Kacper Cierzniewski(218233)

Anna Jankowska (218187)

Termin laboratorium: wt, 18:55

Termin oddawania sprawozdania: 10.01.2017

Etap 8: Opracowanie diagramu stanów dla wybranej klasy, reprezentującego wpływ różnych przypadków użycia na zmiany stanów tej klasy, modelowanych za pomocą diagramów sekwencji.

1. Diagramy stanów
   1.1. Diagram stanów obiektu klasy Rezerwacja:

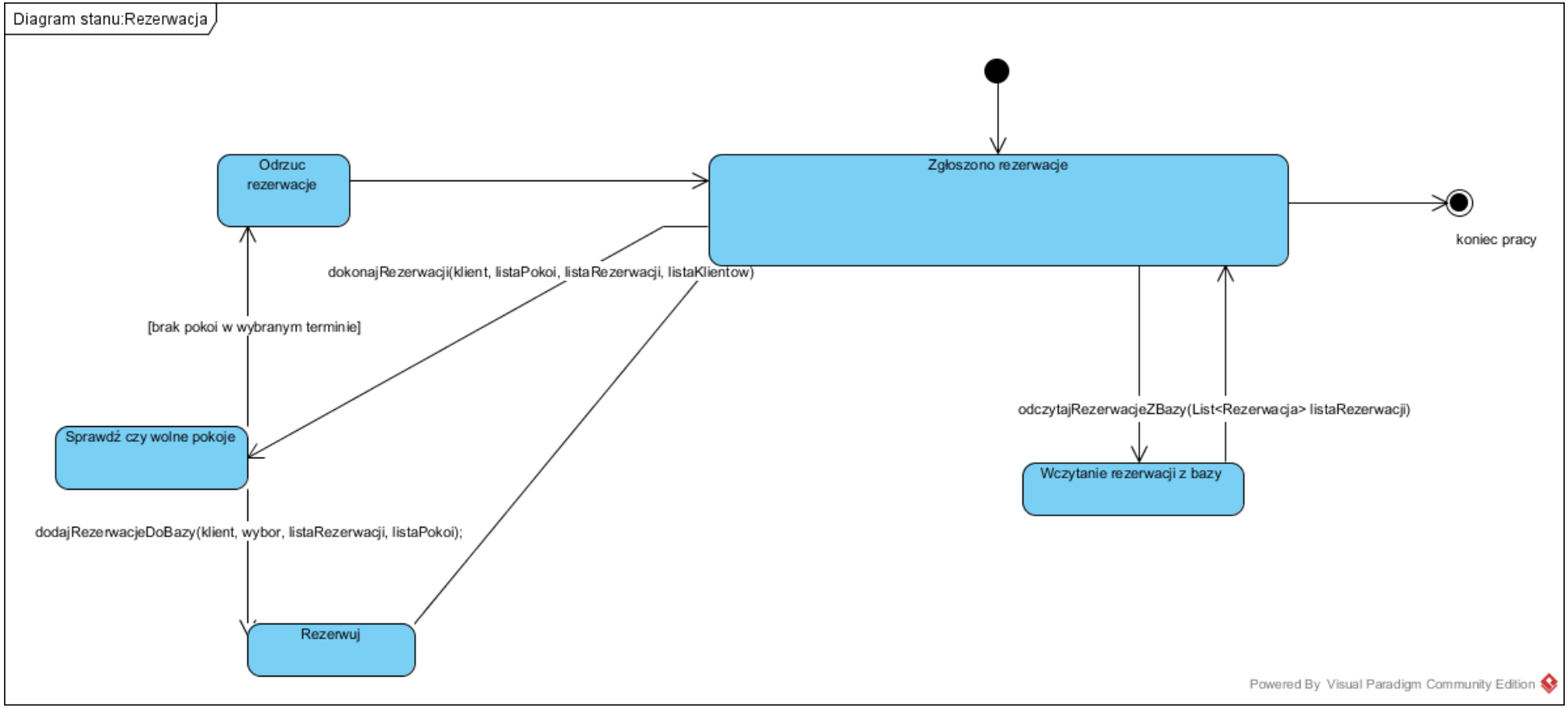

Diagram stanów 1: Klasa Rezerwacja

1.2. Diagram stanów obiektu klasy Rachunek:

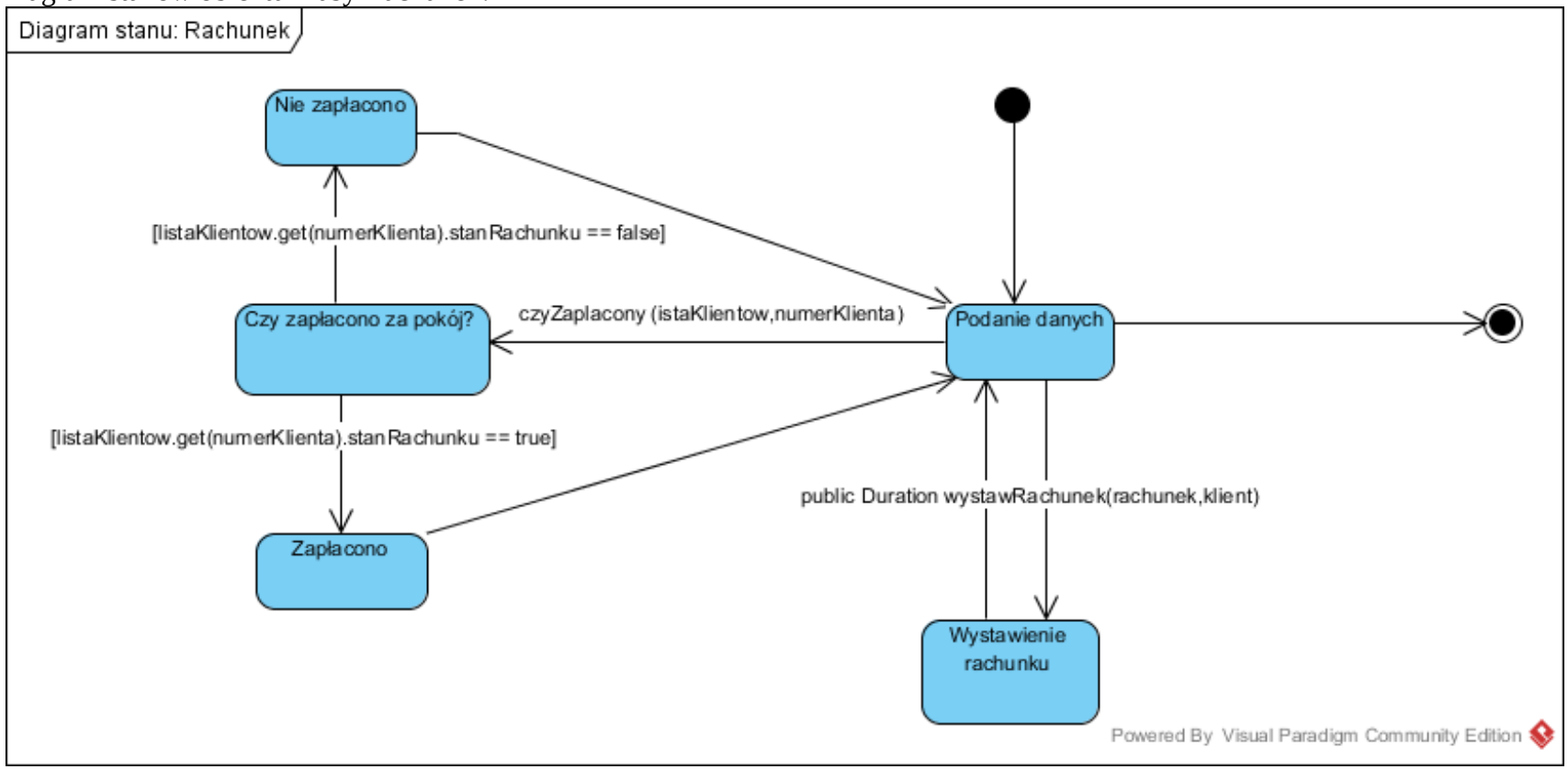

Diagram stanów 2 Klasa Rachunek

2. Kody dla wyszczególnionych klas

2.1 Kod dla klasy Rezerwacja:

```
public void dokonajRezerwacji(Klient klient, List<Pokoj> listaPokoi, List<Rezerwacja> listaRezerwacji, List<Klient>
listaKlientow) {
    Scanner scanner = new Scanner(System.in);
    String wybor;
    List<Pokoj> listaDostepnychPokoi = null;
    System.out.println("Lista dostępnych pokoi: ");
    for (int i = 0; i < listaPokoi.size(); i++) {
        if (klient.odKiedy.before(listaPokoi.get(i).odKiedy) && klient.doKiedy.before(listaPokoi.get(i).odKiedy) ||
klient.odKiedy.after(listaPokoi.get(i).doKiedy)) {
            System.out.println("Numer pokoju: " + listaPokoi.get(i).numerPokoju + " " + listaPokoi.get(i).kwota + "
zł");
        }
        }
        if (listaPokoi.size() > 0) {
            System.out.println("Wybierz pokój do rezerwacji: ");
            wybor = scanner.nextLine();

            try {

                dodajRezerwacjeDoBazy(klient, wybor, listaRezerwacji, listaPokoi);
                klient.dodajKlientaDoBazy(klient, listaKlientow);
                System.out.println("Rezerwacja dokonana pomyślnie! ");
            } catch (ParseException e) {
                e.printStackTrace();
            }

        }
     else{
        System.out.println("Brak dostępnych pokoi w wybranym terminie!");
    }

}
```

```
public void dodajRezerwacjeDoBazy(Klient klient, String wyborPokoju, List<Rezerwacja> listaRezerwacji,List<Pokoj>
listaPokoi) throws ParseException {
    FileWriter zapis ;
    try {
        zapis = new FileWriter("rezerwacje.txt", true);
        numerRezerwacji=listaRezerwacji.size()+1;
        zapis.write("\n" +numerRezerwacji+" " +klient.numerKlienta + " " +wyborPokoju + " " +klient.odKiedy + " " +
klient.doKiedy+" ");
        zapis.close();
    } catch (IOException e) {
        e.printStackTrace();
    }

}
```

2.2 Kod klasy Rachunek

```
public Duration wystawRachunek(Rachunek rachunek, Klient klient) {
    Duration kwota = Duration.between(odKiedy,doKiedy)*100;
    return kwota;
}

public boolean czyZaplacony (List<Klient> listaKlientow, int numerKlienta){
    Recepcja _recepcja = new Recepcja();
    if (listaKlientow.get(numerKlienta).stanRachunku == false) return false;
    else return true;
}
```