ZARZĄDZANIE CYKLEM ŻYCIA APLIKACJI

MODELOWANIE PRZY UŻYCIU UML

CZYM JEST MODELOWANIE?

- Modelowanie jest sposobem radzenia sobie ze złożonością rzeczywistości.
- Modelowanie polega na budowaniu abstrakcji rzeczywistości, w której uwzględnia się elementy istotne a pomija te nieistotne

PO CO MODELOWAĆ OPROGRAMOWANIE?

- Proste reprezentacje złożonych systemów
- Kod jest mało zrozumiały dla programistów nie będących jego twórcami
- Modelowanie promuje myślenie przed działaniem

WPROWADZENIE

UNIFIED MODELING LANGUAGE (UML)

UNIFIED MODELING LANGUAGE

- Stworzony w latach 90-tych przez Grady Boocha, Jamesa Rumbaugha oraz Ivara Jacobsona.
- Zaprojektowany by definiować, wizualizować, konstruować i dokumentować oprogramowanie
- Używany także do modelowania procesów biznesowych, inżynierii systemów i reprezentowania struktur organizacyjnych
- Głównie używany w swojej reprezentacji graficznej

MODELOWANIE STRUKTURALNE

- Modelowanie strukturalne
  - diagram pakietów
  - diagram klas i diagram obiektów
  - diagram struktur złożonych
  - diagram komponentów
  - diagram wdrożenia

MODELOWANIE BEHAWIORALNE

- Modelowanie behawioralne
  - diagram przypadków użycia
  - diagram czynności
  - diagram maszyny stanowej
  - diagramy interakcji (sekwencji, komunikacji, przeglądu interakcji)
  - diagram uwarunkowań czasowych

MODELOWANIE STRUKTURALNE

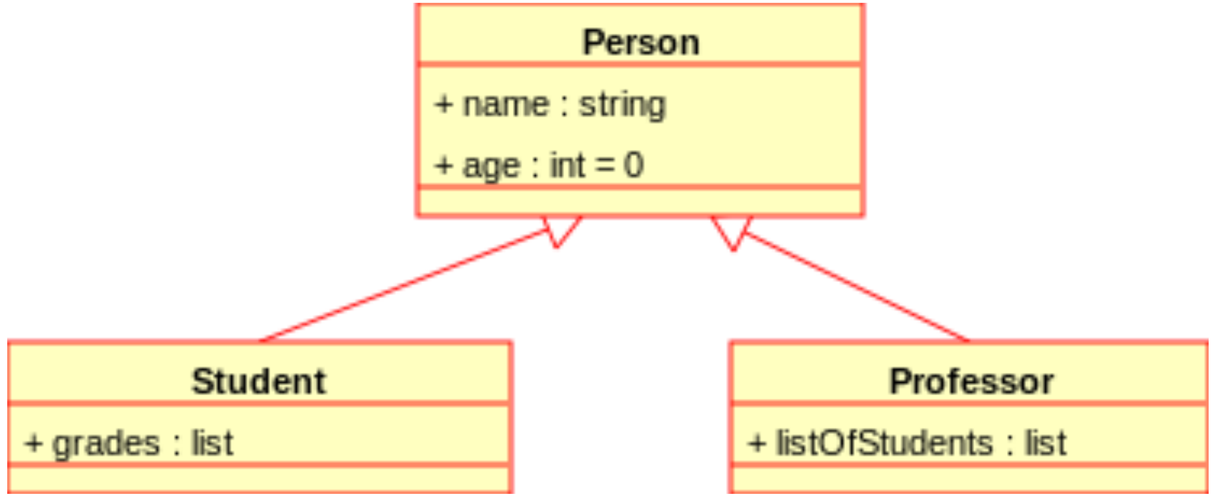
- Modelowanie strukturalne
  - diagram pakietów
  - diagram klas i diagram obiektów
  - diagram struktur złożonych
  - diagram komponentów
  - diagram wdrożenia

PERSPEKTYWA OBIEKTOWA

- Model zbudowany jest z wielu niezależnych obiektów
- Obiekty współpracują ze sobą, według określonych reguł w celu osiągnięcia jakiegoś celu.

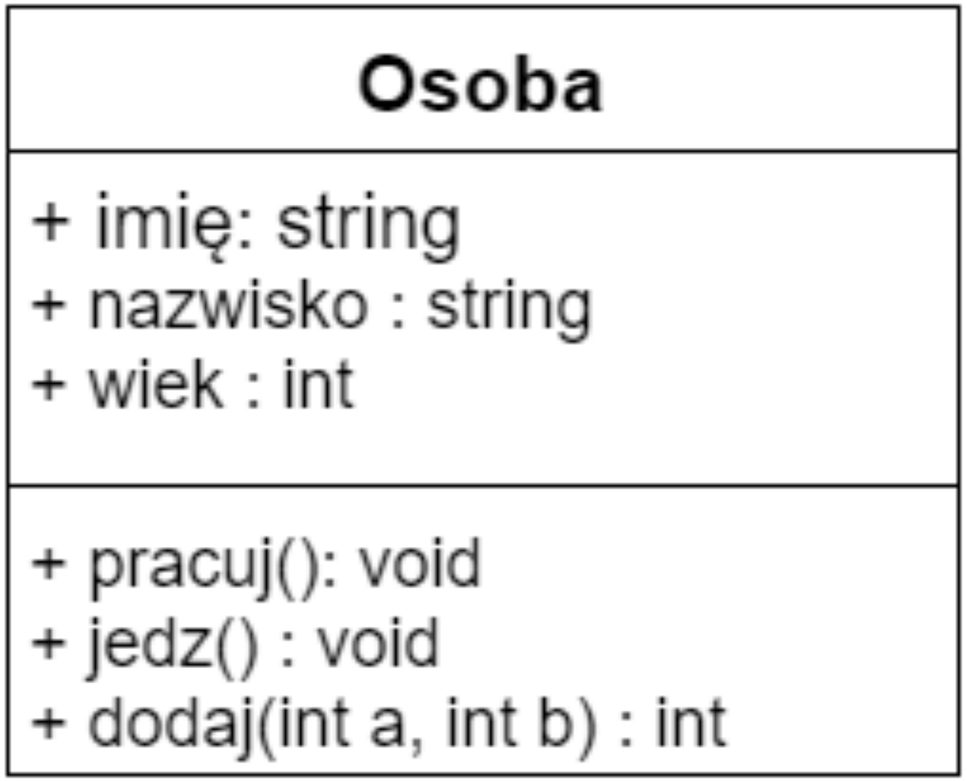
KLASYFIKACJA

- *Systematyczny podział przedmiotów lub zjawisk na klasy, działy, poddziały, wykonywany według określonej zasady (SJP, PWN)*
- Podstawowy sposób budowy modelu świata
- Umożliwia rozszerzanie wiedzy i definiowanie nowych klas obiektów i nowych zachowań



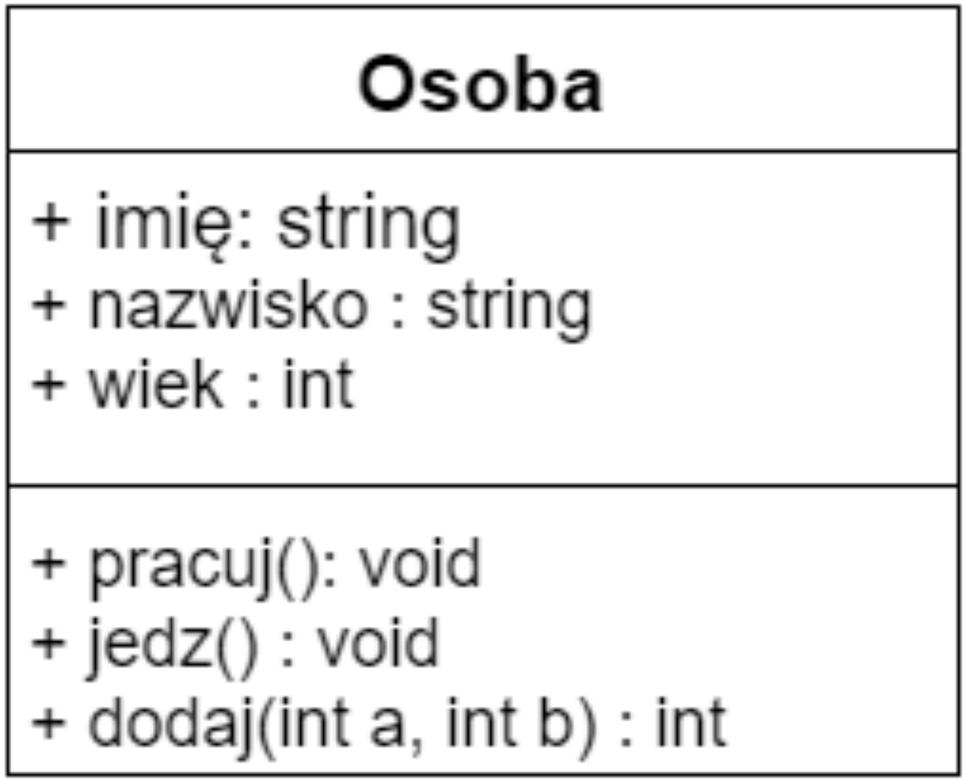
SKŁADNIKI KLASY

- Nazwa
- Atrybuty
- Metody



ATRYBUTY

- Widoczność (+, #, -)
- Nazwa
- Typ
- Krotność
- Ograniczenia
- Wartości domyślne



ATRYBUTY

Składnia:

WIDOCZNOŚĆ NAZWA : TYP [KROTNOŚĆ] {OGRANICZENIE} = WARTOŚĆ DOMYŚLNA

Przykład atrybutu osoby (klasy Person):

+ Daughters : Person [*] {unique}

METODY

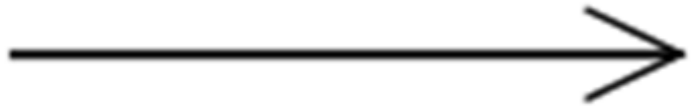
Składnia:

WIDOCZNOŚĆ NAZWA(parametry) : TYP ZWRACANY {OGRANICZENIA}

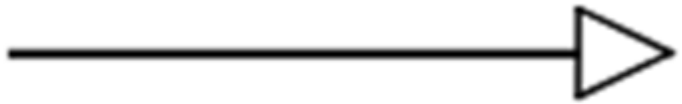
Przykład metody w klasie osoby (klasy Person):

+ dodaj : (a : int, b : int) : int

RELACJE POMIĘDZY KLASAMI

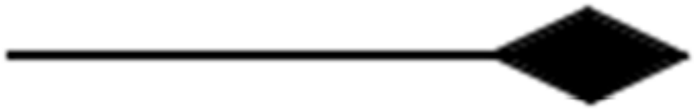


Asocjacja



Dziedziczenie

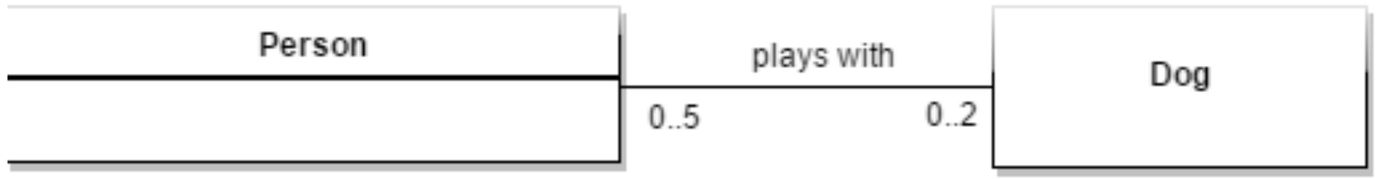
Agregacja



Kompozycja

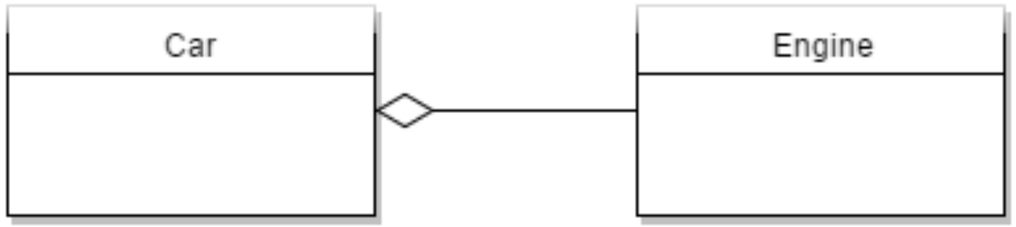
ASOCJACJA

- Jedna klasa używa drugiej lub ma dotyczące jej informacje
- Czasy życia obiektów związanych asocjacja nie są ze sobą powiązane

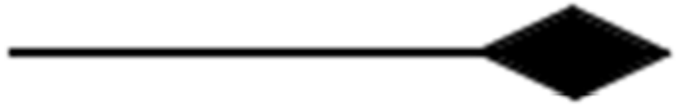


AGREGACJA

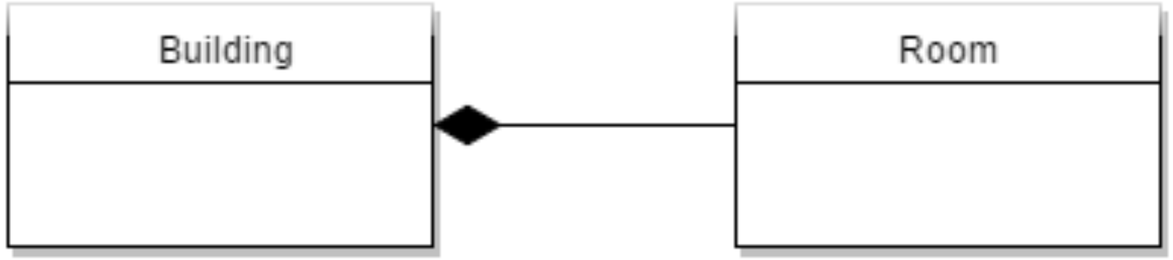
- Silniejsza wersja asocjacji
- Oznacza posiadanie jednaj klasy przez drugą



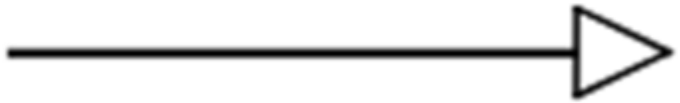
KOMPOZYCJA



- Wyraża związek „całość-część”.
- „Część” może być zaangażowana w jeden związek tego typu w danym czasie
- Czas życia jest ścisle powiązany - zniszczenie całości powoduje zniszczenie części

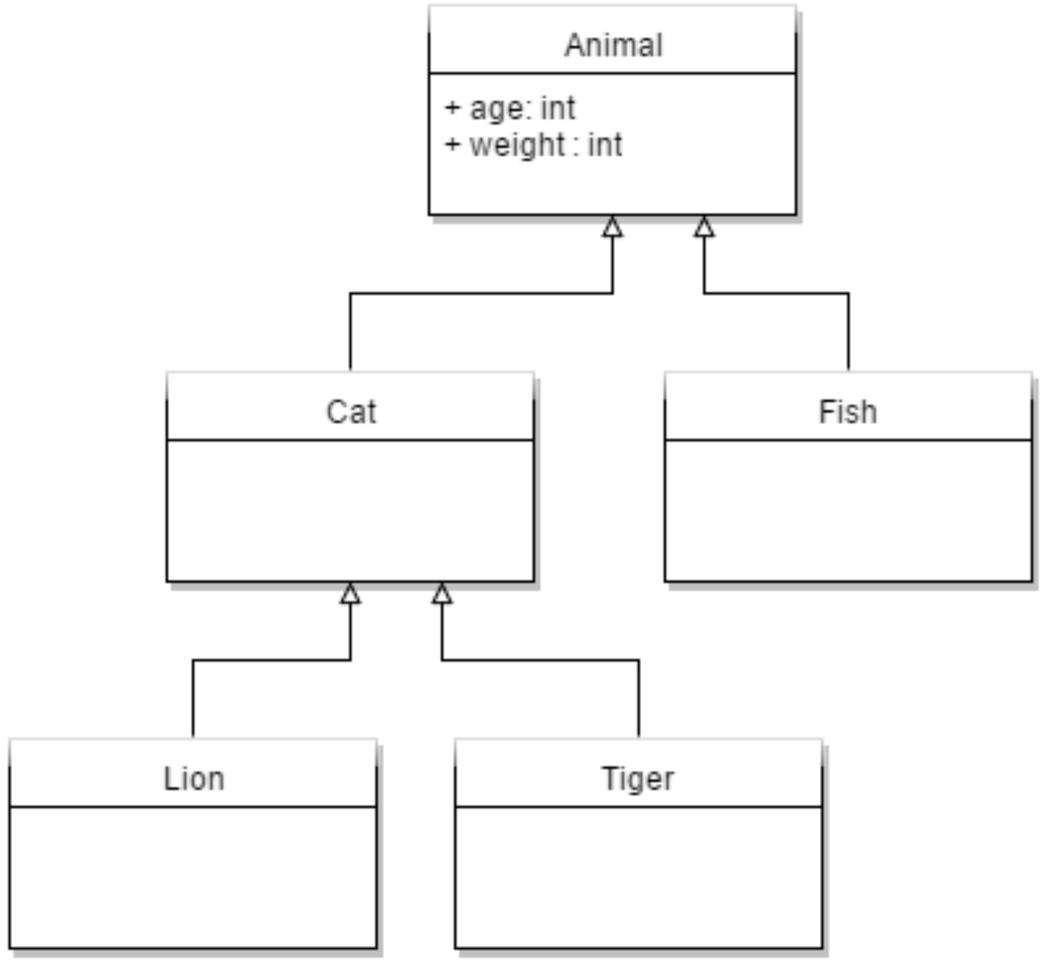


DZIEDZICZENIE



- Dziedziczenie jest sposobem wyrażania relacji specjalizacji/generalizacji, czyli relacją typu jest-czymś, np.: Ssak jest zwierzęciem.
- Relacja używana to wyciągania wspólnych cech klas.
- W UML dozwolone jest wielodziedziczenie
- W językach programowania różnie:
  - JAVA, C# nie pozwalają na wielodziedziczenie
  - C++ pozwala

DZIEDZICZENIE



MODELOWANIE BEHAWIORALNE

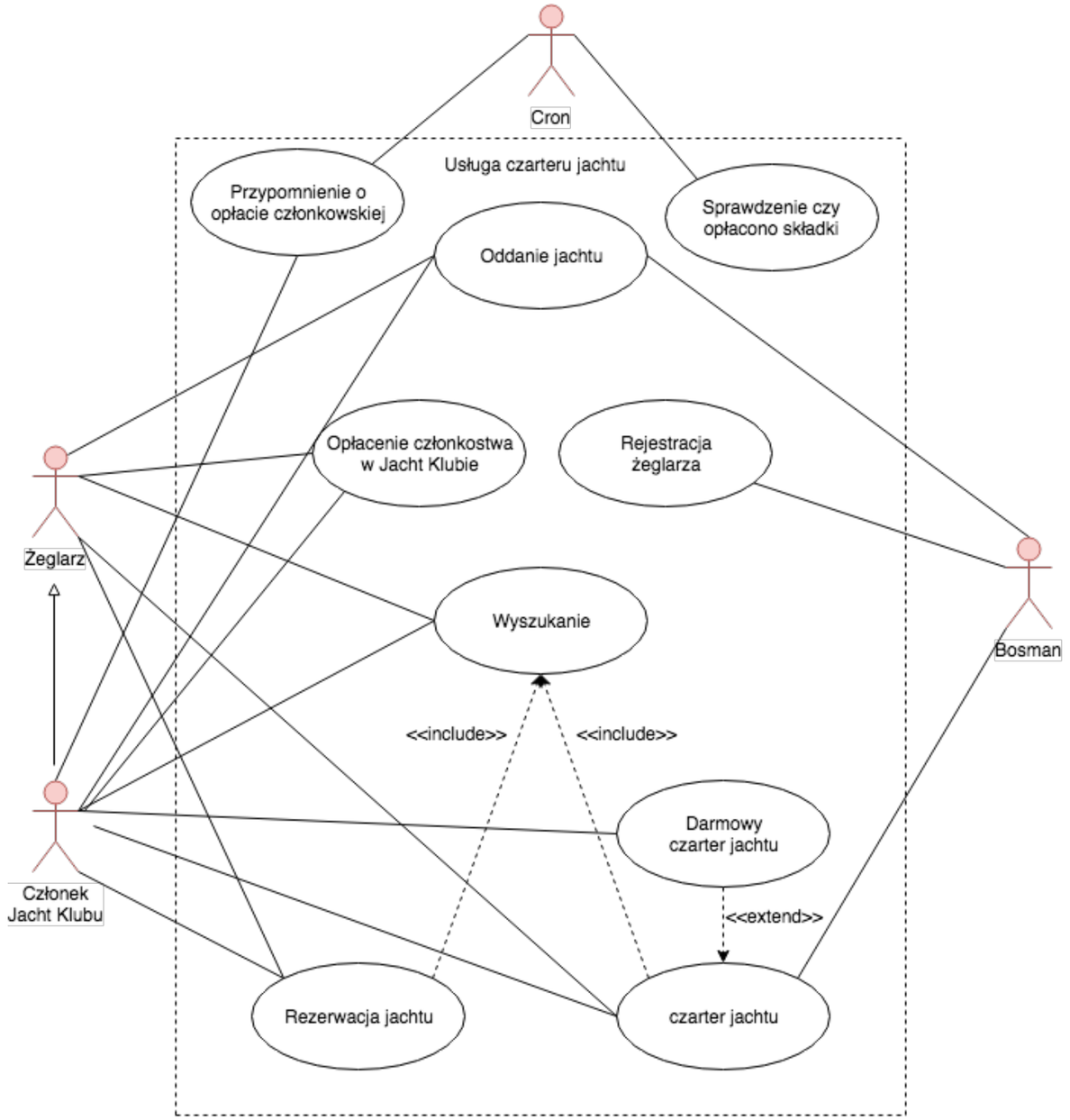
- Modelowanie behawioralne
  - diagram przypadków użycia
  - diagram czynności, procesów
  - diagram maszyny stanowej
  - diagramy interakcji (sekwencji, komunikacji, przeglądu interakcji)
  - diagram uwarunkowań czasowych

DIAGRAM PRZYPADKÓW UŻYCIA

- definiuje granice modelowanego systemu
- określa kontekst
- wymienia użytkowników systemu i jednostki zewnętrzne
- przedstawia funkcje dostępne dla użytkowników
- określa powiązania i zależności pomiędzy nimi

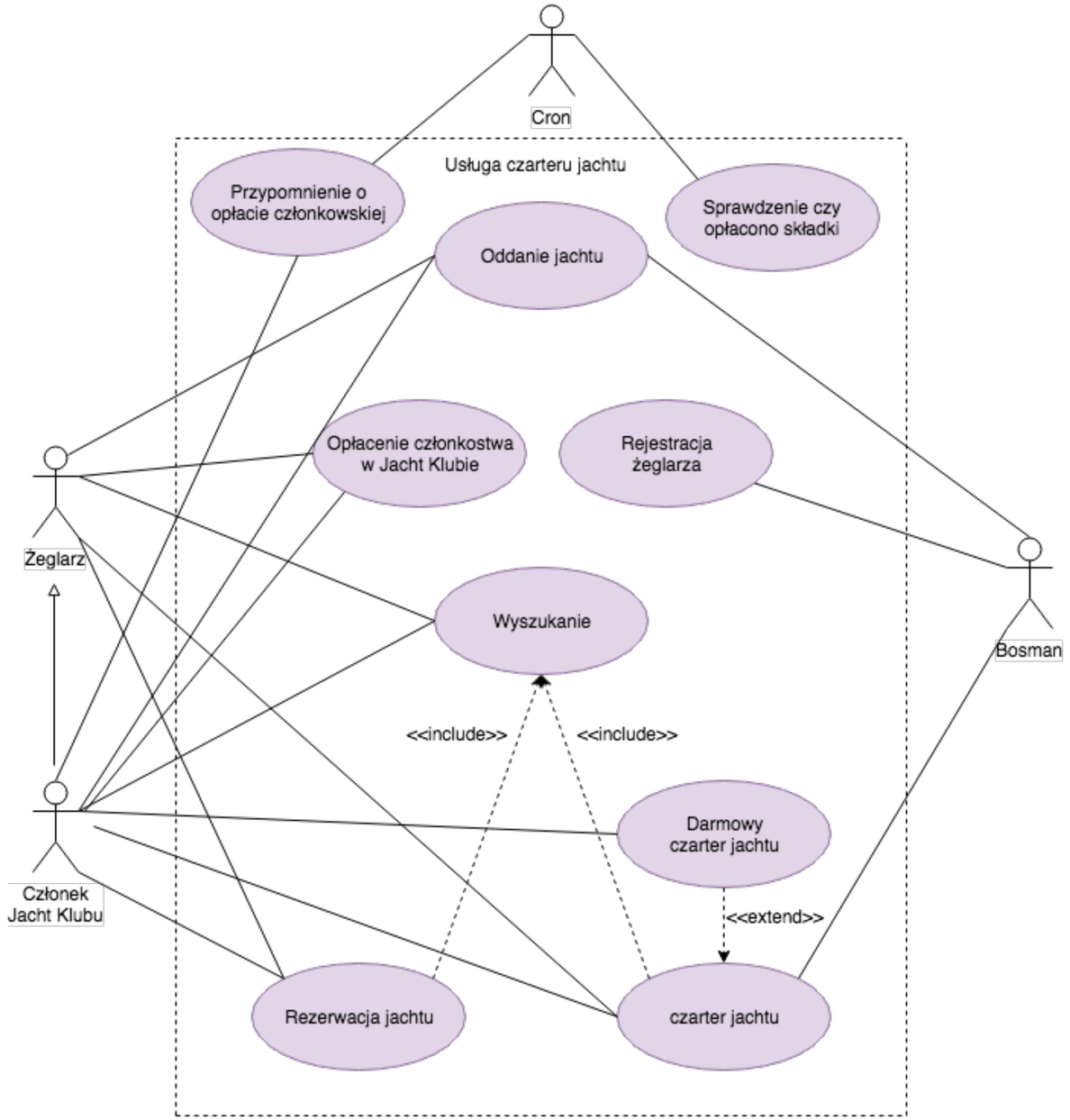
AKTOR

- Inicjuje wykonanie funkcji systemu
- Wymaga dostępu do systemu
- Reprezentuje perspektywę na system
- Jest osobą fizyczną, rolą w systemie lub systemem zewnętrznym



PRZYPADEK UŻYCIA

- Reprezentuje funkcję dostępną dla aktora
- Mogą być powiązane relacją:
  - Uszczegółowienia
  - Rozszerzania
  - Zawierania



PRZYKŁAD

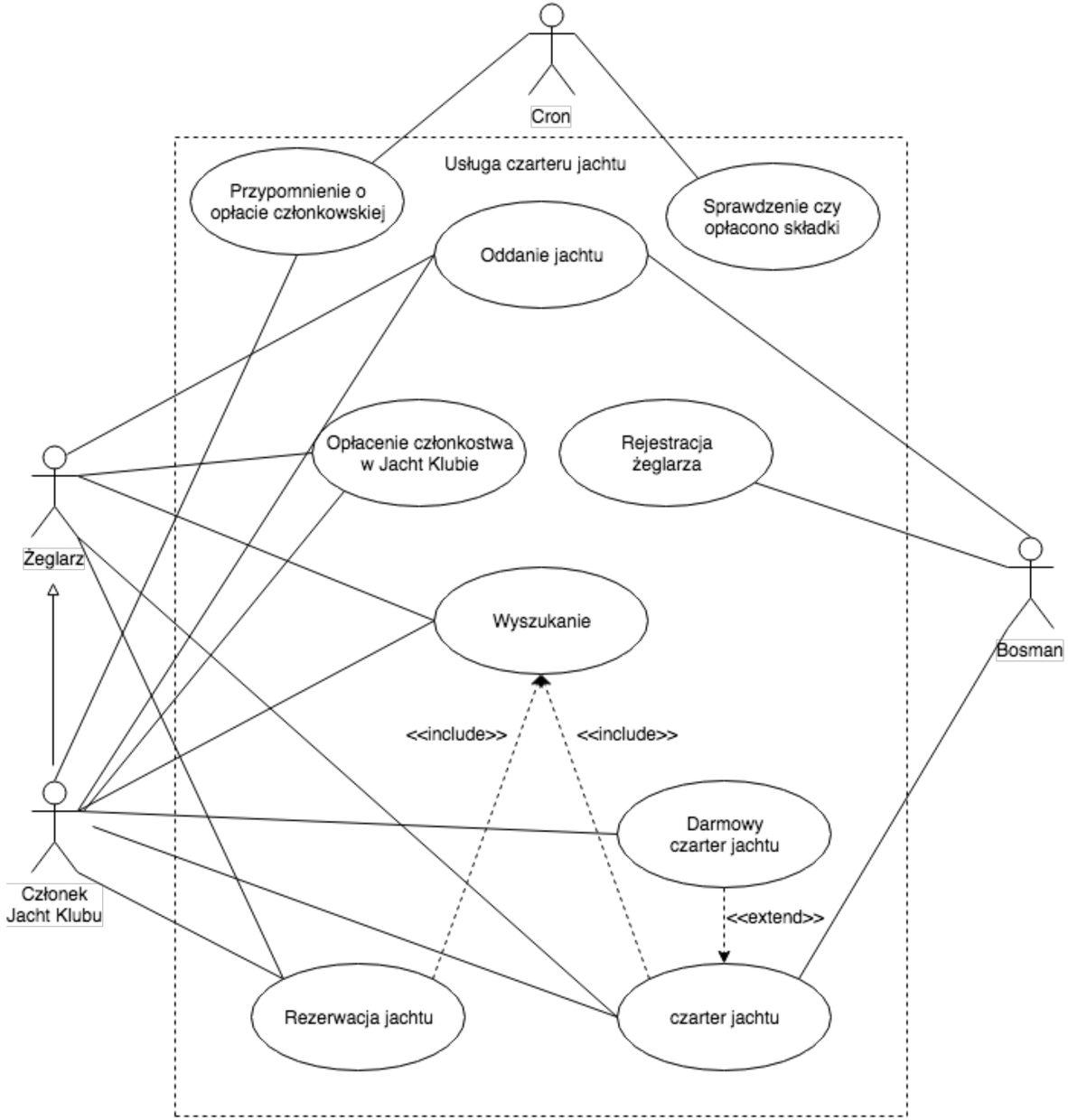
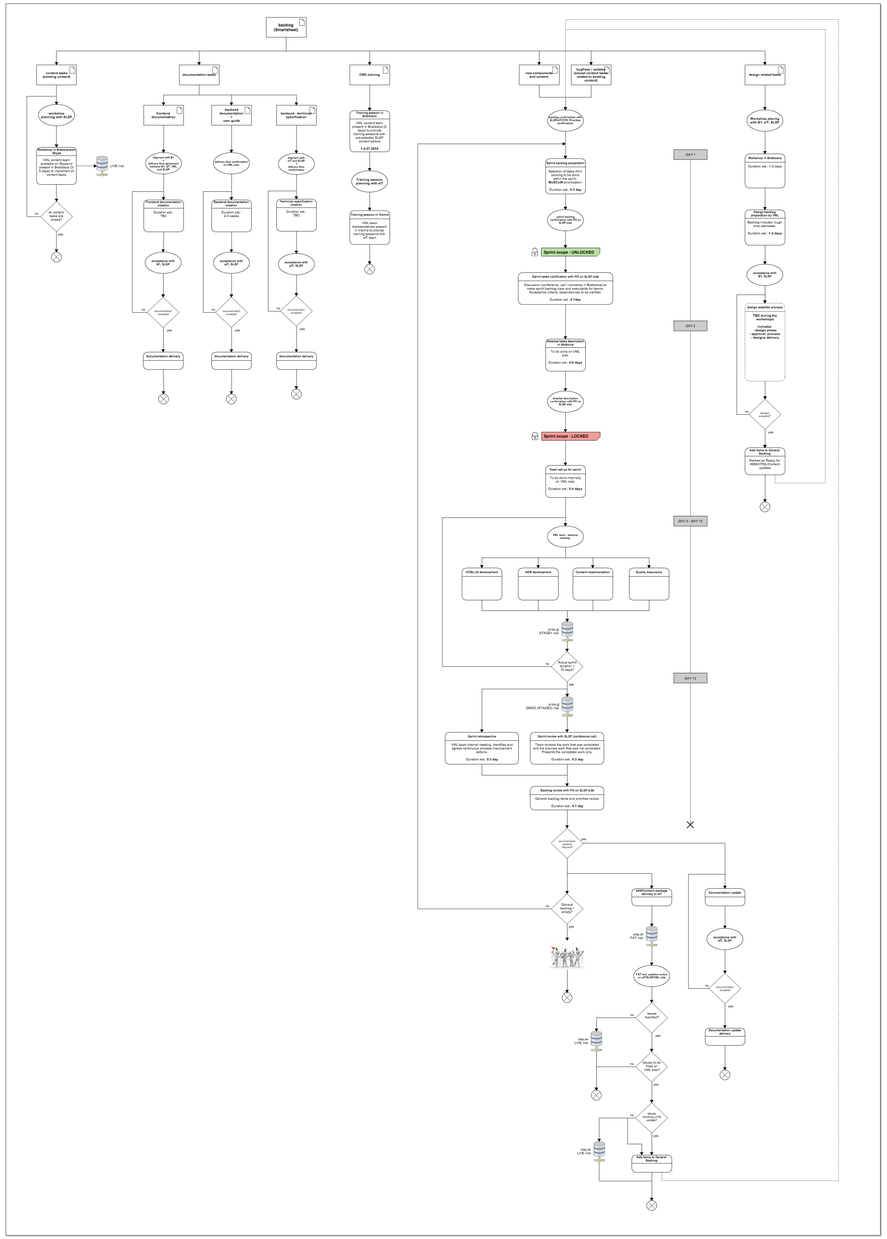

Cron
Usługa czarteru jachtu
Przypomnienie o opłacie członkowskiej
Sprawdzenie czy opłacono składki
Oddanie jachtu
Opłacenie członkostwa w Jacht Klubie
Rejestracja żeglarza
Żeglarz
Wyszukanie
Bosman
<<include>>
<<include>>
Darmowy czarter jachtu
<<extend>>
Członek Jacht Klubu
Rezerwacja jachtu
czarter jachtu

DIAGRAM PROCESU

- określa możliwe kroki i ścieżki procesu
- definiuje kolejność operacji w ramach procesu
- wskazuje zakres odpowiedzialności uczestników



ĆWICZENIE

Korzystając z utworzonych Use Case'ów zaprojektuj diagram, który zilustruje relacje między aktorami, przypadkami użycia i określi granice systemu.

Możesz wzbogacić diagram o dodatkowych aktorów i przypadki użycia - same nazwy ról i UC - nieuwzględnione w przygotowanej uprzednio dokumentacji.

- Narzędzie: https://www.draw.io/
- UML Quick Reference Guide: https://holub.com/uml/