz pozostałymi PU zdefiniowanymi w etapie 3. laboratoriów.

1.2.1. Analiza klas bazowych:

Program posiada cztery klasy główne: Service, ServicesManager, Order, OrdersManager.

W aplikacji nie zastosowano dziedziczenia, ponieważ żaden z obiektów nie potrzebuje dostępu do tej samej grupy metod. Każda z występujących klas istotnie różni się od siebie i pełni różne funkcje w programie

1.2.2. Powiązania między klasami:

- Zdefiniowano klasę OrdersManager, która przechowuje listę wszystkich zleceń (obiektów typu Order) wykonywanych przez przedsiębiorstwo oraz wykonuje operacje związane z zarządzaniem tą listą.
- Zdefiniowano klasę ServicesManager, która przechowuje listę wszystkich usług (obiektów typu Service) świadczonych przez przedsiębiorstwo oraz wykonuje operacje związane z zarządzaniem tą listą.
- Powyższe klasy (OrdersManager i ServicesManager) pozwalają GUI na dostęp do operacji na poszczególnych zleceniach. Przykładem jest dodanie nowego zlecenia, gdzie OrdersManager tworzy i dodaje nowe zlecenie (obiekt typu Order), a ServicesManager wyświetla dostępne usługi (obiekty typu Service), które następnie przypisywane są do aktualnego zlecenia.
- Zdefiniowano związek typu asocjacja między klasami ServicesManager i OrdersManager. Zastosowano to ponieważ przy usuwaniu poszczególnej usługi z bazy usług, program musi zaktualizować tę zmianę w aktualnie świadczonych usługach.
- Klasa Order posiada metodę Bill(), która zwraca tekst podsumowujący dane zlecenie (ID poszczególnego zamówienia, listę usług oraz ich cen, na końcu podsumowanie – czyli sumę cen wszystkich usług). Wywoływana jest przez OrdersManagera dla aktualnie wybranego zlecenia (obiektu typu Order) w interfejsie graficznym aplikacji.

1.2.3. Wykryto następujące związki pomiędzy klasami:

- Silna agregacja między obiektem typu OrdersManager i obiektami typu Order (menedżer zleceń (OrdersManager) posiada kolekcję wszystkich zleceń(Order))
- Silna agregacja między obiektem typu ServicesManager i obiektami typu Service (menedżer usług (ServicesManager) posiada kolekcję wszystkich dostępnych usług(Service))
- Relacja <<use>> między obiektem typu Order i obiektami typu Service (zlecenie (Order) potrzebuje usług(Service) do pełnej implementacji)
- Asocjacja między obiektem typu ServicesManager i obiektem typu OrdersManager (menedżer usług (ServicesManager) posiada referencje do obiektu menedżera zleceń (OrdersManager))

1.2.4. Wykorzystane wzorce projektowe w aplikacji:

- Wzorzec wytwórczy Fabryka został poniekąd zastosowany w klasach ServicesManager oraz OrdersManager. Za pomocą tych klas tworzymy nowe zlecenia (obiekty typu Order) oraz nowe usługi (obiekty typu Service).
- Wzorzec strukturalny Fasada został zastosowany w interfejsie graficznym aplikacji (GUI). Każdy interaktywny element korzysta z instancji klas OrdersManager oraz ServicesManager (relacja <<use>>). Za pomocą tych klas, interaktywne elementy GUI ładują z nich odpowiednio potrzebne zestawy metod oraz danych.
- Wzorzec strukturalny Pyłek jest wykorzystywany przy dodawaniu usług (Services) do zleceń (Orders). Obiekty typu Service tworzone są tylko raz, a następnie ich referencje przypisywane są do poszczególnych zleceń - co prowadzi do wykorzystywania tego samego obiektu w całej aplikacji.

2. Wstępny diagram klas (reprezentujący warstwę biznesową oprogramowania):