Politechnika Wrocławska
Katedra Informatyki Technicznej

Inżynieria Oprogramowania

Magdalena Biernat
Mateusz Bortkiewicz

Opiekun
prof. dr hab. inż. Jan Magott

15 grudnia 2017

1 Wprowadzenie

Sprawozdanie dotyczy dziewiątych i dziesiątych zajęć. Na tych laboratoriach kontynuowaliśmy swój projekt.

1.1 Cel laboratorium

Definiowanie w sposób iteracyjno - rozwojowy modelu projektowego programowania opartego na:

- Modelowaniu logiki biznesowej reprezentowanej przez wybrany przypadek użycia za pomocą diagramów sekwencji po wykonaniu pierwszego przypadku użycia podczas laboratorium 8, stanowiącego bazową logikę biznesową, z której korzystają kolejne przypadki użycia. Należy definiować operacje i atrybuty kolejnej klasy (dziedziczenie, powiązania i agregacje) na diagramie klas zidentyfikowanej w wyniku modelowania kolejnego przypadku użycia i wykonanie scenariusza tego przypadku użycia za pomocą diagramu sekwencji.
- Implementacja modelu projektowego wybranego przypadku użycia za pomocą języka Java SE – rozszerzanie kodu źródłowego programu wykonanego podczas laboratoriów 7 i 8.

1.2 Plan pracy

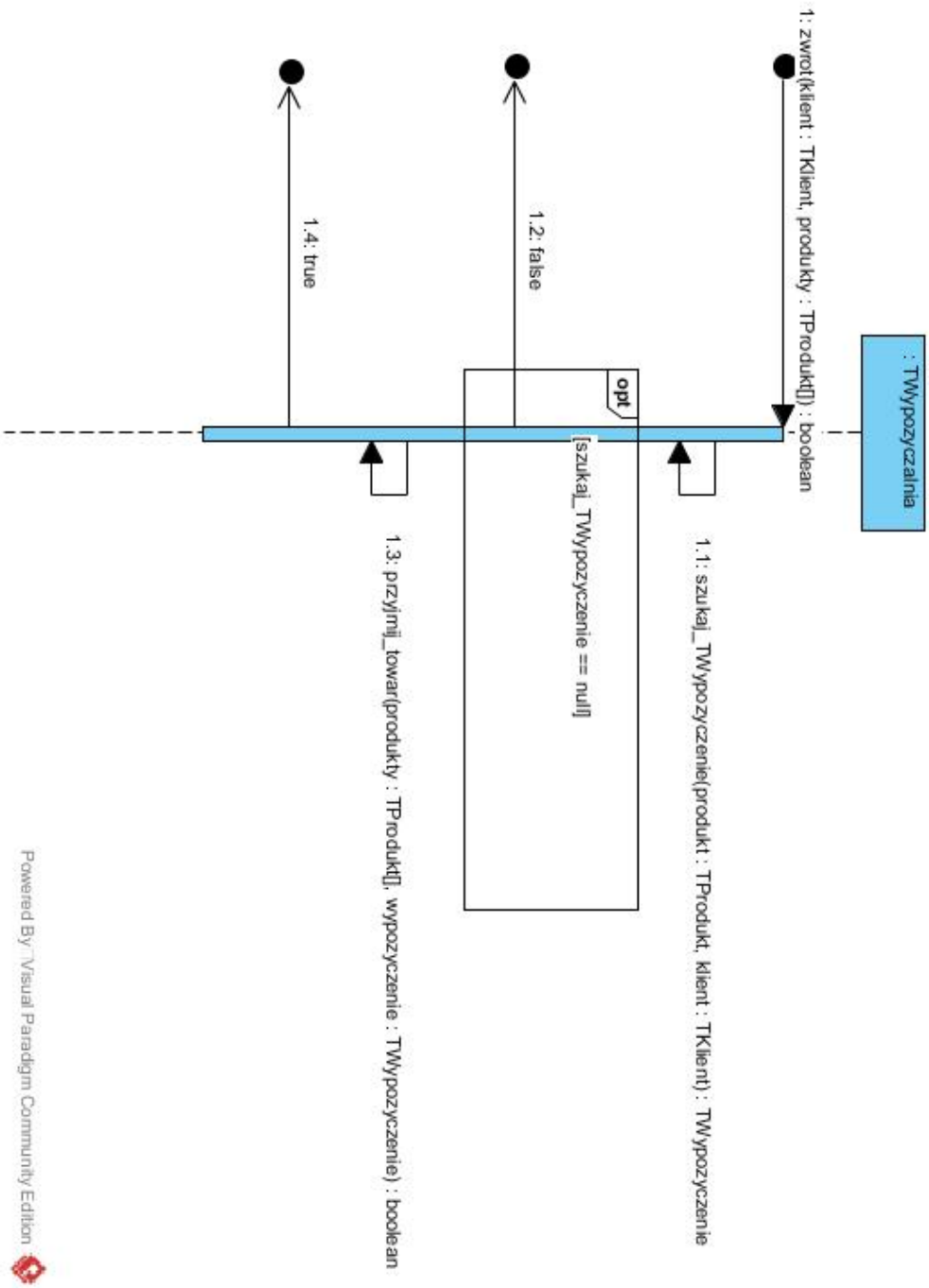
Zadania wykonaliśmy wg instrukcji 7:

- Wykonanie diagramów sekwencji.
- Poprawa diagramu klas.
- Implementacja kodu w javie.

2 Laboratorium

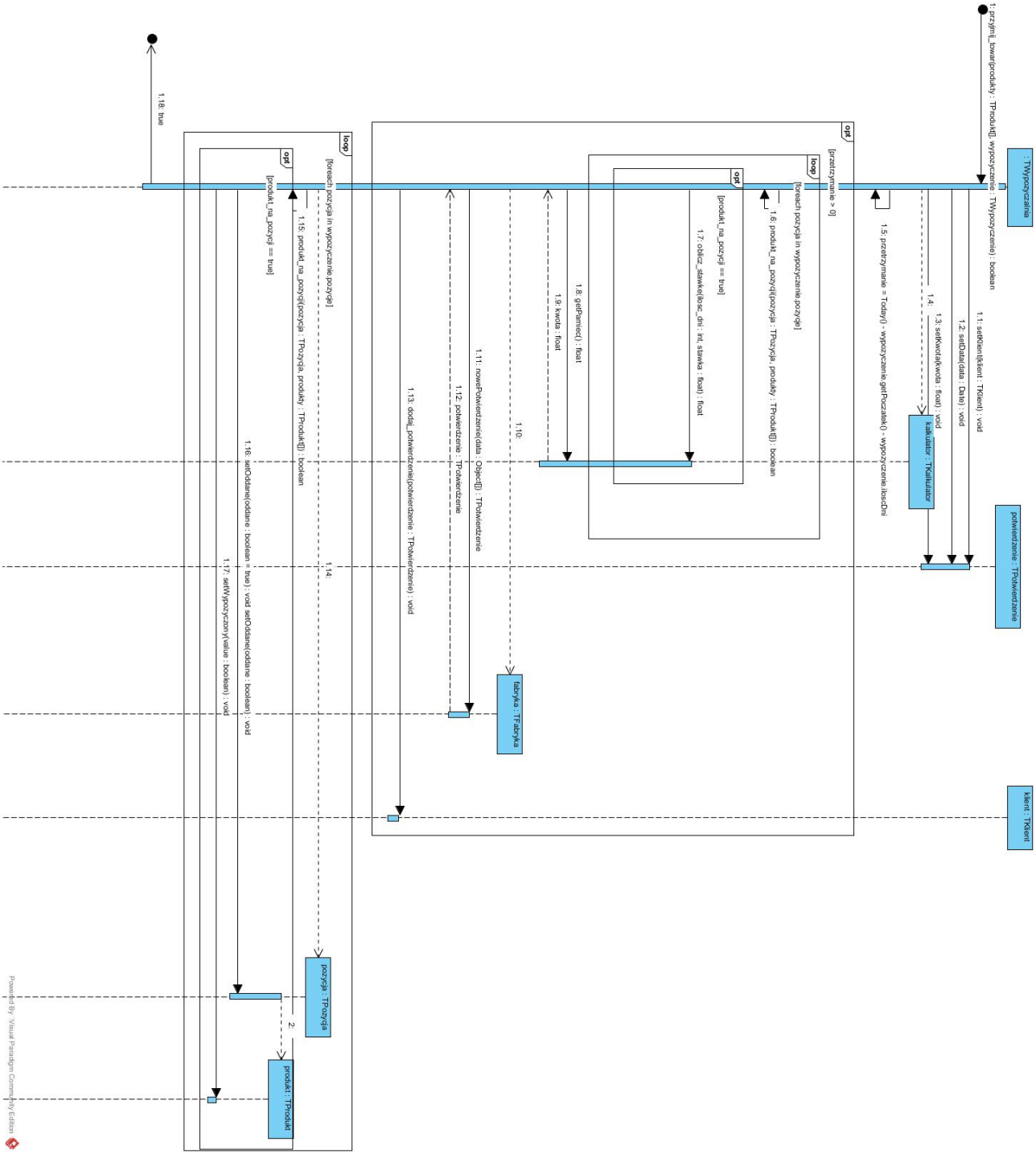
2.1 Wykonane diagramy sekwencji

2.1.1 Diagram sekwencji zwrotu



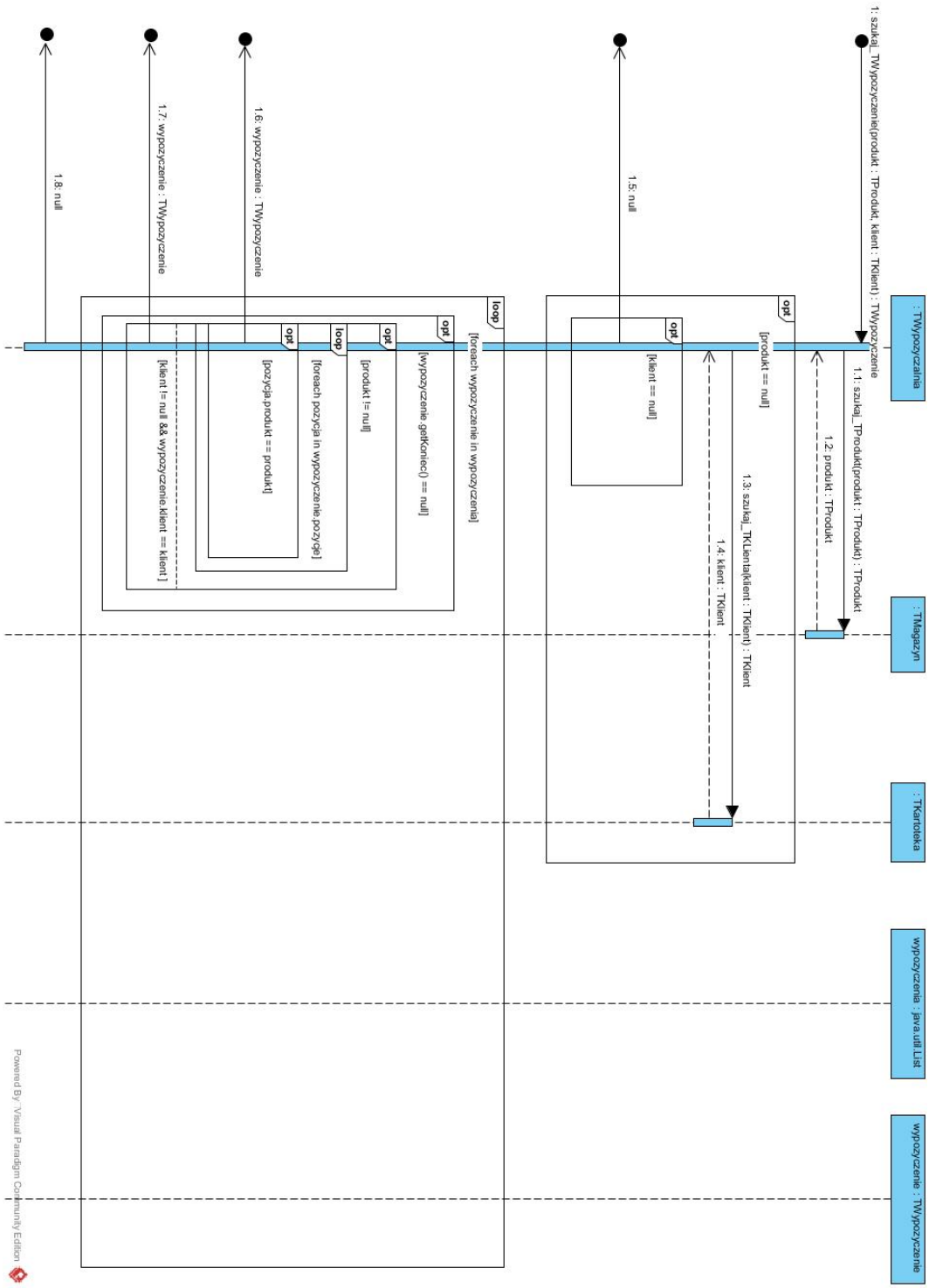
Rysunek 1: Stworzony diagram sekwencji

2.1.2 Diagram sekwencji wypożyczenia



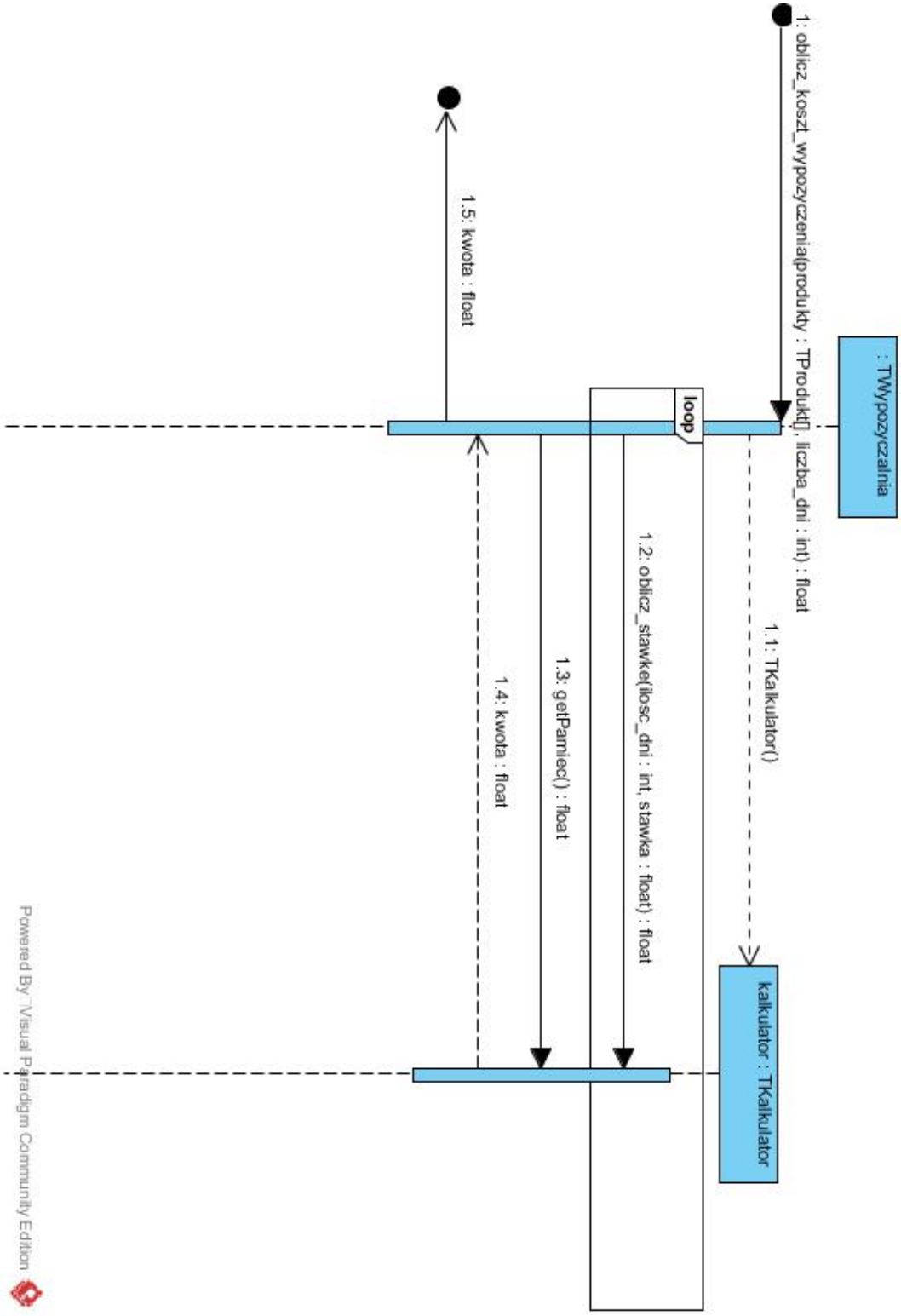
Rysunek 2: Stworzony diagram sekwencji

2.1.3 Diagram sekwencji Szukanie wypożyczenia



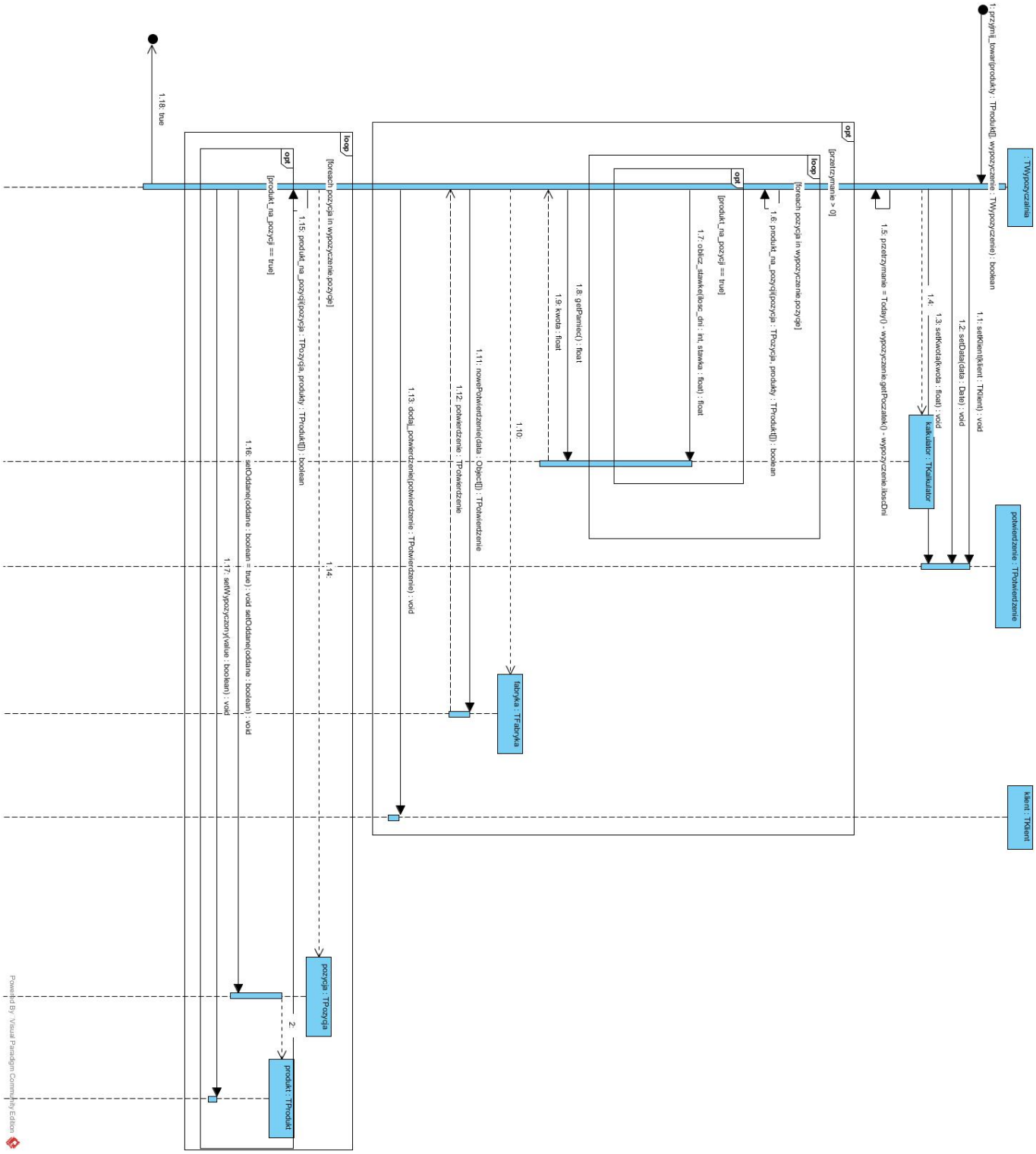
Rysunek 3: Stworzony diagram sekwencji

2.1.4 Diagram sekwencji PU Obliczanie terminu zwrotu i kosztu wypożyczenia



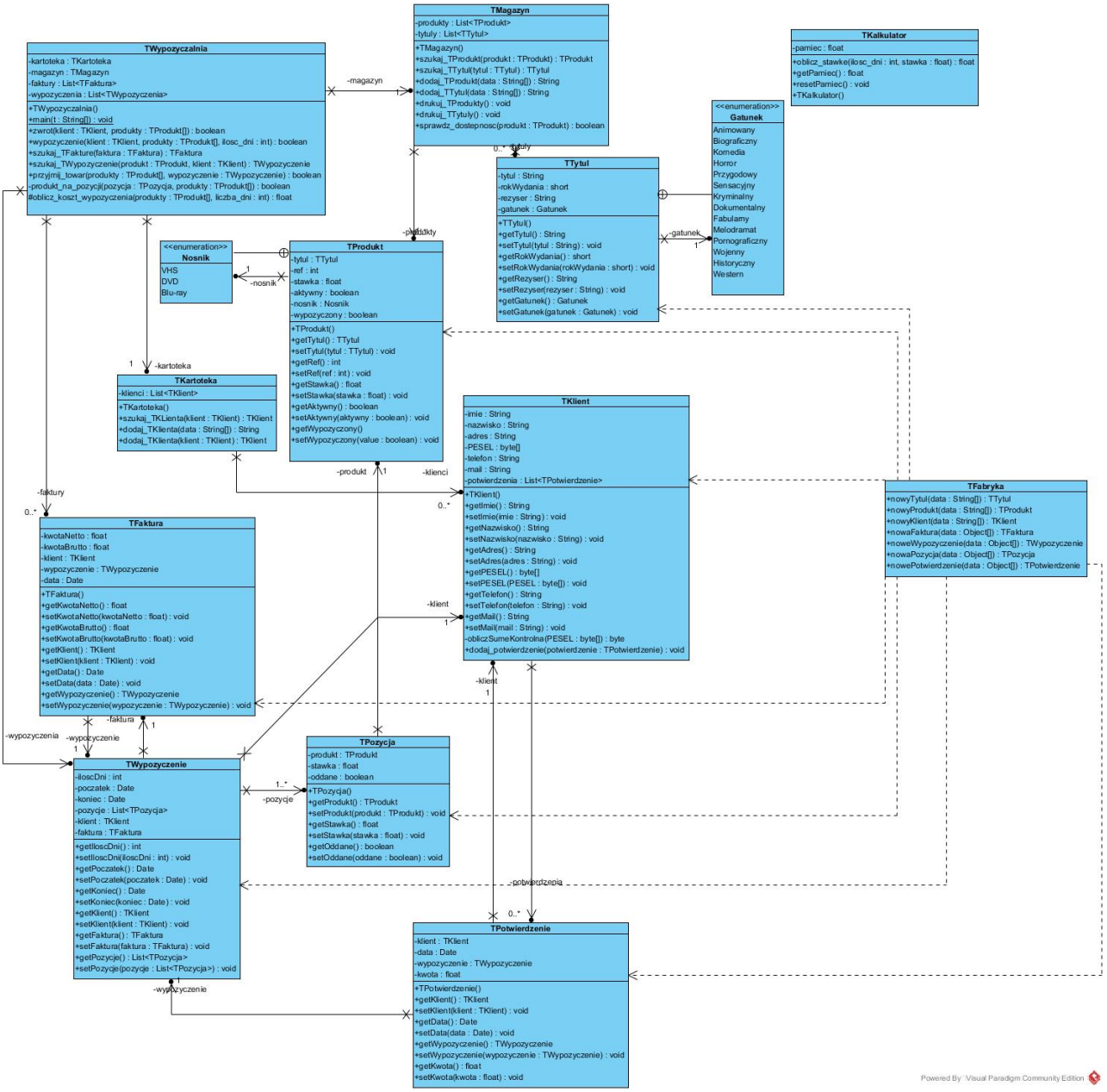
Rysunek 4: Stworzony diagram sekwencji

2.1.5 Diagram sekwencji PU Przyjęcie towaru



Rysunek 5: Stworzony diagram sekwencji

2.2 Diagram klas



Rysunek 6: Poprawiony diagram klas

2.3 Kod w javie

Poniżej zawartość klasy TWypozyczalnia.

```
import java.util.ArrayList;
import java.util.Calendar;
import java.util.List;
import java.util.Date;

public class TWypozyczalnia {

private TKartoteka kartoteka;
private TMagazyn magazyn;
private List<TFaktura> faktury;
private List<TWypozyczenie> wypozyczenia;

public TWypozyczalnia() {
super();
}

public static void main(String[] t) {
return;
}

public boolean zwrot(TKlient klient, TProdukt[] produkty) {
TWypozyczenie wypozyczenie = szukaj_TWypozyczenie(produkty[0], klient);

if(wypozyczenie == null)
return false;
else {
przyjmij_towar(produkty, wypozyczenie);
return true;
}
}

public boolean wypozyczenie(TKlient klient, TProdukt[] produkty, int ilosc_dni) {
TKlient nowyKlient;
TKalkulator kalkulator = new TKalkulator();
TFabryka fabryka = new TFabryka();
TWypozyczenie wypozyczenie;
TFaktura faktura;

boolean dostepnosc;
```

```
float kwota;

for(int i=0; i < produkty.length; i++) {
dostepnosc = magazyn.sprawdz_dostepnosc(produkty[i]);

if(dostepnosc == false)
return false;
}

nowyKlient = kartoteka.szukaj_TKlienta(klient);

if(nowyKlient == null)
nowyKlient = kartoteka.dodaj_TKlienta(klient);

for(int i =0; i < produkty.length; i++)
kalkulator.oblicz_stawke(ilosc_dni, produkty[i].getStawka());

kwota = kalkulator.getPamiec();
List<TPozycja> pozycje = new ArrayList<TPozycja>();
for(int i=0; i < produkty.length; i++)
pozycje.add(fabryka.nowaPozycja(new Object[]{produkty[0], produkty[0].getStawka(), false}));

wypozyczenie = fabryka.noweWypozyczenie(new Object[]{ilosc_dni, Calendar.getInstance().getTime(), nul
faktura = fabryka.nowaFaktura(new Object[]{100*kwota/123, kwota, nowyKlient, wypozyczenie, wypozyczer
wypozyczenie.setFaktura(faktura);
faktury.add(faktura);
wypozyczenia.add(wypozyczenie);

return true;
}

public TFaktura szukaj_TFakture(TFaktura faktura) {
return null;
}

public TWypozyczenie szukaj_TWypozyczenie(TProdukt produkt, TKlient klient) {
produkt = magazyn.szukaj_TProdukt(produkt);

if(produkt==null) {
klient = kartoteka.szukaj_TKlienta(klient);
if(klient==null)
```

```
return null;
}

for(TWypozyczenie wypozyczenie : wypozyczenia) {
if(wypozyczenie.getKoniec()==null)
{
if(produkt != null) {
for(TPozycja pozycja : wypozyczenie.getPozycje()) {
if(pozycja.getProdukt() == produkt)
return wypozyczenie;
}
}
else if(klient != null && wypozyczenie.getKlient() == klient)
return wypozyczenie;
}
}
return null;
}

public boolean przyjmij_towar(TProdukt[] produkty, TWypozyczenie wypozyczenie) {
TKalkulator kalkulator = new TKalkulator();
int przetrzymanie = daysBetween(Calendar.getInstance().getTimeInMillis(), wypozyczenie.getPoczatek()
wypozyczenie.getIloscDni();
float kwota;
if (przetrzymanie > 0) {
for(TPozycja pozycja : wypozyczenie.getPozycje()) {
if(produkt_na_pozycji(pozycja, produkty)) {
kwota = kalkulator.oblicz_stawke(przetrzymanie, pozycja.getStawka());
}

kwota = kalkulator.getPamiec();
TFabryka fabryka = new TFabryka();
TPotwierdzenie potwierdzenie = fabryka.nowePotwierdzenie(new Object[]{Calendar.getInstance().getTime
wypozyczenie.getKlient(),
kwota,
wypozyczenie});
wypozyczenie.getKlient().dodaj_potwierdzenie(potwierdzenie);
}

for(TPozycja pozycja : wypozyczenie.getPozycje()) {
boolean produkt = produkt_na_pozycji(pozycja, produkty);
```

```
if(produkt) {
pozycja.setOddane(true);
pozycja.getProdukt().setWypozyczony(false);
}
}
}

return true;
}

private boolean produkt_na_pozycji(TPozycja pozycja, TProdukt[] Produkty) {
return false;
}

private int daysBetween(long t1, long t2) {
return (int) ((t2 - t1) / (1000 * 60 * 60 * 24));
}
}
```