Wybrane elementy praktyki projektowania oprogramowania
Wykład 14/15
Projektowanie obiektowe

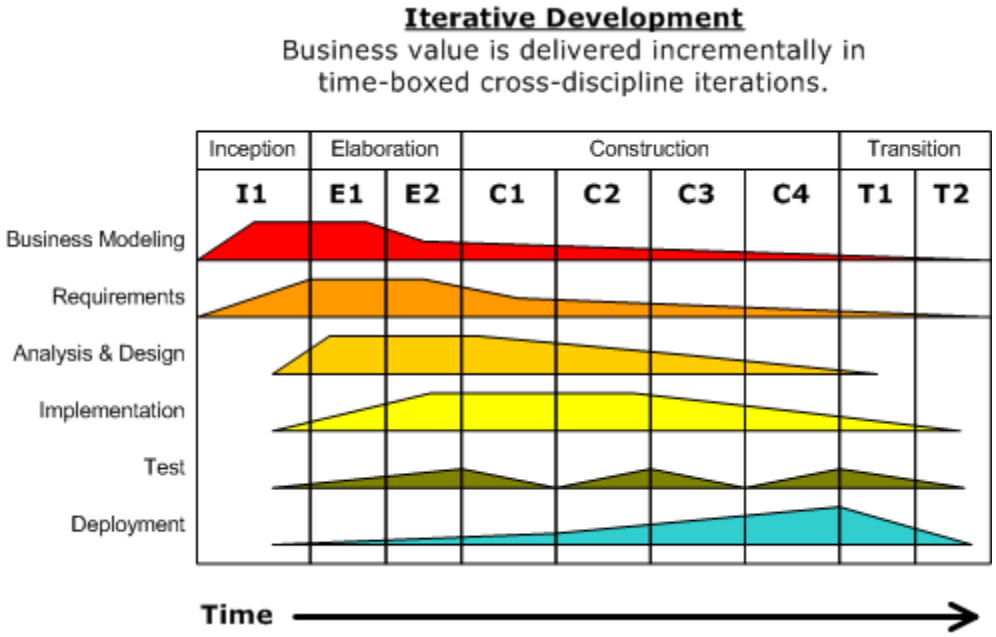
Wiktor Zychla 2019/2020

1 Spis treści

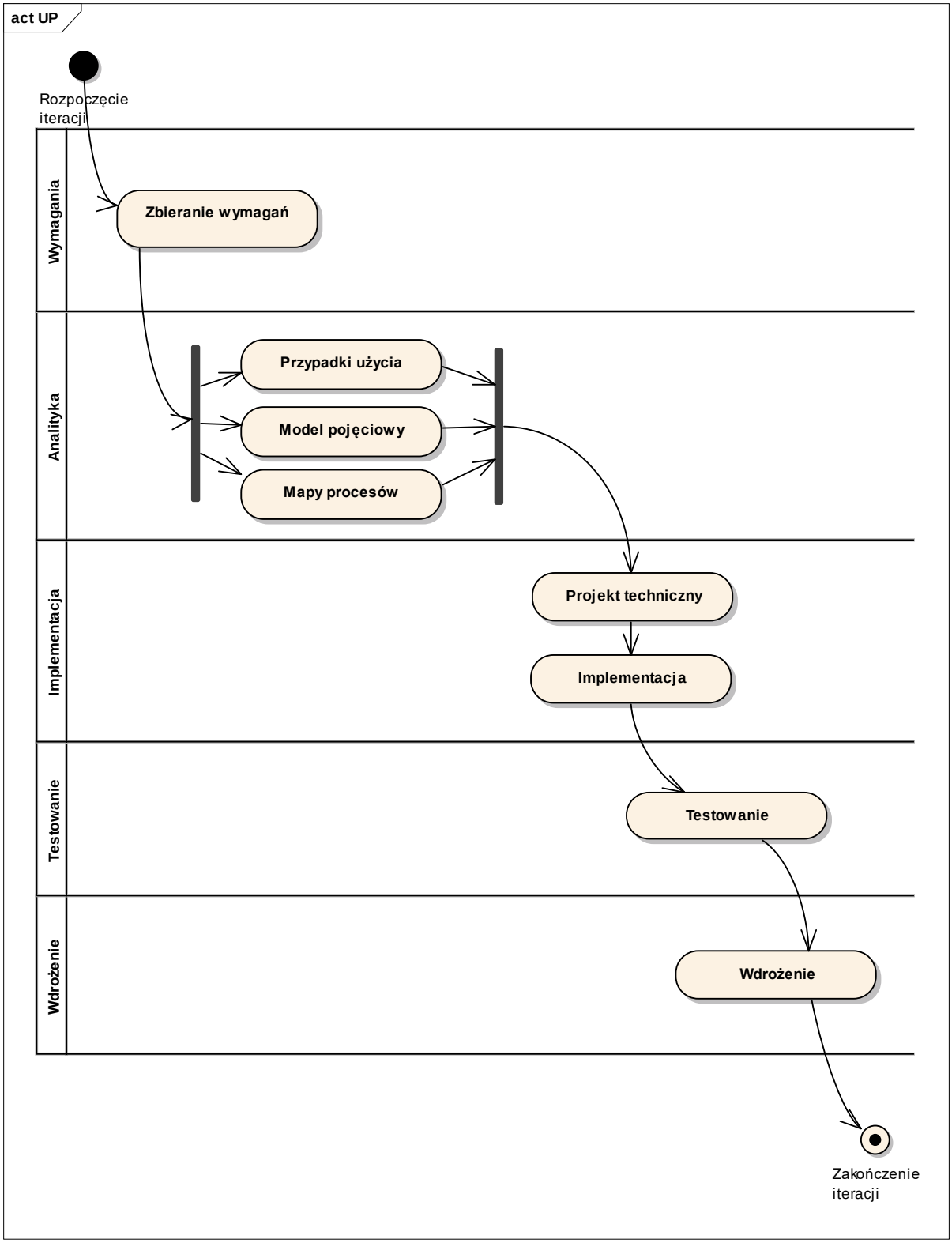
2 Unified Process

Rama organizacji procesu wytwarzania oprogramowania. Posiada wyodrębnione fazy inicjowania, projektowania, implementowania, testowania i wdrażania.

Na niemal wszystkie metodyki wytwarzania oprogramowania można patrzeć jak na warianty UP.



Rysunek 1 http://upload.wikimedia.org/wikipedia/commons/0/05/Development-iterative.gif



Rysunek 2 Diagram procesowy UP

3 Faza rozpoczęcia (Business Modelling)

Definicja. **Faza rozpoczęcia** = określenie zakresu, wizji i uwarunkowań biznesowych.

Typowe artefakty:

Wizja i analiza biznesowa	Opis **celów** i uwarunkowań biznesowych
Słowniczek	Podstawowa terminologia dziedzinowa
Prototyp	Udowodnienie poprawności rozwiązań technicznych; doprecyzowanie wizji
Plan pierwszej iteracji	
Specyfikacja dodatkowa	Lista dodatkowych wymagań mających istotny wpływ na architekturę
Plan zarządzania ryzykiem	(aktualizowane na bieżąco) scenariusze alternatywne – biznesowe, technologiczne, organizacyjne

4 Zbieranie wymagań (Requirements)

4.1 FURPS+ - obszary wymagań

Definicja. **Wymagania =** zdolności które system musi posiadać i ograniczenia do których musi się dostosować.

Uwaga! Wymagania ewolucyjne – zmieniają się w czasie.

http://www.ibm.com/developerworks/rational/library/4706.html#N100A7

Functional	Funkcjonalności, możliwości, bezpieczeństwo
Usability	Czynnik ludzki, pomoc, dokumentacja
Reliability	Awaryjność, odzyskiwanie, przewidywalność
Performance	Czas reakcji, przepustowość, dokładność, dostępność, wykorzystanie zasobów
Supportability	Dostosowanie, utrzymanie, konfiguracja, lokalizacja
*D*esign	Wszelkie ograniczenia projektowe (np. relacyjna baza danych)
*/*Implementation	Narzędznia, sprzęt, zasoby, standardy
*/*Interface	Interfejsy zewnętrznych systemów
*P*hysical	

Przykładowy formularz wymagań RUP od IBM (szczegółowy)

http://www.ibm.com/developerworks/rational/library/4710.html

Typowe problemy:

1. „Shopping cart" mentality – wszystko co tylko można bez świadomości kosztów
2. „All are equal" – brak priorytetyzacji
3. „Requirements can't be measured" – wymagania są niejasne albo niemierzalne

Przykłady wymagań narzuconych przez regulacje prawne:

- Wymagania dotyczące ochrony danych osobowych (GIODO)
  http://www.giodo.gov.pl/1520074/id_art/3910/j/pl/
  www.giodo.gov.pl/plik/id_p/285/j/pl/
- Wymagania Krajowych Ram Interoperacyjności (KRI)
  http://www.dziennikustaw.gov.pl/du/2012/526/1
- Rozporządzenie o Ochronie Danych Osobowych (RODO)
  https://giodo.gov.pl/pl/569/9276

SIWZ, Specyfikacja istotnych warunków zamówienia

4.2 S.M.A.R.T. – kryteria oceny wymagań

Jak ocenić sformułowane wymagania?

- Szczegółowy/prosty (**S**imple)
- Mierzalny (**M**easurable)
- Osiągalny/atrakcyjny (**A**chievable)
- Istotny/realistyczny (**R**elevant)
- Terminowy (**T**ime-specific)

Specific – 5W

- What: What do I want to accomplish?
- Why: Specific reasons, purpose or benefits of accomplishing the goal.
- Who: Who is involved?
- Where: Identify a location.
- Which: Identify requirements and constraints.

Measuareble

- How much?
- How many?
- How will I know when it is accomplished?
- Indicators should be quantifiable

Achievable

- How: How can the goal be accomplished?

Relevant

- Does this seem worthwhile?
- Is this the right time?
- Does this match our other efforts/needs?
- Are you the right person?
- Is it applicable in current socio- economic- technical environment?

Time-bound

- When?
- What can I do six months from now?
- What can I do six weeks from now?
- What can I do today?

5 Projektowanie analityczne – przypadki użycia (Analysis)

Definicja. **Przypadek użycia =** sekwencja prostych kroków opisująca interakcję między aktorem (użytkownikiem) a systemem.

Uwaga! Przypadki użycia należą do **wymagań funkcjonalnych**.

Uwaga! Przypadki użycia dokumentuje się w postaci tekstu, wspartego opcjonalnie diagramami przypadków użycia UML.

Aktor – byt charakteryzujący się zachowaniem (w tym sam system)

Aktor pierwszoplanowy – realizuje cele użytkownika (np. kasjer)

Aktor drugoplanowy – dostarcza informacji (np. system kart płatniczych) (opis zewnętrznych interfejsów i protokołów)

5.1 Dokumentacja:

Nieformalna (*brief*) – zwięzłe streszczenie o długości jednego akapitu, podstawowy scenariusz sukcesu

Obsłuż sprzedaż – klient staje przy kasie z produktami, które chce kupić. Kasjer korzysta z systemu kasowego w celu odnotowania każdego produktu. System wyświetla informacje na temat poszczególnych produktów oraz sumę do zapłaty. Klient podaje dane pozwalające określić sposób płatności. Kasjer przyjmuje zapłatę, system aktualizuje stan magazynu, klient otrzymuje paragon.

Pełna (*fully dressed*) – wszystkie kroki i warianty opisane szczegółowo

- **Poziom** – cel użytkownika lub podprocedura
- **Interesariusze** i ich cele
- **Warunki początkowe**
- **Warunki zakończenia** (powodzenia)
- **Główny scenariusz sukcesu** – brak obsługi błędów, wyrażeń warunkowych
- **Rozszerzenia** – obsługa błędów i sytuacji nietypowych
- **Kroki**:
  - **Zdarzenie** inicjujące przypadek użycia
  - **Interakcja** między aktorami
  - **Walidacja**
  - **Zmiana stanu** systemu (np. zapisanie danych)
- **Dodatkowe wymagania**
- **Technologia i format danych**

5.1.1 Przykład

https://jira.atlassian.com/secure/attachment/48985/Use+case+POS.pdf

5.1.2 Przykład – licytuj towar

Nazwa: Licytuj towar
Numer: 1
Twórca: Jan Kowalski
Poziom ważności: Wysoki
Typ przypadku użycia: Ogólny, niezbędny

Aktorzy: Uczestnik aukcji [kupujący]
Krótki opis: Licytacja wskazanego towaru
Warunki wstępne: Uczestnik aukcji posiada niezablokowane konto
Warunki końcowe: Oferta została zarejestrowana lub wyświetlony został komunikat o błędzie a stan systemu nie uległ zmianie
Główny scenariusz sukcesu:
1) Uczestnik aukcji wskazuje aukcję, w której chce uczestniczyć
2) System wyświetla formularz do wpisania oferty
3) Uczestnik aukcji wpisuje ofertę, a następnie wybiera opcję licytuj
4a) System rejestruje ofertę i informuje o tymUczestnika aukcji
5) Następuje rozszerzenie aukcji o przypadek Finalizuj transakcję
Alternatywne przepływy zdarzeń:
4b) Jeżeli w kroku 3) Uczestnik aukcji wprowadził kwotę niezgodną z regułami licytacji, system informuje o błędzie i następuje przejście do kroku 2)
Wyjątki w przepływach
4c) Jeżeli z powodu awarii technicznej lub zakończenia aukcji system nie może zarejestrować
oferty, informuje o tym Uczestnika aukcji i następuje zakończenie przypadku
Specjalne wymagania: brak
Notatki i kwestie: Po zakończeniu aukcji system informuje kupującego i sprzedającego o wyniku licytacji W dowolnym momencie Uczestnik aukcji może zrezygnować z licytacji i następuje zakończenie przypadku

5.1.3 Przykład – finalizuj transakcję

Nazwa: Finalizuj transakcję
Numer: 2
Twórca: Jan Kowalski
Poziom ważności: Wysoki
Typ przypadku użycia: Ogólny, niezbędny
Aktorzy: Uczestnik aukcji [kupujący], oraz Uczestnik aukcji [sprzedający]
Krótki opis: Finalizacja rozstrzygniętych aukcji
Warunki wstępne:
1) Uczestnik aukcji posiada niezablokowane konto
2) Uczestnik aukcji [sprzedający] był oferentem aukcji
3) Uczestnik aukcji [kupujący] wygrał licytację
Warunki końcowe:
Transakcja została zakończona lub aukcja została unieważniona
Główny scenariusz sukcesu:
1) System informuje Uczestników aukcji o zakończeniu licytacji
2a) Kupujący określa sposób płatności oraz wybiera formę dostarczenia towaru
3) System wysyła do sprzedającego informację o sposobie płatności oraz wybranej przez kupującego formie dostarczenia towaru
4) Sprzedający wystawia ocenę kupującemu
5) W przypadku negatywnej oceny system wysyła informację do Administratora
6) Kupujący wystawia ocenę sprzedającemu
7) W przypadku negatywnej oceny system wysyła informację do administratora
8) Administrator w przypadku uzasadnionych skarg uczestników transakcji i (lub) naruszenia regulaminu może unieważnić transakcję
Alternatywne przepływy zdarzeń:
2b) Jeżeli w ciągu 3 dni od zawarcia transakcji nie poinformował sprzedawcy o wyborze sposobu
płatności, sprzedawca może unieważnić transakcję
Specjalne wymagania: brak
Notatki i kwestie: Pomiędzy kolejnymi zdarzeniami mogą wystąpić kilkudniowe odstępy czasowe

Kroki 6) i 7) mogą wystąpić przed krokami 4) i 5)

5.2 Poszukiwanie przypadków użycia

Najbardziej oczywiste - określenie aktorów i celów.

Kasjer	• Przetwarzanie sprzedaży • Wkładanie pieniędzy do kasy • Wypłacanie pieniędzy z kasy • …
Kierownik	• Uruchamianie systemu • Wyłączanie systemu …
Administrator systemu	• Zarządzanie użytkownikami • Określanie zasad bezpieczeństwa …

Dodatkowe przypadki użycia:

1. Kto uruchamia i zatrzymuje system
2. Kto administruje systemem
3. Kto zarządza użytkownikami
4. Czy działanie systemu samoistnie zmienia się z upływem czasu
5. Kto ocenia działanie i wydajność
6. Jak obsługuje się aktualizacje
7. Itd… wynikają wprost z wymagań

5.3 Kryteria oceny przypadków użycia

Test EBP (Elementary Business Process) – Zadanie wykonywane przez jedną osobę w jednym miejscu i określonym czasie w odpowiedzi na pewne zdarzenie biznesowe. Zadanie prowadzi do uzyskania mierzalnej wartości biznesowej. Po jego wykonaniu dane są w spójnym stanie.

Test rozmiaru – intuicyjnie zbyt mały lub zbyt duży

Test szefa – Szef pyta pracownika „Co Pan robił przez cały dzień”. Pracownik „Logowałem się!”

1. Negocjuj umowę z dostawcą – nie przechodzi testu rozmiaru (za duży)
2. Obsłuż zwroty – przechodzi testy
3. Zaloguj się – nie przechodzi testu szefa
4. Przesuń pionek na planszy – nie przechodzi testu rozmiaru (za mały)

6 Projektowanie analityczne – modele pojęciowe (Analysis)

Definicja. **Model dziedziny (model pojęciowy, konceptualny) =** przedstawienie **pojęć** reprezentujących byty ze świata rzeczywistego („użytkownik”, „drukarka”, „faktura”, „ocena”), istotnych dla danej dziedziny oraz **relacji** między nimi („ma”, „zjada”, „lubi”, „rozpoczyna”).

6.1 Tworzenie modelu pojęciowego

6.1.1 Metoda „fraz rzeczownikowych”

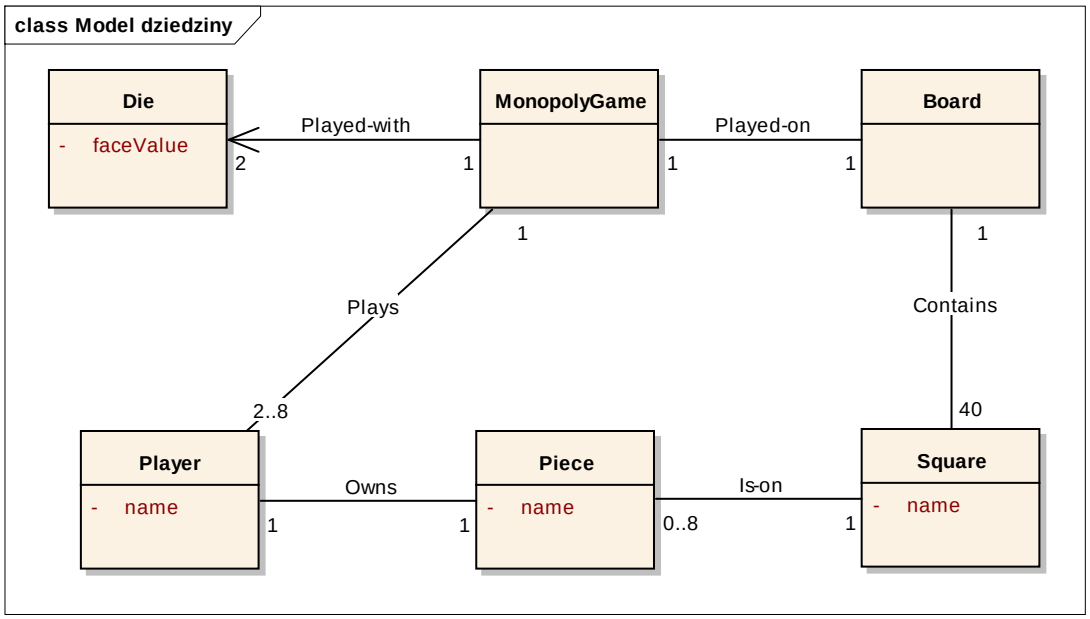
Wykorzystuje się przypadki użycia, np.

Główny scenariusz sukcesu:
1) **Uczestnik aukcji** wskazuje **aukcję**, w której chce uczestniczyć
2) System wyświetla formularz do wpisania **oferty**
3) Uczestnik aukcji wpisuje ofertę, a następnie wybiera **opcję** licytuj
4a) System rejestruje ofertę i informuje o tymUczestnika aukcji
5) Następuje rozszerzenie aukcji o przypadek Finalizuj transakcję

Główny scenariusz sukcesu:
1) System informuje Uczestników aukcji o zakończeniu **licytacji**
2a) Kupujący określa sposób **płatności** oraz wybiera formę **dostarczenia towaru**
3) System wysyła do sprzedającego **powiadomienie** o sposobie płatności oraz wybranej przez kupującego formie dostarczenia towaru
4) Sprzedający wystawia **ocenę** kupującemu
5) W przypadku negatywnej oceny system wysyła informację do Administratora
6) Kupujący wystawia ocenę sprzedającemu
7) W przypadku negatywnej oceny system wysyła informację do administratora
8) Administrator w przypadku uzasadnionych **skarg** uczestników transakcji i (lub) naruszenia regulaminu może unieważnić transakcję

6.2 Przykład z podręcznika

Do reprezentacji modelu pojęciowego używa się diagramów klas UML (będziemy o tym mówić na kolejnym wykładzie).



Uwaga! To nie jest jeszcze diagram klas tylko właśnie diagram modelu pojęciowego. Diagram modelu pojęciowego próbuje jedynie uchwycić pojęcia i relacje między nimi, nie bardzo przejmując się tym czy i jak z **pojęć** powstaną **klasy** (obiekty).

Diagram klas z kolei jest elementem projektu architektury i stanowi część prac implementacyjnych i byłby o wiele bardziej szczegółowy. Obejmowałby m.in. kwalifikatory dostępu i akcesory, relacje dziedziczenia i implementowania interfejsów. Z kolei asocjacje na diagramie klas mają inne znaczenie niż na diagramie pojęć: na diagramie pojęć asocjacja określa związek (jakiś) między pojęciami. Na diagramie klas asocjacja oznacza istnienie relacji między klasami (klasa ma pole typu takiego jak klasa na którą wskazuje asocjacja).

6.3 Przykład z życia

Demo: Fragment rzeczywistego dokumentu analitycznego.

Cechy samego oddziału (nie dziedziczone z grupy oddziałów):

- **rok szkolny otwarcia oddziału -** można to pamiętać jako zwykły rok np. dla roku szkolnego 2007/2008 pamiętać 2007.
- **poziom początkowy** ze słownika (domyślnie 1 w szkołach, w przedszkolach bez domyślnego).
  - dwulatki
  - trzylatki
  - czterolatki
  - pięciolatki
  - 0
  - 1
  - 2
  - 3
  - 4
  - 5
  - 6
  - 7
  - 8
  - 9 (bo mamy 9-letnią szkołę baletową)

  nie przewidujemy poziomu mieszanego (tak jest teraz w arkuszu) ponieważ będziemy mieli mechanizm „częściowych" oddziałów. Jak ktoś będzie chciał koniecznie opisać oddział mieszany, to opisze 0.5 oddziału 3-latków i 0.5 oddziału 4-latków.
- **Poziom końcowy –** z tego samego słownika, co powyżej (w podstawówce to będzie najczęściej 6, w trzyletniej, semestralnej szkole dla dorosłych również 6, które jednak w SP oznaczają 6 lat, a w szkole dla dorosłych 6 semestrów, czyli trzy lata)
- **„trwanie poziomów"** – rzadko, ale zdarza się, że oddziały nie zmieniają poziomu z biegiem czasu. Tak się zdarza np. w oddziałach specjalnych, w których uczniowie są 2 lata w jednej klasie, po czym dopiero uzyskują promocję do następnej klasy. Aby w takich sytuacjach nie mieć kłopotu z wyznaczeniem poziomu oddziału i móc

zapamiętać, że oddział pozostaje np. na poziomie 1 przez dwa lata należy zapamiętać dla każdego poziomu liczbę okresów jego trwania. Domyślnie okres trwania dla każdego poziomu to 1, ale należy umożliwić dowolną tego zmianę. Wpisanie okresu trwania 100 na poziomie 1 oznacza w praktyce, że oddział nie zmienia nigdy poziomu. Trwanie poziomów inne niż 1 będzie się w praktyce zdarzało bardzo rzadko. Wprowadzanie i modyfikacja tych danych powinna być zatem dość głęboko ukryta, aby nie przeszkadzała typowym użytkownikom (np. po wciśnięciu dodatkowego przycisku umieszczonego przy liście typowych atrybutów oddziału).

- **poziom (klasa) –** Pole wyliczane na życzenie lub automatycznie 1 IX i 1 II (nie ma możliwości, aby poziom oddziału/semestru zmienił się innego dnia). Wykorzystywany na zestawieniach i w budowie widocznego identyfikatora oddziału. Poziom nieokreślony mógłby być wyróżnikiem oddziału archiwalnego .
- **status**
  - projektowany
  - istniejący
  - archiwalny

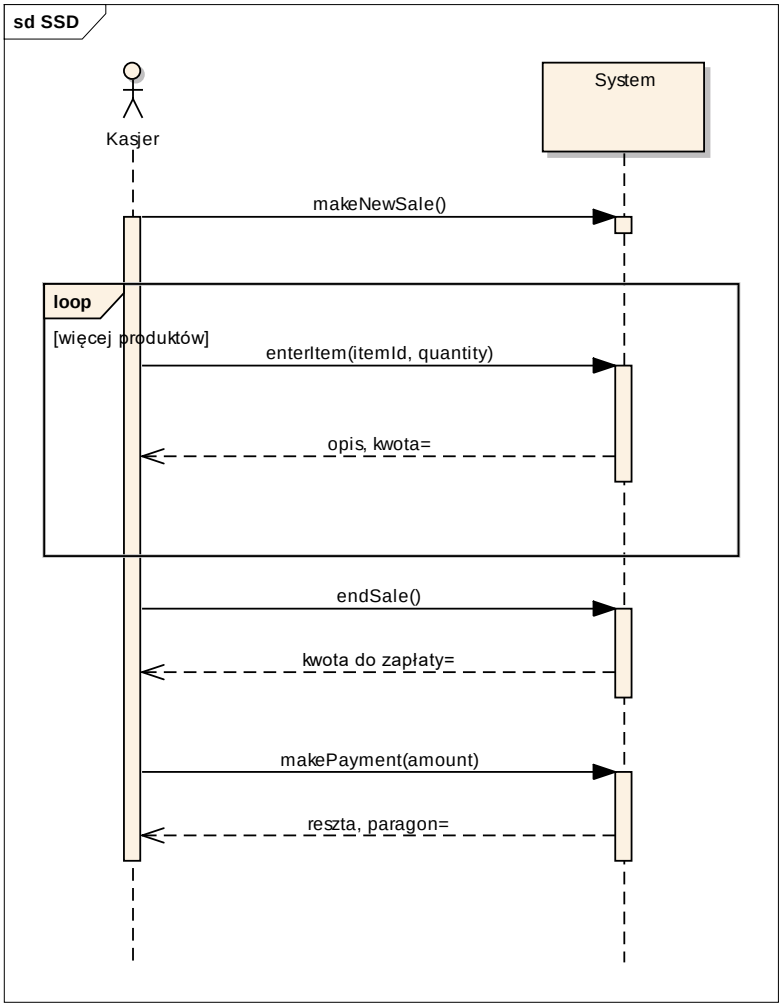
  Pole wyliczane równocześnie z poziomem. Oddział, którego data założenia jest późniejsza od bieżącej, jest projektowany. Oddział, którego poziom jest większy od maksymalnego jest archiwalny

ltd..

7 Projektowanie analityczne – mapy procesów (Analysis)

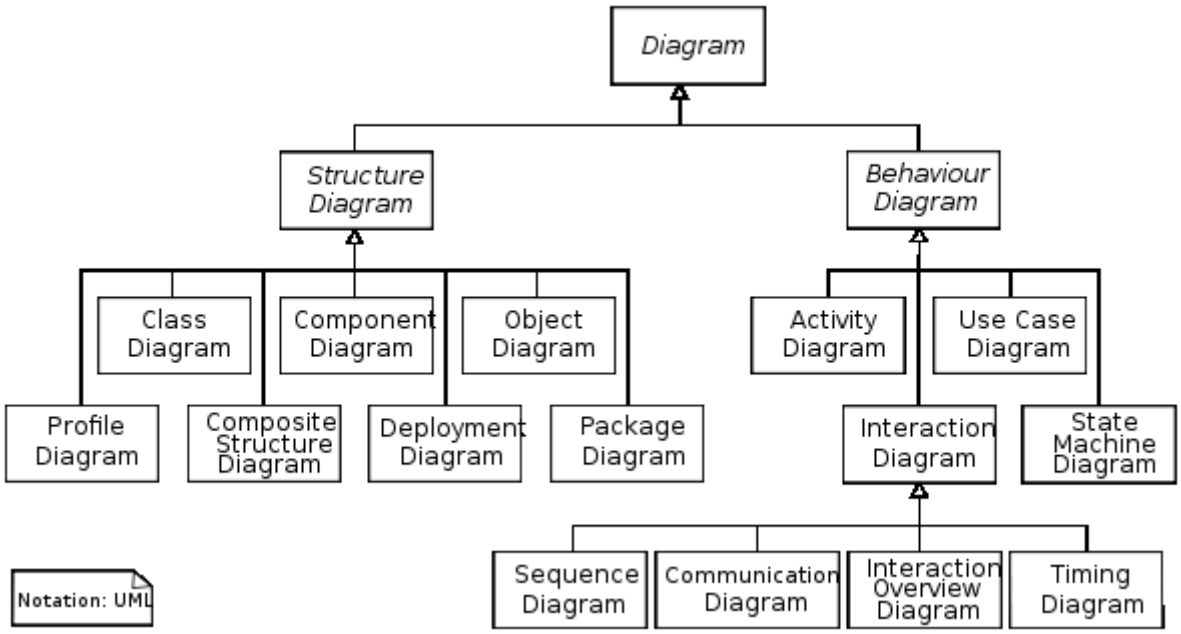
7.1 Diagramy czynności i sekwencji jako uzupełnienie przypadków użycia

Systemowy Diagram Sekwencji - zdarzenia systemowe w jednym, głównym scenariuszu przypadku użycia.



Demo: przykład procesu biznesowego z rzeczywistego dokumentu projektowego

8 Wprowadzenie do UML



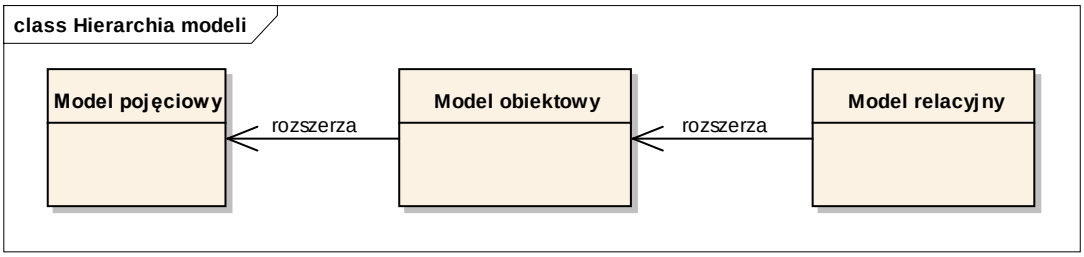
Dwie rodziny diagramów - diagramy **struktur** i diagramy **zachowań** (dynamiki):

- Diagramy struktur – służą do dokumentowania statycznych elementów systemu i relacji/powiązań między nimi
- Diagramy zachowań - służą do dokumentowania dynamicznych elementów systemu np. procesów/algorytmów/przypadków użycia

9 Diagramy klas

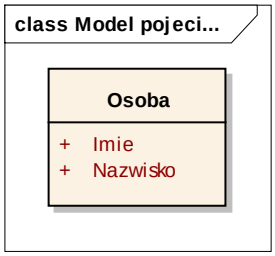
9.1 Hierarchia modeli

Formalnie w UML występuje Diagram klas (Class Diagram). Ale ten sam formalizm służy do reprezentacji **trzech** różnych typów diagramów, z których każdy występuje w innej fazie projektowania.

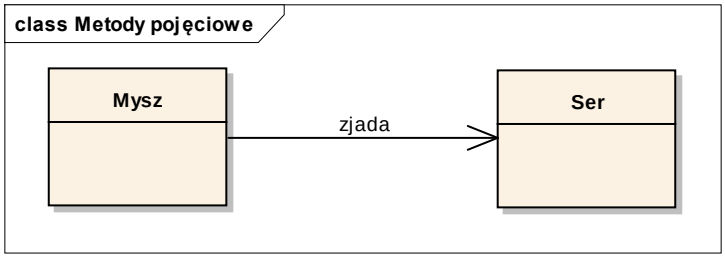


9.1.1 Diagram modelu pojęciowego

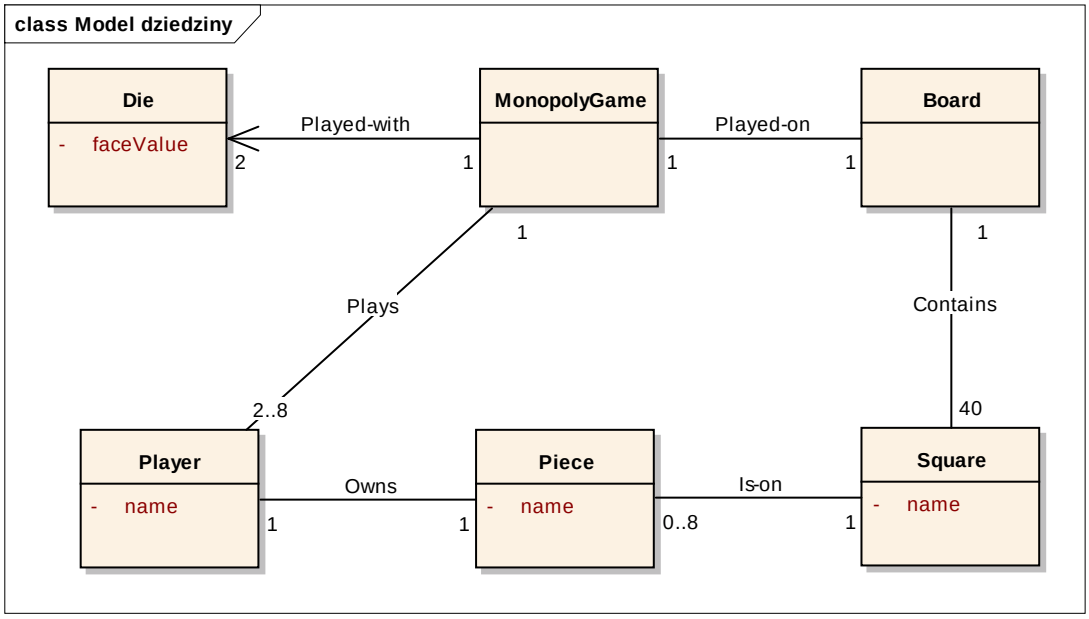
- Jest elementem projektu analitycznego
- Służy ustaleniu wspólnego języka w projekcie
- Służy weryfikacji podstawowych elementów struktury dziedziny projektu
- Pojęcia i atrybuty (tylko publiczne)



- Asocjacje (relacje) między pojęciami („ma”, „używa”, „płaci się za pomocą”)
- Asocjacje mogą być skierowane, wtedy kierunek strzałki i jej etykieta powinien być tak dobrany żeby można było „przeczytać” zdanie



- Brak dziedziczenia i innych ograniczeń specyficznych dla struktury stricte obiektowej
- Brak metod (!)

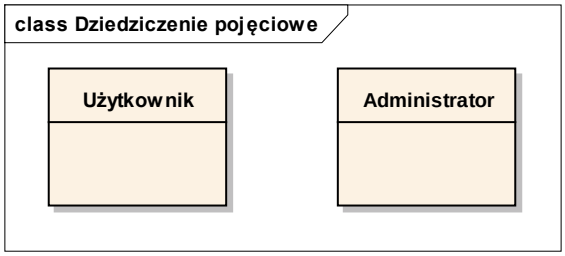


9.1.2 Diagram modelu obiektowego (diagram klas)

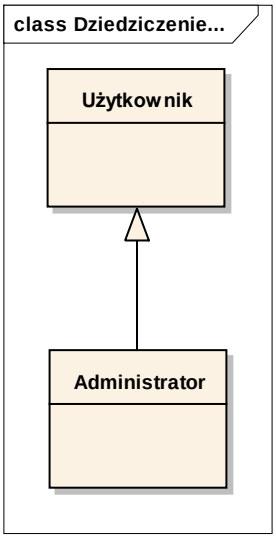
- Jest elementem projektu architektury
- Punktem wyjścia jest diagram modelu pojęciowego
- Na etapie projektowania obiektowego należy refaktoryzować model pojęciowy do stanu, w którym pojęcia reprezentowane są przez klasy (w rozumieniu obiektowym)
- Refaktoryzacja polega na:
    - **Usuwaniu** zbędnych pojęć, które są reprezentowane przez jedną i tę samą klasę (na przykład pojęcia Użytkownik i Administrator staną się jedną i tą samą klasą)
    - **Dodawaniu** nowych klas (na przykład tam gdzie do reprezentacji relacji potrzebna jest pomocnicza klasa)
    - **Rozróżnianiu** atrybutów publicznych, prywatnych, statycznych itd.
    - **Wprowadzaniu** metod do interfejsów klas
    - **Zamienianiu** wszystkich relacji z diagramu modelu pojęciowego na relacje występujące w świecie obiektowym:
        - asocjacja,
        - agregacja,
        - kompozycja,
        - dziedziczenie,
        - implementowanie interfejsu
        - metoda obiektu (przyjmująca parametr lub zwracająca wartość)

    Po tej fazie zamiany relacji, na diagramie klas nie może pozostać żadna asocjacja, której nie da się zaimplementować w języku obiektowym

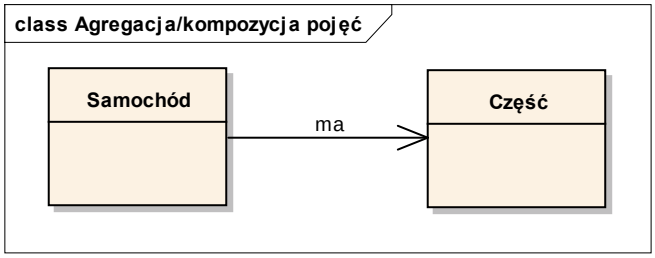
Przykład 1:



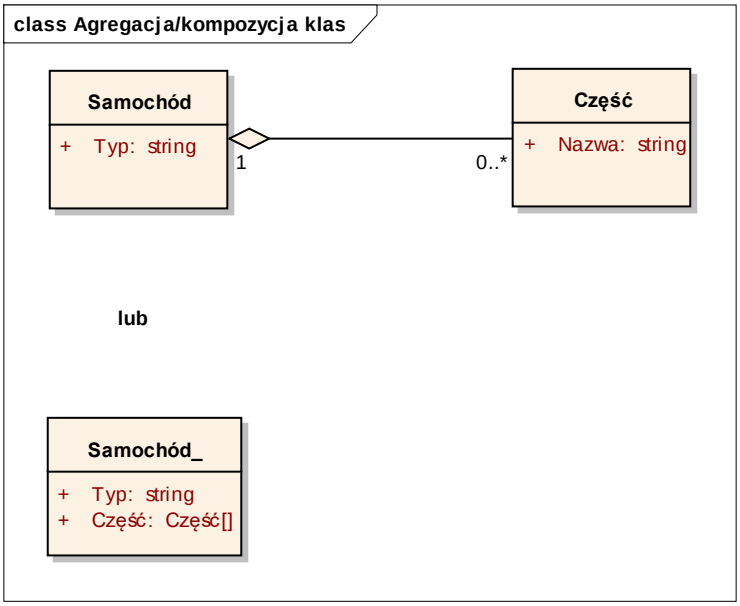
VS



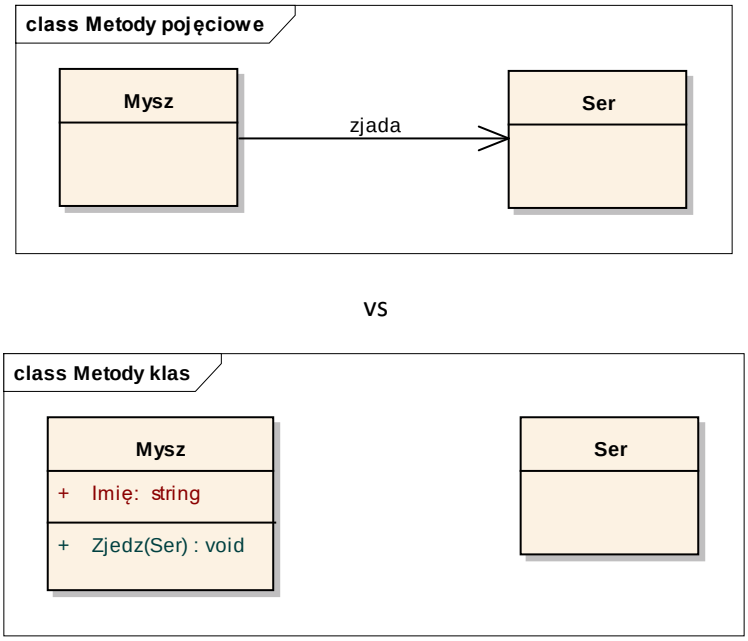
Przykład 2:



VS



Przykład 3:



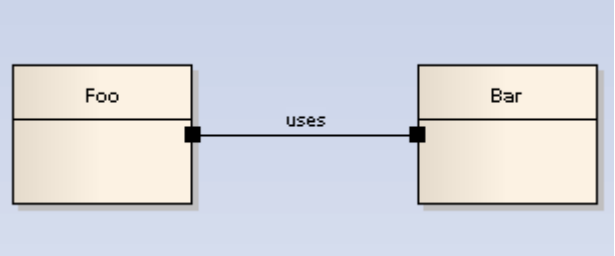
9.1.3 Diagram modelu implementacyjnego (relacyjnego)

- To diagram reprezentujący strukturę relacyjnej bazy danych
- Reprezentuje fizyczną strukturę właściwą do utrwalania obiektów
- Punktem wyjścia jest diagram klas
- Podczas refaktoryzacji usuwa się z diagramu klas wszystkie te relacje, których nie da się reprezentować w świecie relacyjnym
    - Nie ma metod
    - Nie ma dziedziczenia, zamiast tego wybiera się jeden ze sposobów implementacji dziedziczenia
        - Table-per-concrete-type
        - Table-per-hierarchy

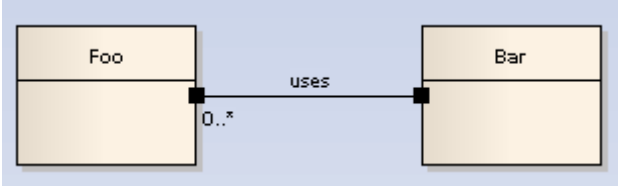
        - Table-per-type

9.2 Jeszcze o formalizmie diagramów klas - klasy i asocjacje

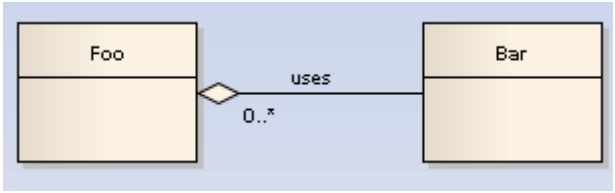
- Zależności (strzałka przerywana) – brak informacji o rodzaju zależności, może być:
    - Tworzy
    - Wykorzystuje (zmienna lokalna)
    - Wykorzystuje (parametr metody)
    - Nadklasa lub interfejs
- Nazwy asocjacji



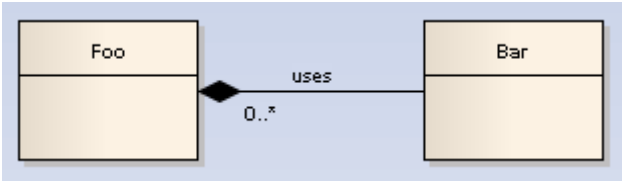
- Liczebność : 1, 1..*, 0..1, *, 0..*, n, 1..n, 0..n, n..m, n..*



- Agregacja vs kompozycja
    - Agregacja – luźniejsza

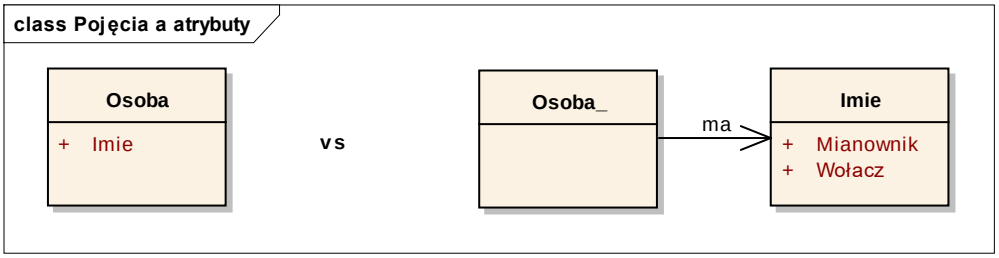


    - Kompozycja - ściślejsza
        - Instancja reprezentująca część może należeć tylko do jednej instancji złożonej
        - Czas życia części jest powiązany z czasem życia całości

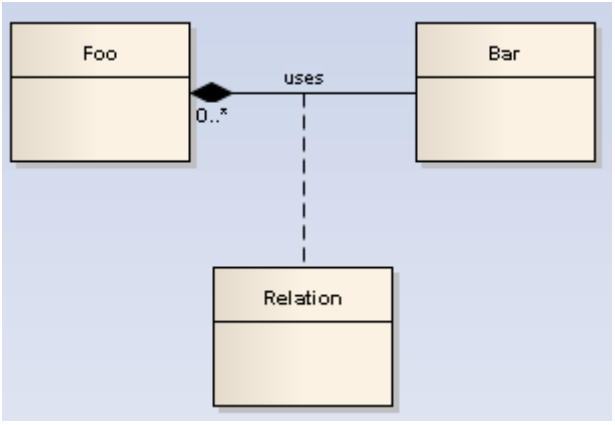


9.3 Składowe

- Składowa prywatna, publiczna, chroniona, stała, statyczna, kolekcja, atrybut pochodny
- Metoda prywatna, publiczna, chroniona, internal, abstrakcyjna, statyczna, konstruktor, parametry
- Atrybut wpisany vs asocjacja – kiedy używać? Atrybut: typ prosty, asocjacja do typu złożonego

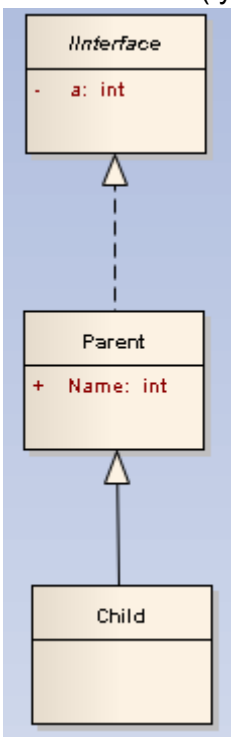


- Klasa asocjacyjna – do modelowania relacji wiele-wiele



9.4 Dziedzieczenie

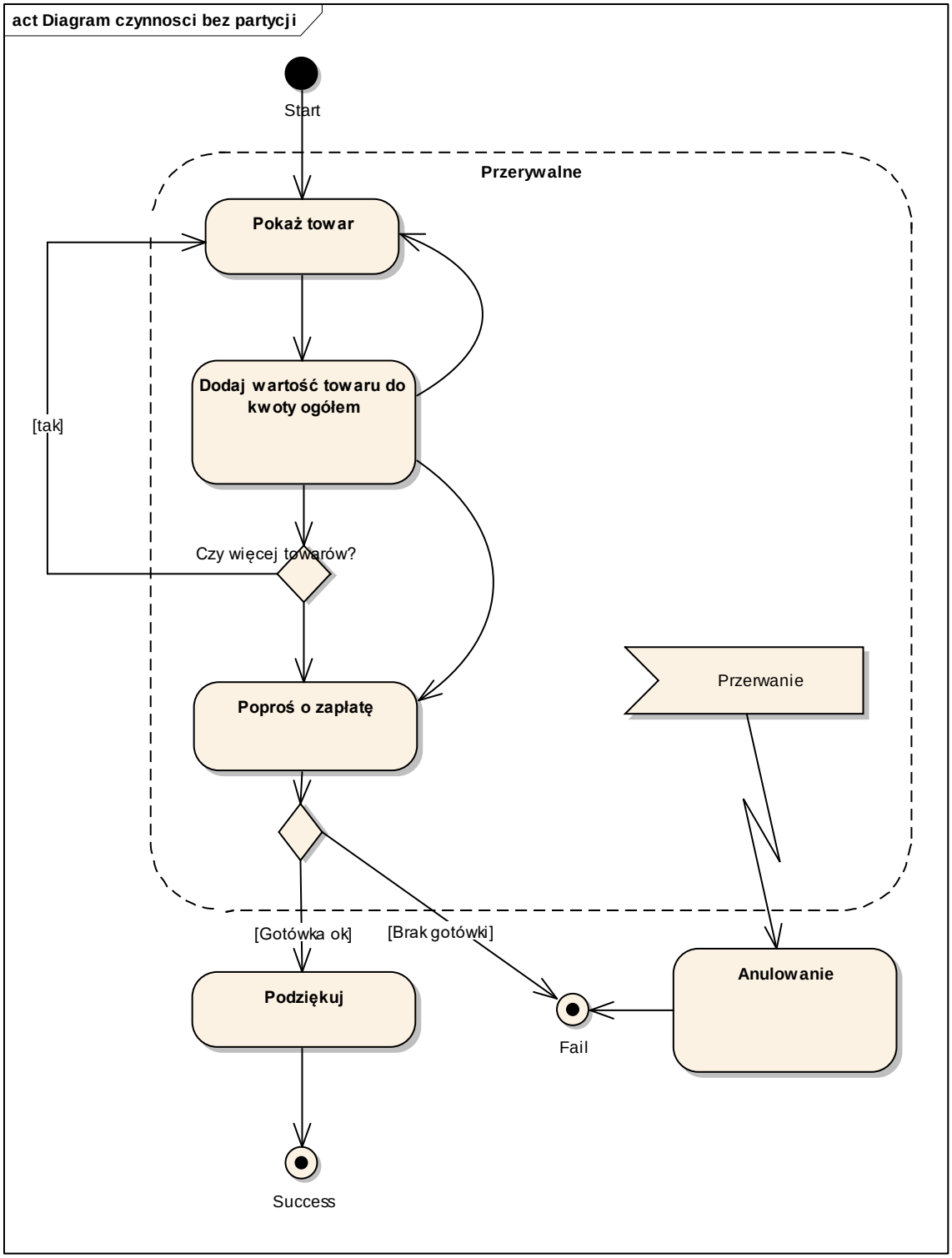
- Realizacja – implementacja interfejsu
  - Generalizacja, specjalizacja – dziedziczenie (tylko w zależności od kierunku)



10 Diagramy czynności

- Czynności vs akcje
    - Czynności – długotrwałe, podzielne, ogólne
    - Akcje – krótkotrwałe, niepodzielne, szczegółowe – nazwane czasownikowo (wprowadź/wybierz/zatwierdź/wydrukuj/aktualizuj/weryfikuj)
- Różnica w stosunku do diagramu stanów jeśli chodzi o semantykę bloków vs strzałek – tam bloczek = stan, strzałka = akcja; tu bloczek = akcja, strzałka – wyznacza następstwo akcji
    - Sygnały (zdarzenia) – wyślij, odbierz
    - Wariant – „if”
    - Zdarzenia – send/receive
    - Regiony – na przykład „przerywalny”, pojawia się zdarzenie „przerwij”, anulowanie
    - Partycje – podział na aktorów

Diagramy stanów i czynności wykorzystują niemalże ten sam formalizm do reprezentowania różnych kategorii diagramów.


act Diagram czynnosci bez partycji
Start
Przerywalne
Pokaż towar
Dodaj wartość towaru do kwoty ogółem
[tak]
Czy więcej towarów?
Poproś o zapłatę
Przerwanie
[Gotówka ok]
[Brak gotówki]
Podziękuj
Anulowanie
Fail
Success

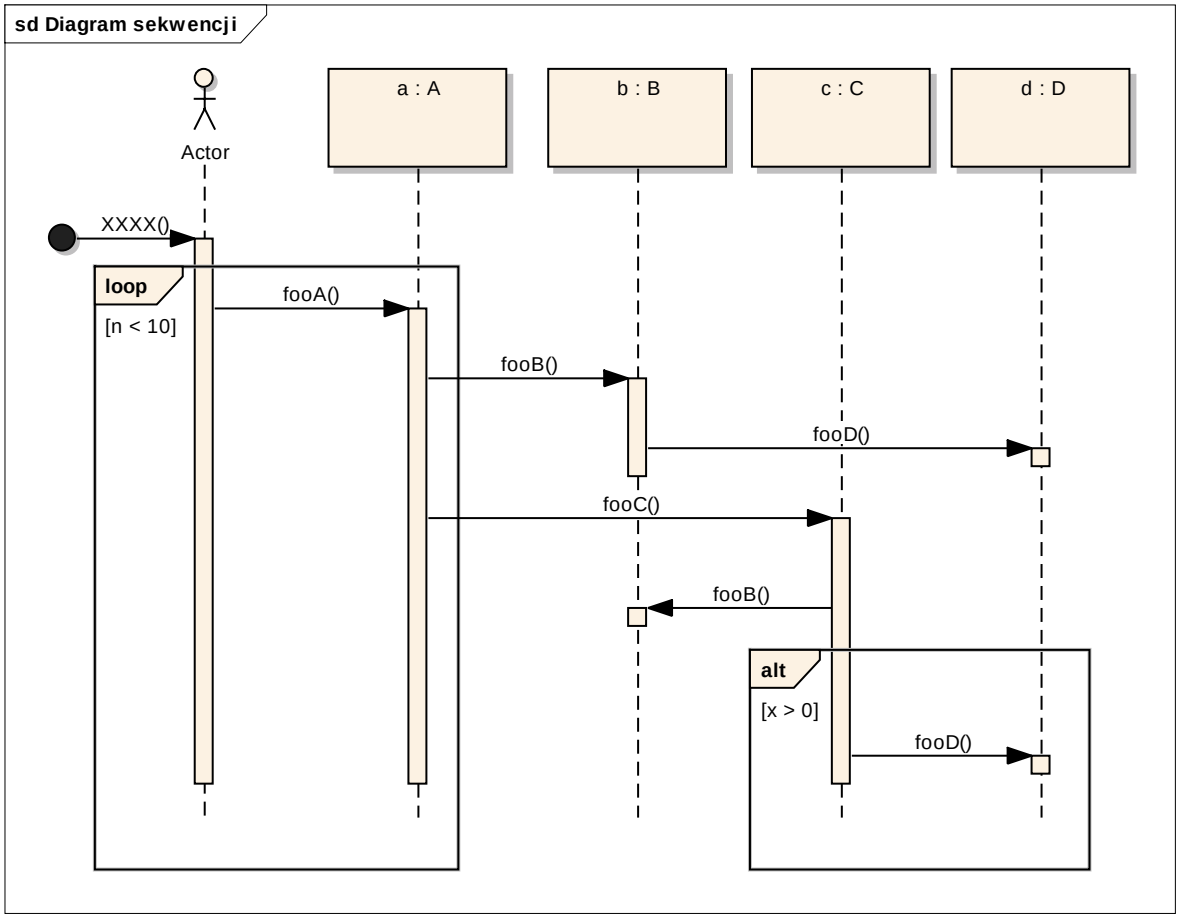
11 Diagramy sekwencji

- Linie życia, paski aktywacji/ośrodki sterowania (execution specification)
- Typy obiektów
  - Boundary – widok
  - Control – kontroler
  - Entity – model
- Związek między diagramem sekwencji a diagramem klas – ustalanie typu obiektu
- Komunikat – wartość zwrotna
  wartość = komunikat(p1:P1, p2:P2, …) : typ
- lub przerywana strzałka zwrotna (EA – niekoniecznie)
- Singleton – jedynka w rogu, metoda statyczna – stereotyp „class”, „metaclass”
- Komunikat odnaleziony – „od nikogo”
- Create/destroy
- Ramki, można zagnieżdżać
  - Loop – pętla
  - Alt – if-then-else
  - Opt – if
  - Neg – czynność nieprawidłowa, wyjątek
  - Par - współbieżność
  - Ref – odwołanie do innej, nazwanej ramki
  - Sd – nazwana ramka

Przykładowy pseudokod:

```
public class Actor {
        public void XXXX() {
                while ( n < 10 ) {
                  a.fooA();
                }
        }
}
public class A {
        public void fooA() {
                b.fooB();
                c.fooC();
        }
}
public class B {
        public void fooB() {
                d.fooD();
        }
}
public class C {
        public void fooC() {
                b.fooB();
                if ( x > 0 )
                   d.fooD();
        }
}
```

i jego diagram


sd Diagram sekwencji
Actor
a : A
b : B
c : C
d : D
XXXX()
loop
[n < 10]
fooA()
fooB()
fooD()
fooC()
fooB()
alt
[x > 0]
fooD()