Programowanie obiektowe

Wykład 12

Marcin Młotkowski

26 maja 2022

Plan wykładu

1. Analiza obiektowa
   - Dziedziczenie
   - Dziedziczenie a składanie
2. Programowanie obiektowe
   - Implementacja związków gen–spec
   - Implementacja agregacji
   - Implementacja asocjacji
   - Mnożenie obiektów

Plan wykładu

Kiedy dziedziczyć

Wskazówka

Podklasa przedefiniowuje operację nadklasy

lub

dodaje nową funkcjonalność

Zły przykład

```
class ListaJednokier
{
    Object obj;
    ListaJednokier nast;
    void dodaj(Object obj);
}

class ListaDwukier : ListaJednokier {
    ListaDwukier poprz;
    ListaDwukier nast;
    void dodaj (Object obj); // na początek
    void naKoniec (Object obj); // na koniec
}
```

Analiza przykładu

Klasa ListaDwukier ma zupełnie inną implementację niż klasa ListaJednokier, nie korzysta ani z odziedziczonych pól, ani z odziedziczonych metod.

Klasy ListaDwukier i ListaJednokier mają podobne interfejsy.

Morał

Identyczny interfejs nie musi implikować dziedziczenia.

Przesłanką do dziedziczenia jest wykorzystanie implementacji z nadklasy (dziedziczenie implementacji) i rozszerzenie funkcjonalności

Co z tym zrobić

Wspólna klasa abstrakcyjna

```
abstract class ListaAbstrakcyjna
{
    void public dodaj(Object obj);
}
```

Wspólny interfejs

```
interface ILista
{
    void dodaj(Object obj);
}
```

Przypomnienie

Klasy powinne mieć precyzyjnie określone zadanie.

W przypadku "szerokiej" funkcjonalności klasy lepiej poskładać ją z mniejszych.

Przykład

Wersja prosta

```
class Osoba {
    string Imie, Nazwisko;
    public void edycja() { ... }
    public void odczyt() { ... }
    public void zapis() { ... }
}
```

Bardziej uniwersalna

```
class Osoba {
    string Imie, Nazwisko;
    Edytor e = new EdytorQt();
    BazaDanych bd = new BSDQLite();
}
```

Wzorce projektowe

Poznane wzorce

- Singleton
- MVC
- Szablon i Strategia

Wzorce projektowe

Poznane wzorce

- Singleton
- MVC
- Szablon i Strategia

Źródło wzorców

Wzorce projektowe, E. Gamma, R. Helm, R. Johnson, J. Vlissides

Plan wykładu

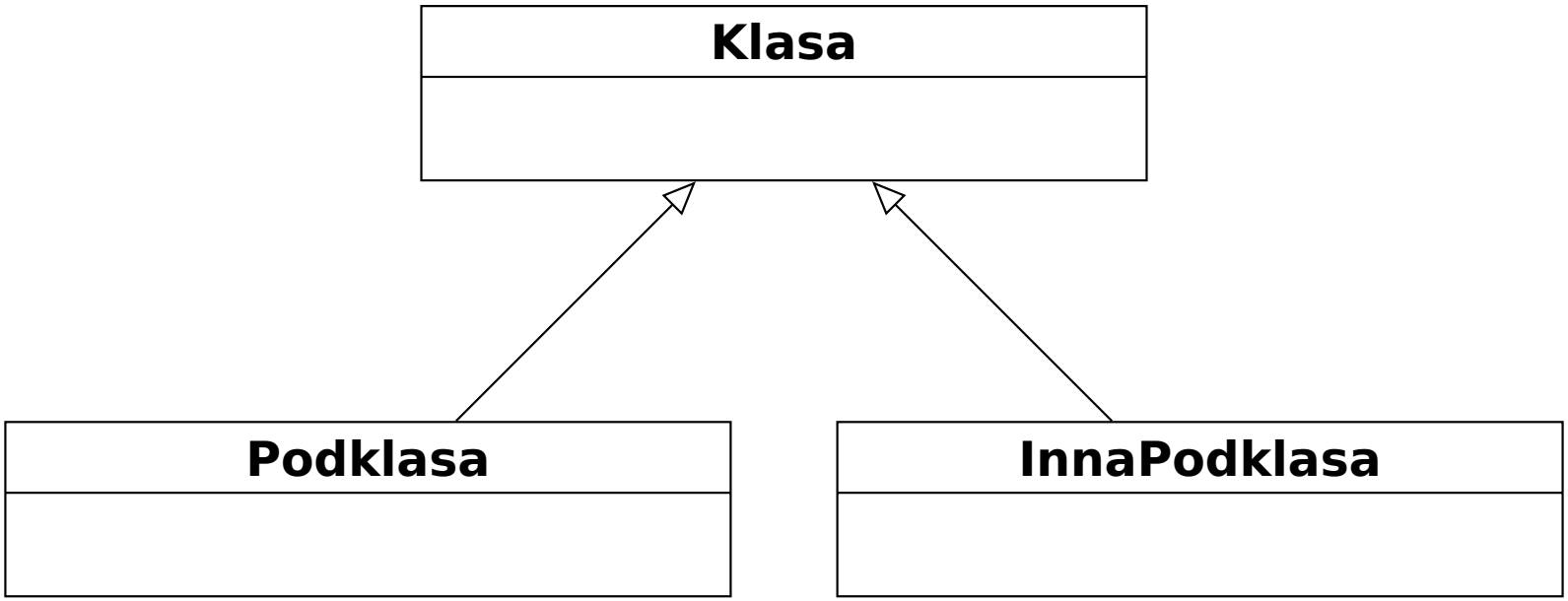
* obiektowe

- Analiza obiektowa
- Projektowanie obiektowe
- Programowanie obiektowe

Programowanie obiektowe

- Implementacja klas wskazanych w analizie
- Implementacja związków
- Uszczegółowienie, tj. dodanie klas

Analiza obiektowa



Implementacja schematu

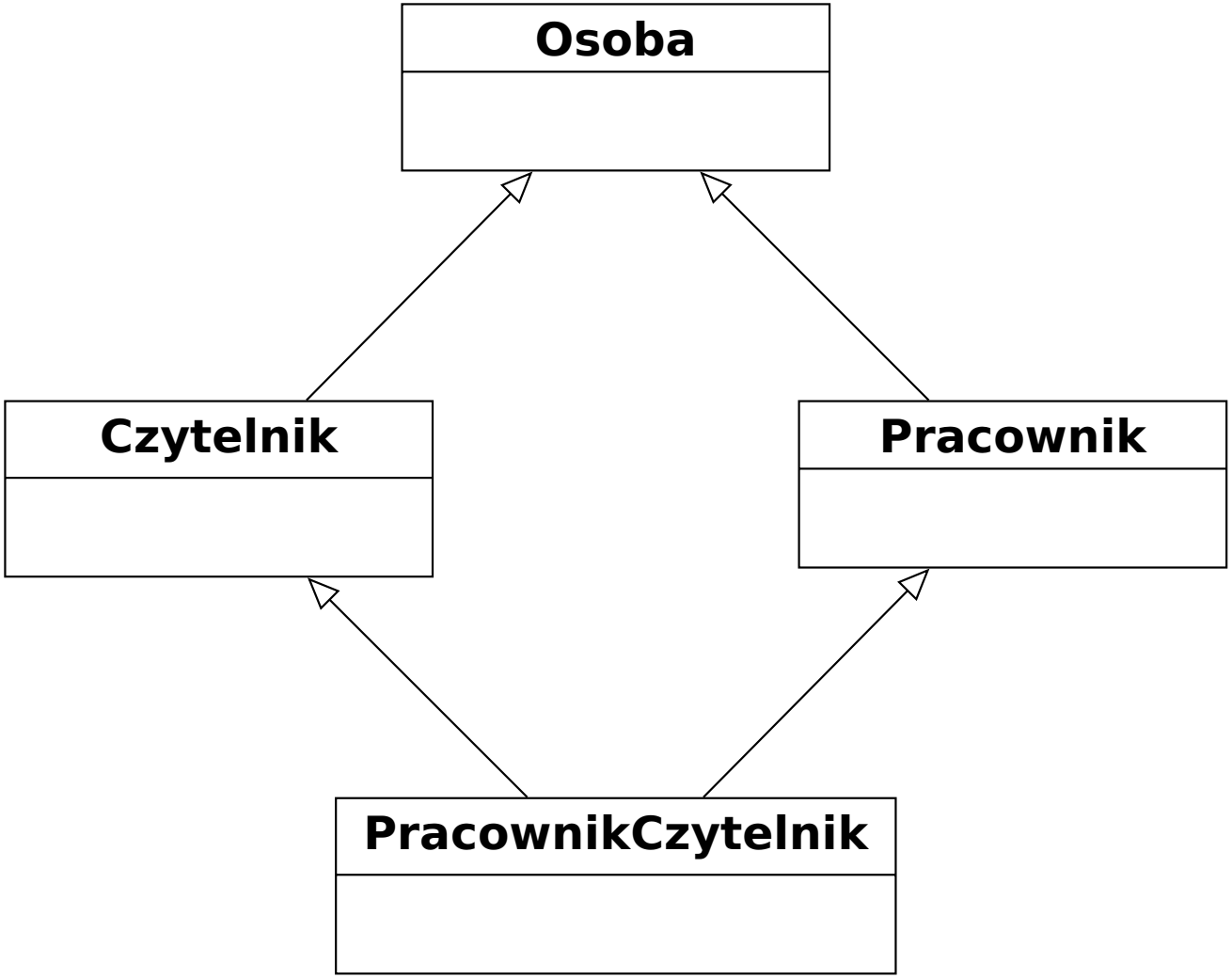
Dziedziczenie

```
abstract class Klasa {
    ...
}

class Podklasa : Klasa {
    ...
}

class InnaPodklasa : Klasa {
    ...
}
```

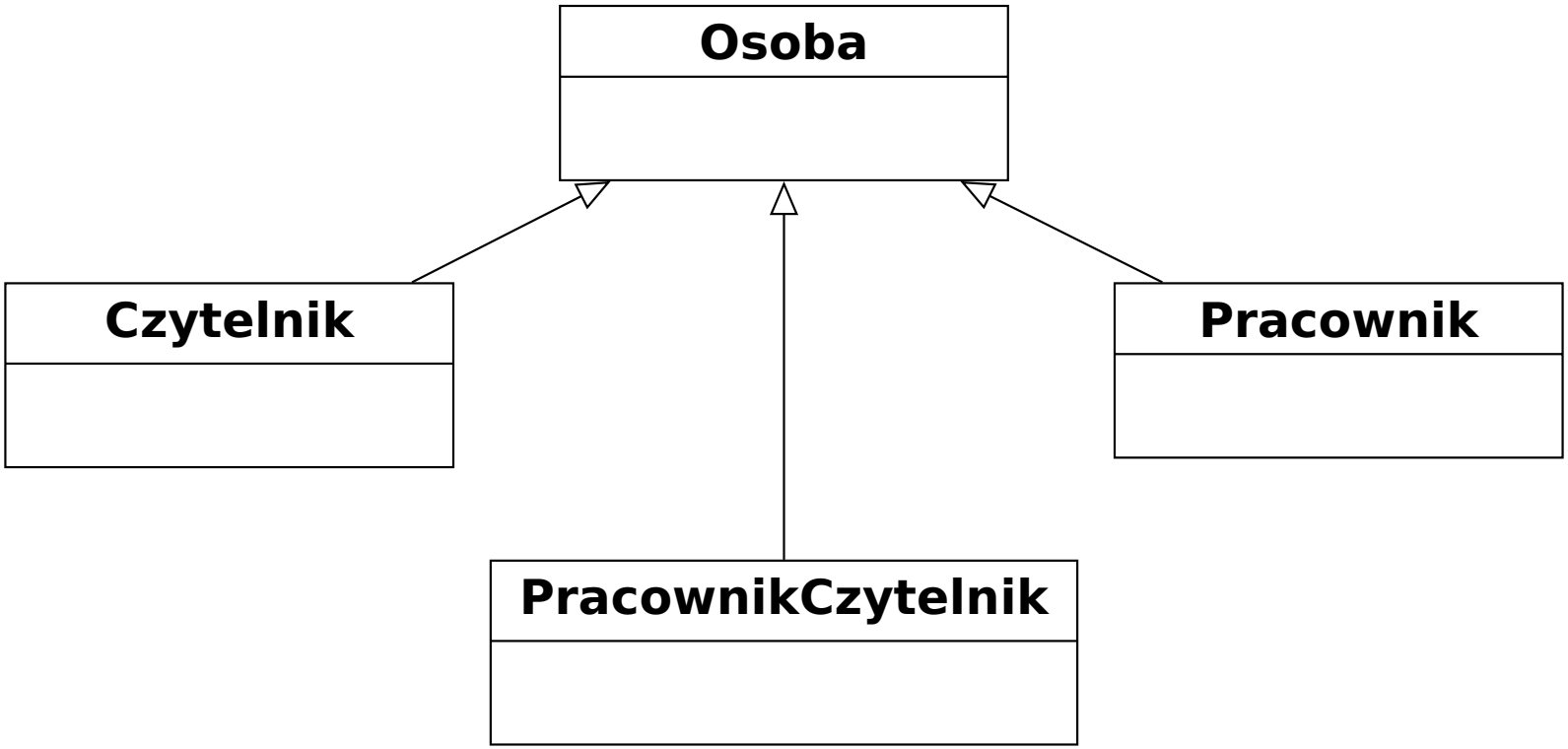
Bardziej skomplikowane zadanie



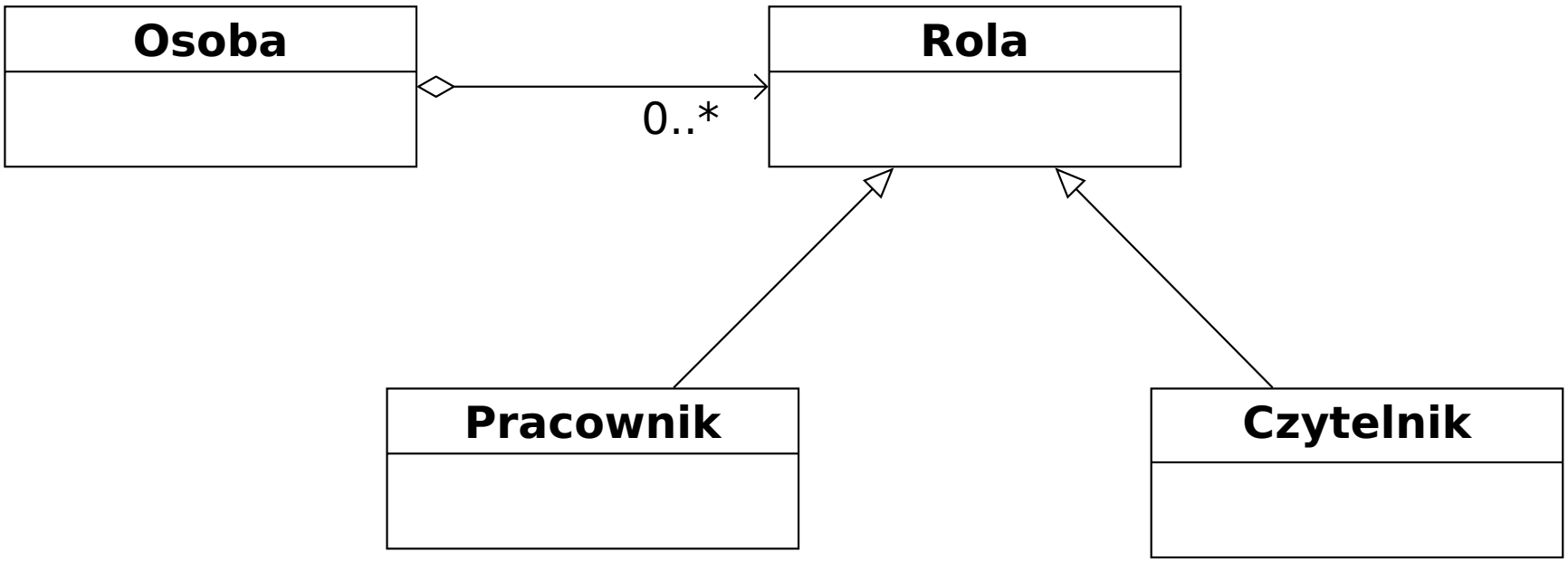
Implementacja

Implementować w języku posiadającym wielodziedziczenie: Python, C++.

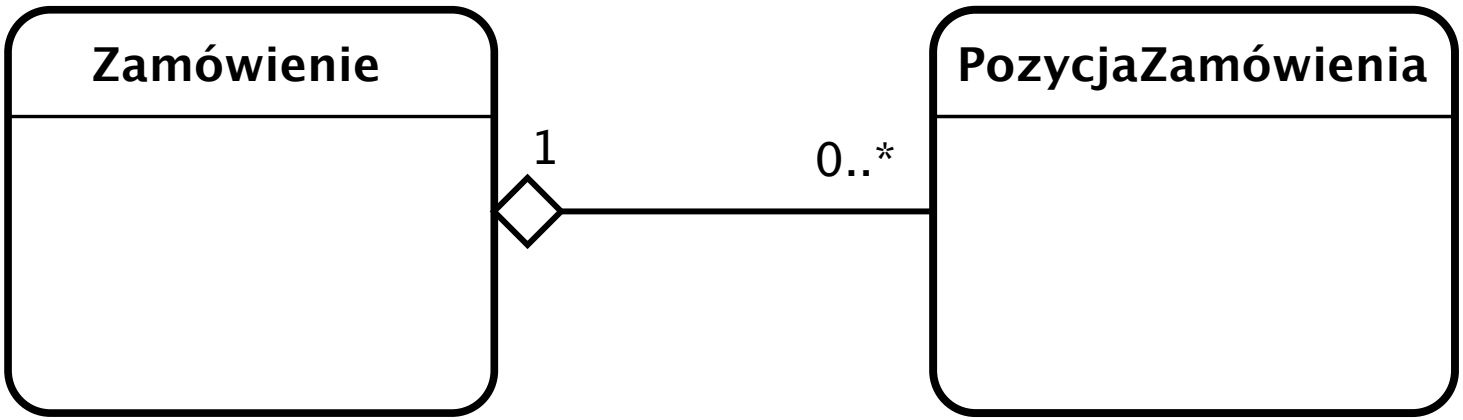
Spłaszczenie hierarchii



Spłaszczenie hierarchii



Agregacja — przypomnienie



Implementacja agregacji

Kolekcje

- Kolekcje pojawiają się jako dodatkowe klasy, nieuwzględniony w ogólnym projekcie
- Kolekcje są obecne w większości (wszystkich?) liczących się środowiskach programistycznych

Samodzielna implementacja kolekcji, 1. podejście

```
class Osoba {
    String Nazwisko;
    Osoba następnik;
    void dołącz(Osoba o) { ... }
}
```

Samodzielna implementacja kolekcji, 1. podejście

```
class Osoba {
    String Nazwisko;
    Osoba następnik;
    void dołącz(Osoba o) { ... }
}
```

Ocena implementacji

- Klasa łączy funkcje listy i Osoby
- Wymaga implementacji mechanizmów listowych dla każdej klasy osobno
- Kłopot z listą pustą

Implementacja kolekcji, 2. podejście

```
class Lista {
    Osoba val;
    Lista następnik;
    void dołącz(Osoba o){ ... }
}
```

Implementacja kolekcji, 2. podejście

```
class Lista {
    Osoba val;
    Lista następnik;
    void dołącz(Osoba o){ ... }
}
```

Ocena implementacji

- Klasa Osoba jest czystą klasą
- Kłopot listą pustą

Implementacja kolekcji, 3. podejście

```
class Lista {
    ElemListy lista;
    bool empty();
    void dołącz(Osoba o);
}

class ElemListy {
    Osoba val;
    ElemListy następnik;
}
```

Ocena implementacji

Zalety

- Klasy mają dokładnie określone zadania
- Klasę Lista można wykorzystywać do przechowywania obiektów innych klas

Wady

Rośnie liczba klas i zależności między nimi.

Implementacja związków między obiektami

- Poprzez referencje
- Utworzenie nowej klasy reprezentującej związek

Przykład

Małżeństwo

Prosty system

zwykła
referencja

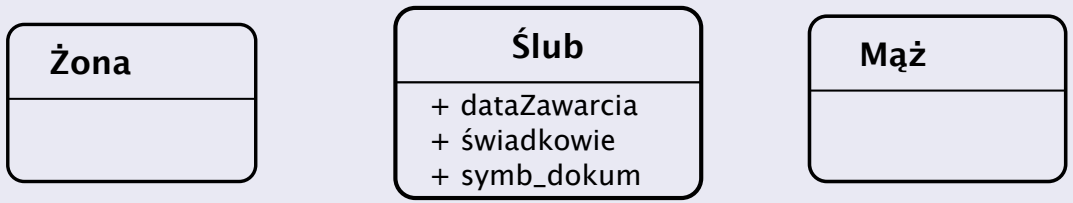
Przykład

Małżeństwo

Prosty system

zwykła referencja

Urząd Stanu Cywilnego



Skąd się jeszcze biorą obiekty

- Przechowywanie danych
- Interfejsy użytkownika
- Aplikacja jako obiekt (singleton)
- ...