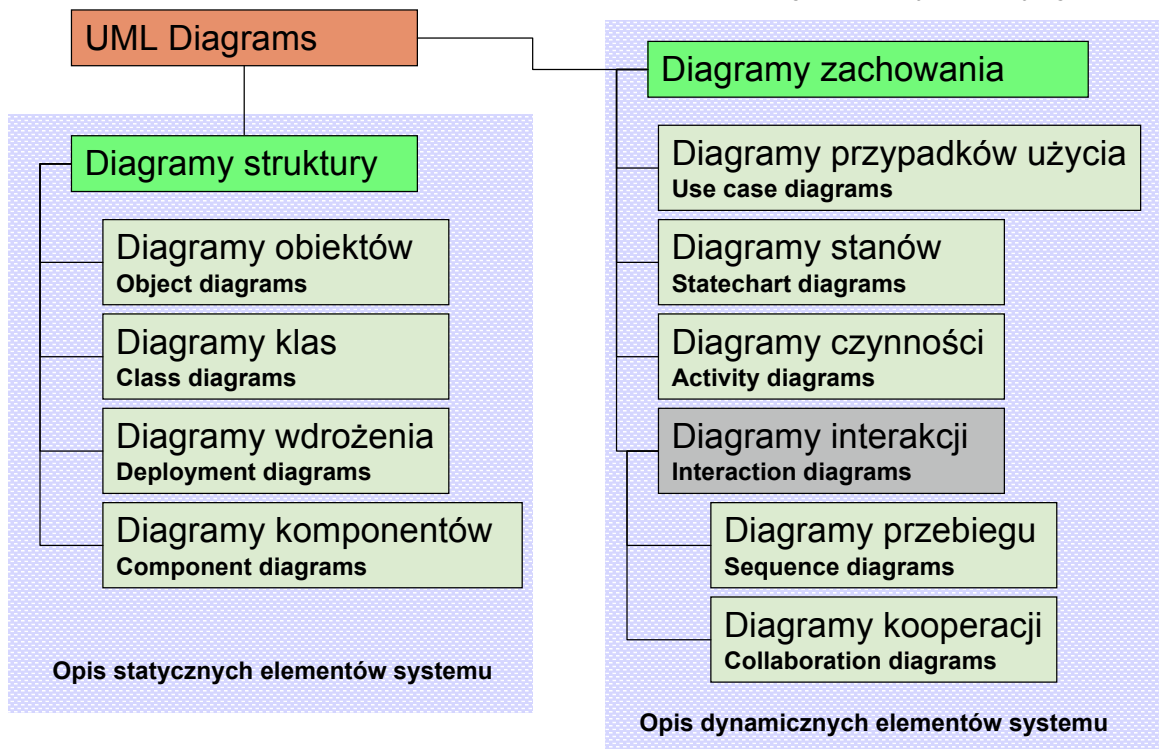


Klasyfikacja diagramów UML

Diagramy UML to schematy przedstawiające zbiór bytów i związków między nimi.



Diagramy klas (Class diagrams) (I)

Definicja:

Schemat przedstawiający zbiór klas, interfejsów, kooperacji oraz związków między nimi.

- Używa się ich do modelowania struktury systemu.
- Stanowią bazę wyjściową dla diagramów komponentów i diagramów wdrożenia.
- Szczególnie przydatne do tworzenia systemów (inżynieria do przodu i wstecz).

Zawartość:

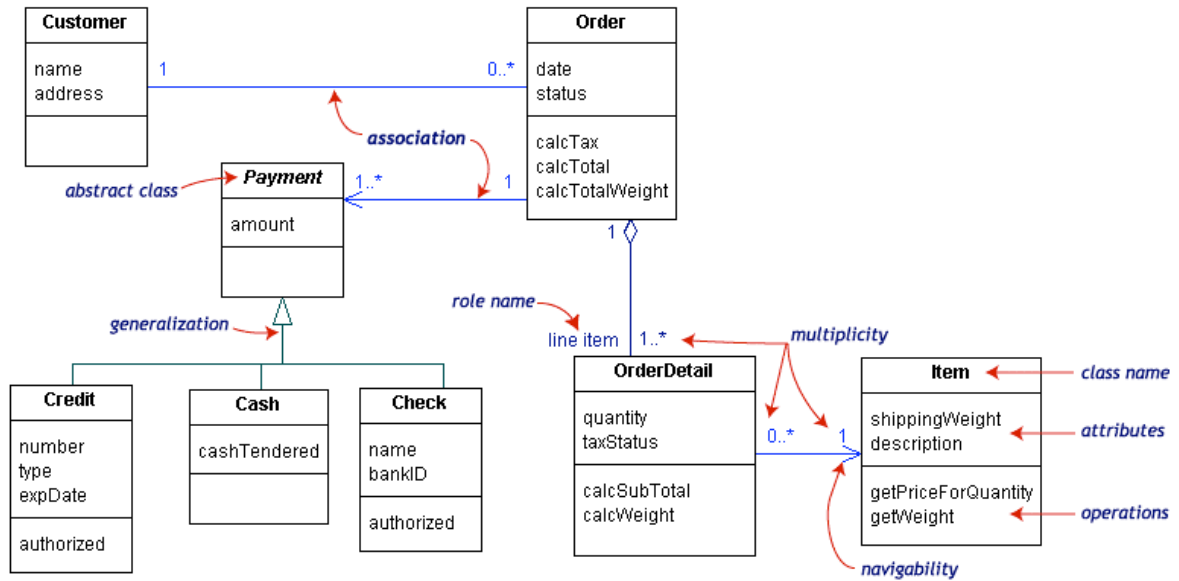
- klasy,
- interfejsy,
- kooperacje,
- zależności, uogólnienia, powiązania,
- notatki, ograniczenia, pakiety, podsystemy.

Zastosowania:

- modelowanie słownictwa systemu (struktura systemu),
- modelowanie prostych kooperacji,
- modelowanie schematu logicznej bazy danych.

Diagramy klas (Class diagrams) (II)

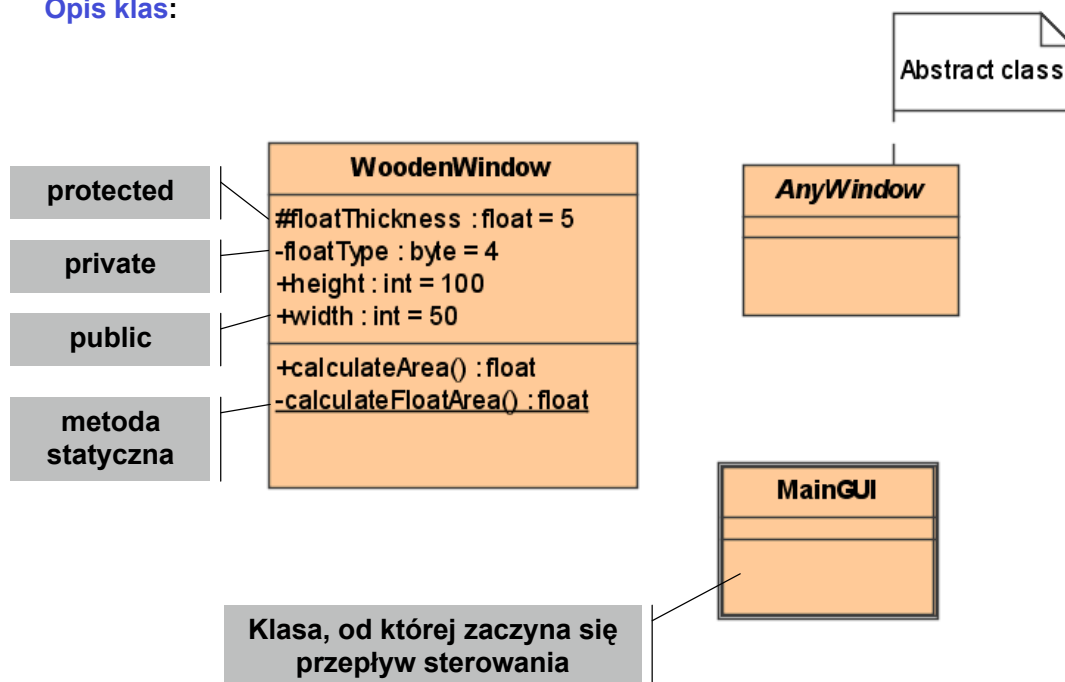
Przykładowy diagram klas



Elementy strukturalne: klasy
Związki: powiązanie, uogólnienie, agregacja

Diagramy klas (III)

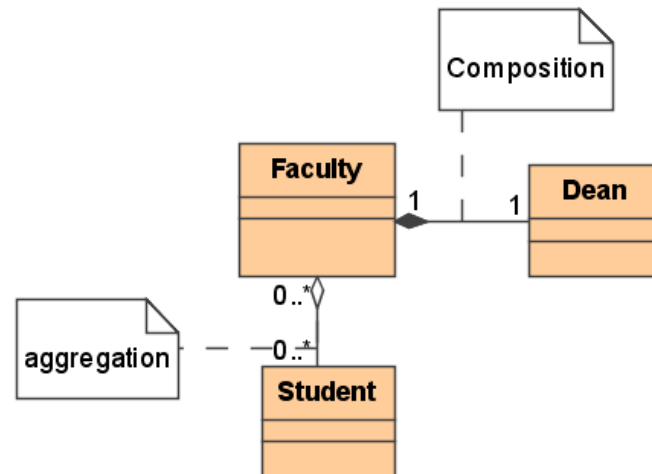
Opis klas:



Diagramy klas (IV)

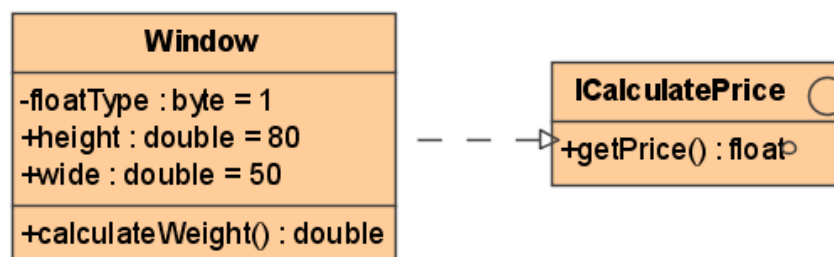
Powiązania:

- agregacja - element jest częścią całości,
- kompozycja (silna agregacja) - element nie może istnieć bez całości.



Diagramy klas (V)

Interfejsy:



Notacja lizakowa (ang. lollipop notation):



Diagramy klas - Zastosowania (I)

Modelowanie struktury systemu - klasy, interfejsy, związki, nazewnictwo, itp.

Modelowanie prostych kooperacji - realizowanie zadań niemożliwych do wykonania przez pojedynczą klasę.

Na diagramie klas powinna być modelowana tylko jedna kooperacja.

Etapy modelowania kooperacji:

1. identyfikacja mechanizmu (pewnej funkcjonalności systemu),
2. wyznaczenie klas, interfejsów, innych kooperacji,
3. określanie związków między ww.,
4. skonfrontowanie diagramu z przypadkami użycia,
5. ocena wartości poszczególnych bytów (może uda się część z nich wyeliminować).

Diagramy klas - Zastosowania (II)

Modelowanie schematu logicznej bazy danych

Diagramy klas są nadzbiorem diagramów ERD (encja-związek). Rozszerzają ERD (modelowanie danych) o modelowanie zachowania.

Etapy modelowania schematu logicznej bazy danych:

1. identyfikacja klas, dla których wartości pól powinny znaleźć się w bazie danych.
2. utworzenie diagramu tych klas, zaznaczenie ich jako trwałe (metka {persistent}),
3. wyspecyfikowanie atrybutów,
4. wyspecyfikowanie powiązań między atrybutami i ich liczebności,
5. odszukanie powiązań cyklicznych, wieloargumentowych, itp., wprowadzenie dodatkowych pośrednich abstrakcji w celu uproszczenia struktury logicznej,
6. zbadanie zagadnień fizycznego dostępu do danych, dostosowanie schematu logicznego do fizycznej bazy danych.

Diagramy obiektów (Object diagrams)

Definicja:

Diagramy obiektów obrazują zbiór obiektów i związków między nimi. Przedstawia się na nich egzemplarze elementów z diagramów klas istniejące w wybranym momencie działania systemu.

Zawartość:

- obiekty,
- wiązania,
- pakiety, notatki i ograniczenia.

Zastosowania:

- modelowanie struktur obiektowych.

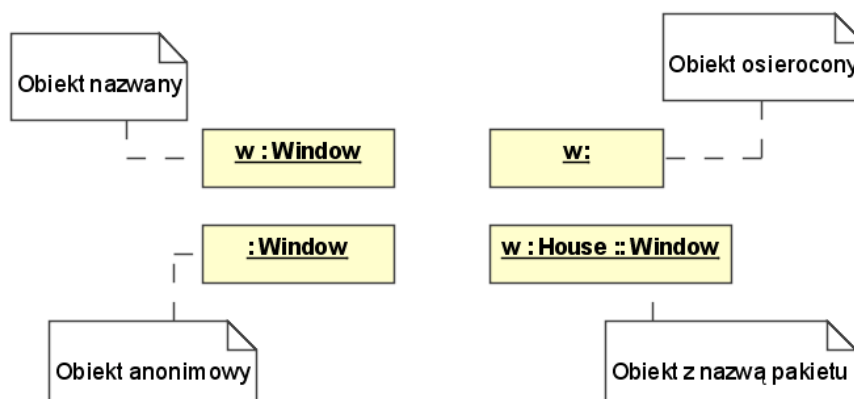
Diagramy obiektów - Egzemplarze

Egzemplarze służą do modelowania rzeczywistych lub prototypowych elementów ze świata rzeczywistego.

Obiekty to egzemplarze (ang. *instances*) klas (pewnych abstrakcji).

Cechy egzemplarzy:

- klasy, komponenty, węzły, przypadki użycia, powiązania mogą mieć egzemplarze,
- klasy abstrakcyjne, interfejsy nie mogą mieć egzemplarzy,
- obiekty nie muszą mieć przyporządkowanej abstrakcji.



Diagramy obiektów - Zastosowania

Modelowanie struktur obiektowych. Uwidocznienie klas o skomplikowanych powiązaniach.

diagram klas

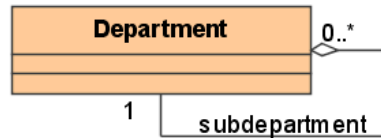
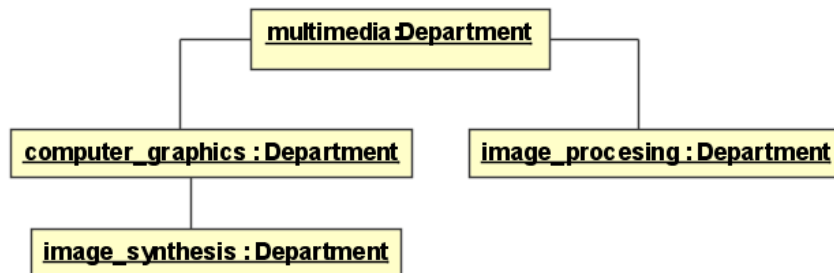


diagram obiektów



Diagramy komponentów (Componet diagrams)

Definicja:

Diagramy komponentów przedstawiają fizyczne aspekty systemów obiektowych. uwzględnia się w nich elementy fizyczne systemu (programy wykonywalne, biblioteki, tabele, dokumenty, itp.).

Mają wiele podobieństw z diagramami klas; główna różnica polega na przedstawianiu komponentów, a nie klas.

Zawartość:

- komponenty,
- interfejsy,
- związki (zależności, powiązania, uogólnienia, realizacje),
- pakiety, notatki i ograniczenia.

Zastosowania:

- modelowanie struktur obiektowych,
- modelowanie kodu źródłowego,
- modelowanie wersji programów wykonywalnych,
- modelowanie fizycznej bazy danych,
- modelowanie elastycznych systemów.

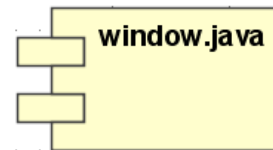
Diagramy komponentów - Komponenty (I)

Komponent - fizyczna, wymienna część systemu, która wykorzystuje i realizuje pewien zbiór interfejsów.

- Służą do modelowania elementów fizycznych: programów wykonywalnych, bibliotek, tabel, plików, dokumentów, itp.
- Komponent to fizyczne opakowanie bytów logicznych (klas, interfejsów, kooperacji).

Cechy komponentów:

- elementy fizyczne,
- elementy wymienne, mogą być zastąpione przez inne komponenty,
- elementy będące częścią systemu,
- wykorzystują (zależność) lub realizują (realizacja) zbiór interfejsów.

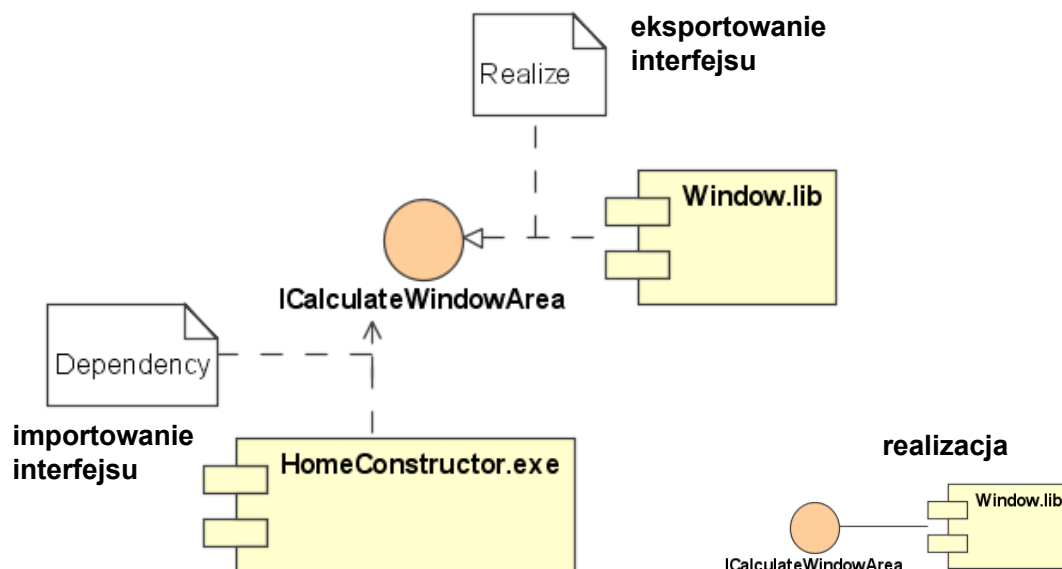


Zastosowania komponentów:

- modelowanie programów wykonywalnych i bibliotek,
- modelowanie tabel, plików i dokumentów,
- modelowanie API,
- modelowanie kodu źródłowego.

Diagramy komponentów - Komponenty (II)

Komponenty realizują bądź wykorzystują interfejsy. Interfejs to zestaw operacji, które wyznaczają usługi oferowane przez komponent (klasę).



Diagramy komponentów - Zastosowania (I)

Modelowanie struktur obiektowych

Modelowanie systemów składających się z wielu bibliotek, programów wykonywalnych, itp. Komponenty łączą się ze sobą za pośrednictwem interfejsów.

Modelowanie tabel, plików i dokumentów

Modelowanie komponentów pomocniczych: dokumentów pomocy, skryptów, plików z danymi, itp. Obrazowanie związków między komponentami pomocniczymi i głównymi.

Modelowanie API

Modelowanie interfejsu programowego (API). Szczegółowa specyfikacja operacji API jest ukryta w interfejsach.

Diagramy komponentów - Zastosowania (II)

Modelowanie kodu źródłowego

Opis plików źródłowych oraz plików pomocniczych wykorzystywanych w czasie pisania oprogramowania. Wersje plików rejestrowane są w **metkach**. Uwidaczniane są **zależności** między plikami (np. kolejność kompilacji).

Modelowanie wersji programów wykonywalnych

Opis struktury oprogramowania: wersji programów wykonywalnych, wymaganych przez nie bibliotek, plików konfiguracyjnych, itp.

- Wersje komponentów w metkach.
- Uwidocznienie zależności pomiędzy komponentami poprzez obrazowanie interfejsów.

Modelowanie fizycznej bazy danych

Obrazowanie fizycznej bazy danych na podstawie jej logicznego modelu (diagram klas).

- W przypadku relacyjnej fizycznej bazy danych - problemy z dziedziczeniem i fizyczną reprezentacją operacji.

Modelowanie elastycznych systemów

Obrazowanie struktury systemu, w którym komponenty przemieszczają się pomiędzy węzłami (systemy rozproszone).

Diagramy wdrożenia (Deployment Diagrams)

Definicja:

Diagramy wdrożenia służą do przedstawiania fizycznych aspektów systemów obiektowych. Obrazują konfigurację węzłów i zainstalowane na nich komponenty.

- Definiują styk sprzętu z oprogramowaniem.
- Określają z jakich fizycznych elementów powinien być zbudowany system.

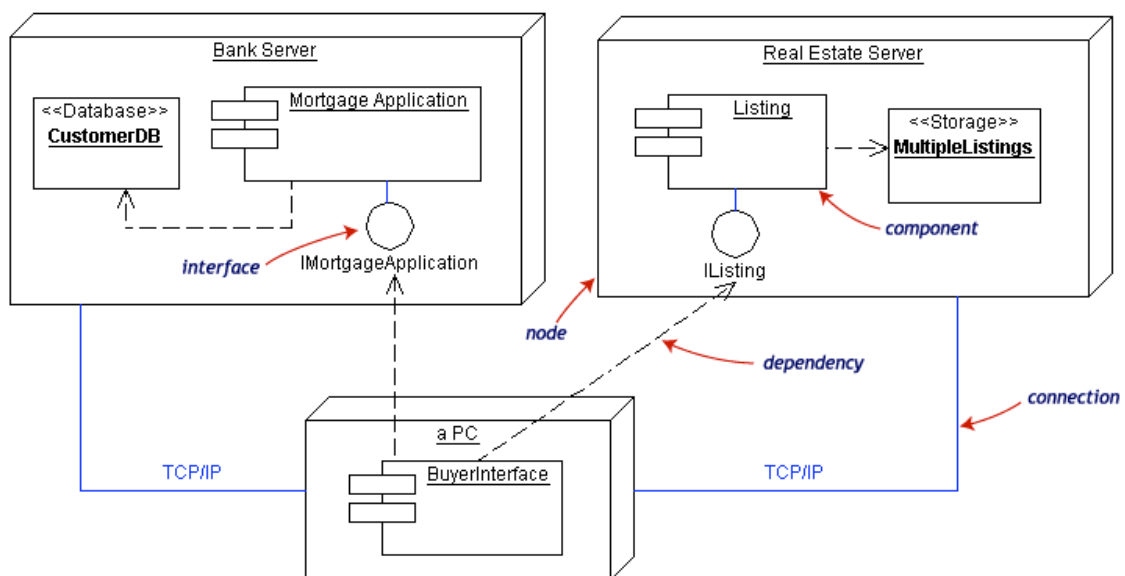
Zawartość:

- węzły,
- zależności i powiązania,
- komponenty przypisane do węzłów,
- notatki i ograniczenia.

Zastosowania:

- modelowanie perspektywy wdrożeniowej systemu:
 - modelowanie systemów wbudowanych,
 - modelowanie systemów typu klient-serwer,
 - modelowanie systemów rozproszonych.

Diagramy wdrożenia - Przykład



Diagramy wdrożenia - Zastosowania

Modelowanie systemów wbudowanych

Elementy fizyczne w systemach wbudowanych charakteryzują się nieliniowością reakcji i odpowiedzi. Diagramy wdrożenia obrazują styk elementów fizycznych z oprogramowaniem.

- Ułatwiają porozumienie między specjalistami i programistami.
- Do obrazowania urządzeń stosuje się stereotypy (specyficzne symbole).

Modelowanie systemu typu klient-serwer

Obrazowanie komponentów serwera (baza danych, programy obliczeniowe) i klienta (interfejs użytkownika).

- Klienci o dużej i małej mocy obliczeniowej.
- Fizyczne położenie klientów i serwera.
- Struktura połączeń sieciowych.

Modelowanie systemu rozproszonego

Obrazowanie struktury złożonych systemów zmieniających się w sposób dynamiczny.

- Dynamiczne przydzielanie komponentów do węzłów.
- Dynamiczna zmiana liczby węzłów.
- Macierze baz danych.
- Logiczne grupy węzłów (Internet może być węzłem).

Diagramy przypadków użycia (Use Case Diagrams)

Definicja:

Diagramy służące do modelowania zachowania systemu. Opisują co system powinien robić z punktu widzenia obserwatora z zewnątrz. Przedstawiają scenariusze realizacji określonych zachowań (funkcji systemu).

Zawartość:

- **przypadki użycia** (ang. *use case*) - opisy zdarzeń,
- **aktorzy** - osoby/rzeczy inicjujące zdarzenia,
- powiązania między aktorami i przypadkami użycia,
- zależności, uogólnienia i powiązania między przypadkami użycia,
- pakiety, notatki i ograniczenia.

Zastosowania:

- **modelowanie zachowania bytów** - opis ciągu akcji zmierzających do realizacji danej funkcji systemu,
- **modelowanie otoczenia systemu** - definiowanie aktorów i ich ról,
- **modelowanie wymagań stawianych systemowi** - określenie co system powinien robić,
- **testowanie systemu.**



Diagramy przypadków użycia - Przypadki użycia (I)

Przykłady przypadków użycia - wypłata pieniędzy z bankomatu.

Aktorzy: Klient

ID	Aktor	Nazwa	Opis	Odpowiedź systemu
B1	Klient	Zrealizuj wypłatę z bankomatu	1. Klient aktywuje bankomat poprzez włożenie karty.	1. Bankomat pyta się o PIN.
			2. Klient wprowadza PIN.	2. Bankomat weryfikuje użytkownika i pyta się o kwotę.
			3. Klient podaje kwotę.	3. Bankomat żąda potwierdzenia.
			4. Klient potwierdza kwotę.	4. Bankomat pyta się czy wydrukować potwierdzenie.
			5.1 Klient żąda wydrukowania potwierdzenia.	5.1 Bankomat żąda zabrania karty.
			5.1.1 Klient zabiera kartę.	5.1.1 Bankomat wypłaca pieniądze i drukuje potwierdzenie.
			5.2 Klient rezygnuje z wydrukowania potwierdzenia.	5.2 Bankomat żąda zabrania karty.
				5.2.1 Bankomat wypłaca pieniądze.
			6. Klient odbiera pieniądze.	6. Bankomat zatwierdza transakcję.

Diagramy przypadków użycia - Przypadki użycia (II)

Przypadki użycia - opis akcji służących do definiowania funkcjonalności budowanego systemu bez konieczności określania sposobu jego implementacji.

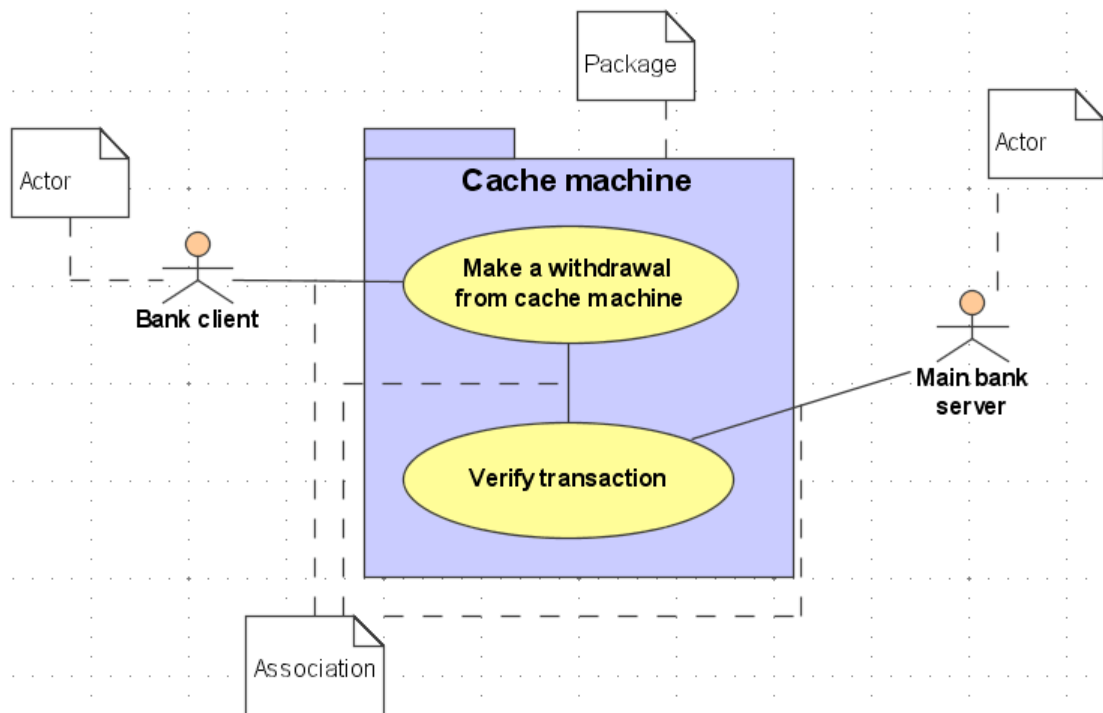
Funkcje:

- zdefiniowanie zachowania (funkcjonalności) systemu,
- wypracowanie porozumienia między **programistami**, **użytkownikami** i **ekspertami**,
- weryfikacja architektury systemu w czasie jego budowania (**testowanie regresywne**),
- zdefiniowanie **kooperacji** realizujących określone funkcje (kooperacją może być pojedynczy przypadek użycia).

Przypadki użycia:

- **zbiór ciągów akcji** opisujących interakcję elementów z otoczenia systemu (aktorów) z samym systemem,
- **aktor** - człowiek lub zautomatyzowany system,
- przypadki użycia mogą mieć ogólne lub bardziej szczegółowe warianty, dany przypadek użycia może być podzbiorem większego przypadku użycia,
- **opisują co system robi, ale NIE jak to robi**,
- mogą pojawiać się alternatywne ciągi akcji.

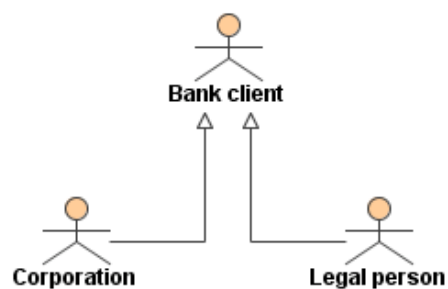
Diagramy przypadków użycia - Przykład



Diagramy przypadków użycia - Związki (I)

Związki między aktorami:

- **uogólnienia.**



Związki między aktorami i przypadkami użycia:

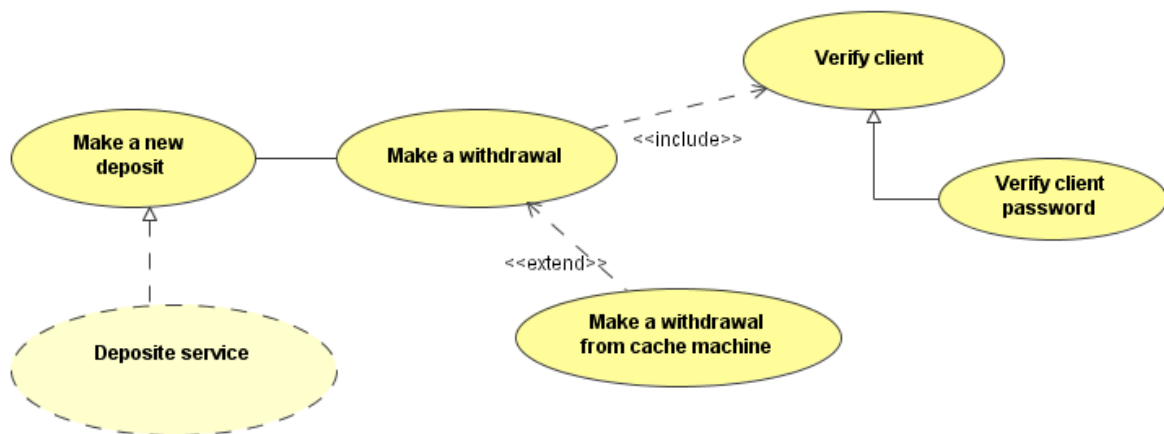
- **powiązania.**



Diagramy przypadków użycia - Związki (I)

Związki między przypadkami użycia:

- powiązanie,
- uogólnienie,
- realizacja,
- zależność (stereotypy: <<include>>, <<extend>>).



Diagramy przypadków użycia - Zastosowania (I)

Modelowanie zachowania bytu

Modelowanie danej funkcjonalności systemu.

Etapy modelowania zachowania:

1. identyfikacja aktorów, uporządkowanie aktorów (role bardziej ogólne i bardziej szczegółowe)
2. określenie przypadków użycia i interakcji między przypadkami użycia,
3. określenie interakcji między aktorami i przypadkami użycia,
4. usystematyzowanie przypadków użycia (związki zawierania i rozszerzania).

Modelowanie otoczenia systemu

Zdefiniowanie aktorów i ich ról w systemie. Określenie co/kto ma **NIE być** aktorem.

Etapy modelowania:

1. identyfikacja aktorów, które grupy potrzebują pomocy systemu w realizacji określonych zadań,
2. hierarchizacja aktorów za pomocą uogólnień,
3. dodanie stereotypów do aktorów,
4. zdefiniowanie powiązań między aktorami i przypadkami użycia.

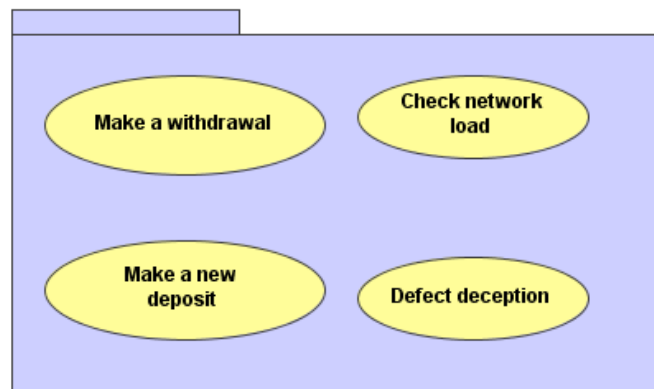
Diagramy przypadków użycia - Zastosowania (II)

Modelowanie wymagań stawianych systemowi

Zdefiniowanie co system ma robić (**określenie wszystkich funkcjonalności systemu**). Modelowanie obejmuje również wymagania niefunkcjonalne, tzn. realizowane przez sam system lub znaczenie NIE realizowane przez aktorów, którzy będą korzystali z systemu.

Etapy modelowania:

1. zdefiniowanie aktorów,
2. zdefiniowanie i uporządkowanie przypadków użycia realizowanych przez aktorów (funkcjonalności),
3. zdefiniowanie przypadków użycia realizowanych przez system (czynności niefunkcjonalnych).



Diagramy interakcji

Diagramy przebiegu (Sequence Diagrams)

Diagramy kooperacji (Collaboration Diagrams)

Definicja:

Diagramy interakcji (ang. Interaction Diagrams) służą do modelowania zachowania systemu. Ilustrują kiedy i w jaki sposób komunikaty przesyłane są pomiędzy obiektami.

Na diagramie przebiegu uwypukla się kolejność wysyłania komunikatów w czasie.

Na diagramie kooperacji kładzie się nacisk na związki strukturalne między obiektami wysyłającymi i odbierającymi komunikaty.

Zawartość:

- obiekty,
- wiązania,
- komunikaty,
- notatki i ograniczenia.

Zastosowania:

- modelowanie przepływu sterowania z uwzględnieniem kolejności komunikatów w czasie,
- modelowanie przepływu sterowania z uwzględnieniem organizacji strukturalnej obiektów.

Diagramy interakcji - Interakcje

Definicja:

Interakcja to zbiór komunikatów wysyłanych w obrębie pewnego zbioru obiektów, w ustalonym otoczeniu i w ściśle określonym celu. Interakcja obrazuje przepływ sterowania w systemie.

Interakcje opisują komunikaty wysyłane pomiędzy obiektami.

Wiązania - powiązania między obiektami (egzemplarze powiązań). Musi istnieć wiązanie między obiektami, aby możliwe było wysyłanie komunikatów pomiędzy nimi.

Stereotypowe wiązania:

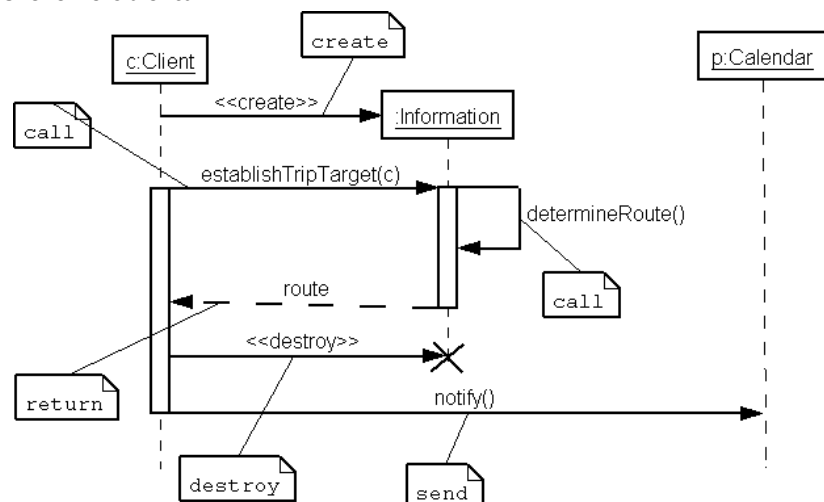
- <<association>> - widoczność obiektów przez powiązanie,
- <<self>> - obiekt jest widoczny, bo sam odebrał komunikat,
- <<global>> - obiekt w globalnym zasięgu,
- <<local>> - obiekt w lokalnym zasięgu,
- <<parameter>> - obiekt jest widoczny, ponieważ jest parametrem.

Diagramy interakcji - Komunikaty

Komunikaty - zlecenia wykonania określonej czynności (aktywacja obiektu, tworzenie lub niszczenie obiektu). Akcja spowodowana przekazaniem komunikatu jest instrukcją wykonywalną.

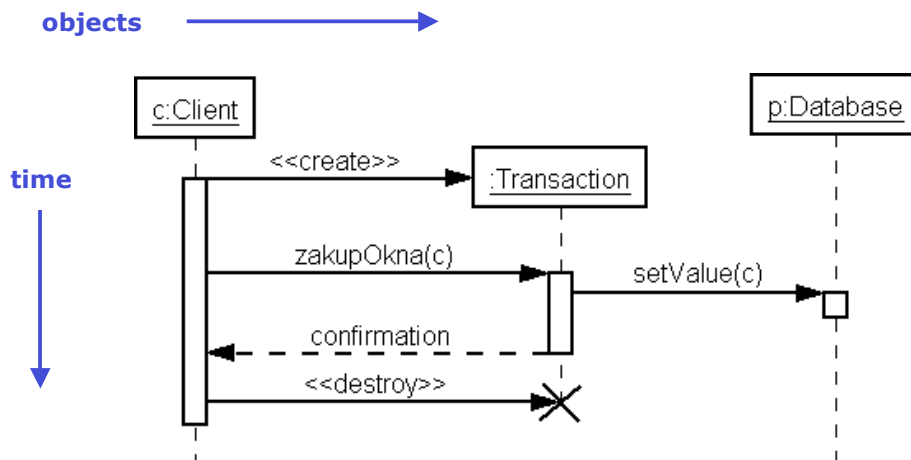
Rodzaje akcji:

- <<call>> - wywołanie operacji,
- <<return>> - przekazanie wartości wywołującemu,
- <<send>> - wysłanie sygnału do obiektu,
- <<create>> - utworzenie nowego obiektu,
- <<destroy>> - zniszczenie obiektu.



Diagramy interakcji - Diagramy przebiegu

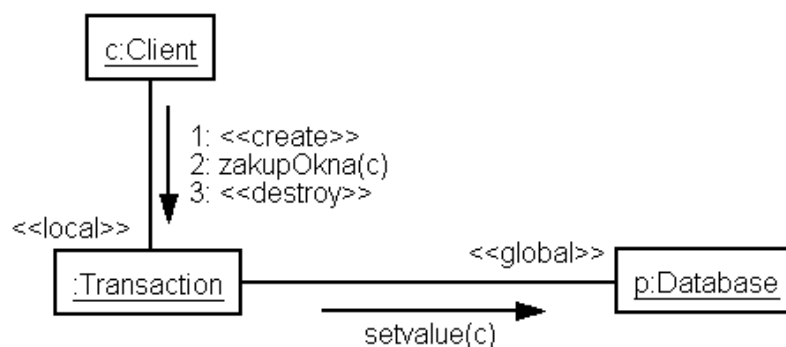
Uwypuklają kolejność komunikatów w czasie.



- linie życia
- ośrodki sterowania
- upływ czasu
- stereotypy <<create>>, <<destroy>>

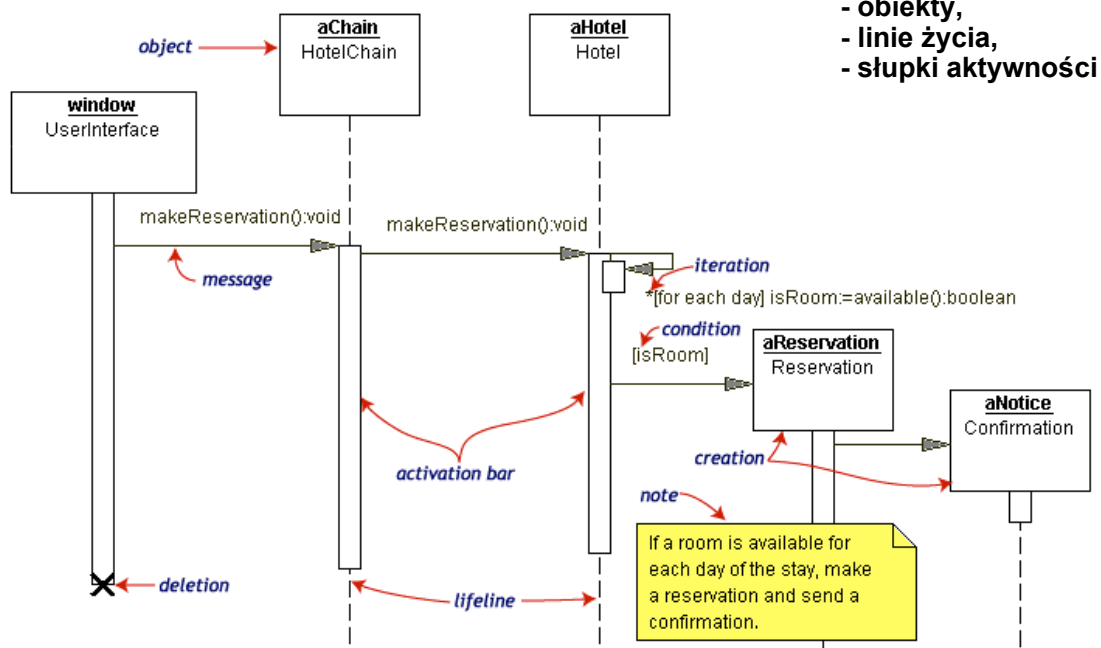
Diagramy interakcji - Diagramy kooperacji

Uwypuklają organizację obiektów uczestniczących w interakcji.



- ścieżki (stereotypy <<local>>, <<global>>, <<self>>, <<parameter>>),
- ciąg komunikatów
- wyrażenia warunkowe → rozgałęzienia

Diagramy przebiegu - Przykład



Rezerwacja pokoju w hotelu.

Diagramy interakcji - Zastosowania

Modelowanie przepływu sterowania z uwzględnieniem kolejności komunikatów w czasie.

Modelowanie przepływu sterowania z uwzględnieniem organizacji strukturalnej obiektów.

Diagramy czynności (Activity Diagrams)

Definicja:

Diagramy czynności (schematy blokowe) przedstawiają przepływ sterowania od czynności do czynności. Większość diagramów czynności przedstawia kroki procesu obliczeniowego.

Zawartość:

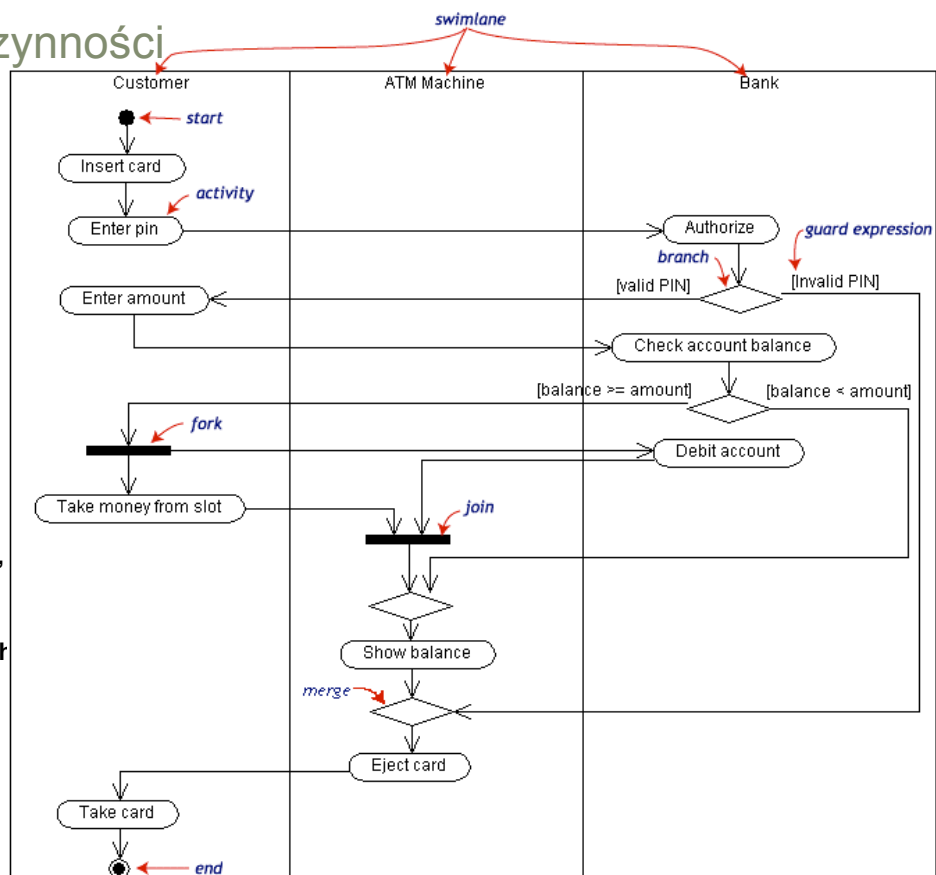
- stany akcji i stany czynności,
- przejścia,
- obiekty,
- notatki i ograniczenia.

Zastosowania:

- modelowanie przepływu czynności,
- modelowanie operacji.

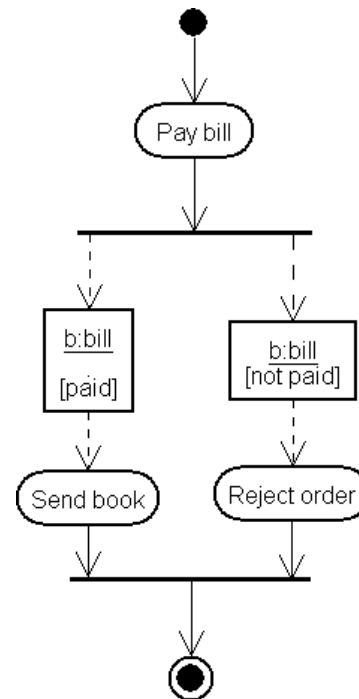
Diagramy czynności - Przykład

- Stany akcji (activity),
- stany czynności,
- przejścia,
- rozgałęzienia (branch)
- rozwidlanie (fork)
- scalanie (join),
- tory (swimlane),
- przepływ obiektów.



Diagramy czynności - Przykład

- **stany akcji** - wykonywalne operacje/obliczenia, niepodzielne - żadne zdarzenie nie może ich przerwać, wykonywane w znikomym czasie,
- **stany czynności** - mogą być dekomponowane na stany akcji i inne stany czynności, mogą trwać długo,
- **przejścia** - przekazanie sterowania z jednego stanu do drugiego,
- **rozgałęzienia** - przejścia alternatywne,
- **rozwidlanie i scalanie ścieżek** - za pomocą pasków synchronizacyjnych,
- **tory** - podział stanów czynności na grupy,
- **przepływ obiektów**.



Diagramy czynności - Zastosowania

Modelowanie przepływu czynności:

- obrazowanie przepływu czynności pomiędzy systemami zautomatyzowanymi i niezautomatyzowanymi.

Modelowanie operacji:

- diagram czynności to schemat blokowy akcji operacji,
- stosowane do obrazowania b. skomplikowanych operacji lub do dokumentowania specyficznych operacji.

Diagramy stanu (Statechart Diagrams)

Definicja:

Diagramy stanu przedstawiają maszynę stanową. Obrazują przepływ czynności między stanami. Służą do modelowania obiektów reaktywnych, tzn. takich, które najlepiej są określane przez odpowiedzi na zdarzenia wywołane w ich otoczeniu. Bieżące działanie obiektu reaktywnego zależy od tego co było w przeszłości.

Zawartość:

- stany zwykłe i złożone,
- przejścia ze zdarzeniami i akcjami,
- obiekty,
- notatki i ograniczenia.

Zastosowania:

- **modelowanie obiektów reaktywnych.**

Diagramy stanów - Zdarzenia

Zdarzenie - specyfikacja zjawiska zachodzącego w czasie i w przestrzeni, np. bodziec uruchamiający przejście między stanami. Zdarzenia mogą być synchroniczne bądź asynchroniczne, wewnętrzne (np. przepełnienie arytmetyczne) lub zewnętrzne (zachodzące między systemem i aktorami).

Rodzaje zdarzeń:

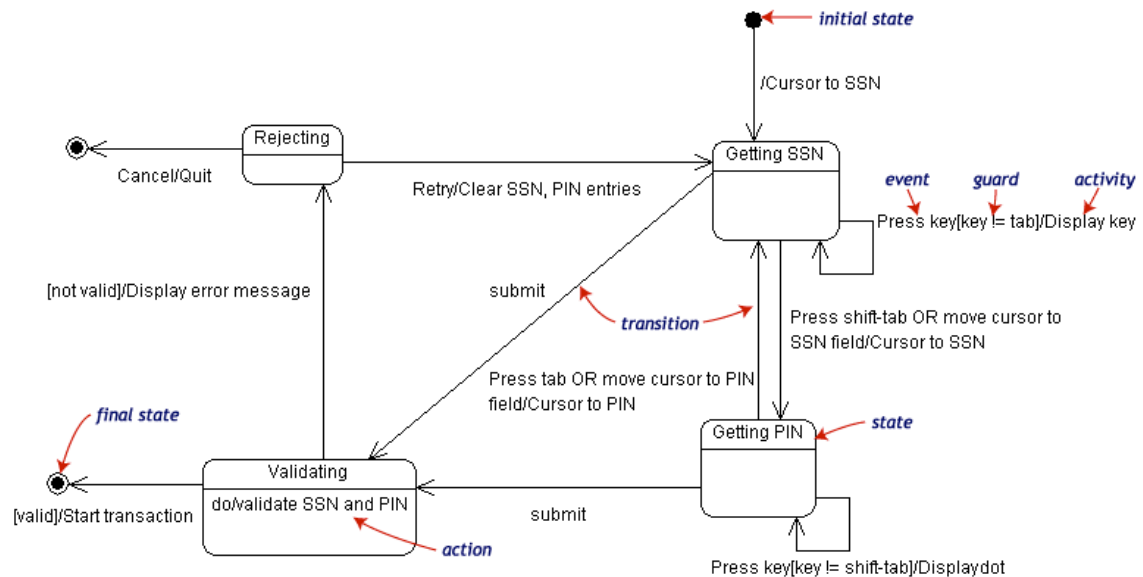
- sygnał,
- wywołanie,
- upływ czasu
- zmiana stanu.

Sygnał - zdarzenie asynchroniczne, obiekt wysyłany z jednego obiektu do drugiego.

Przykładem sygnału jest wyjątek. Może być modelowany jako np. komunikat w interakcji bądź przejście w maszynie stanowej.

Do obrazowania, że operacja wysłała sygnał stosuje się stereotyp <<send>.

Diagramy stanów - Przykład



- **Przejścia (transition),**
- **stany (state),**
- **akcje (action).**

Diagramy stanów - Zastosowania

Modelowanie obiektów reaktywnych - przedstawienie zachowania obiektu w czasie jego życia.

Uwypuklenie elementów:

- stanów stabilnych obiektu (stałe warunki przez zauważalny czas),
- zdarzeń uruchamiających przejścia,
- akcji wykonywanych przy zmianie stanu.

Literatura

1. Booch Grady, Rumbaugh James, Jacobson Ivar, „UML przewodnik użytkownika”, WNT Warszawa 2001, ISBN 83-204- 2618-9 (in Polish)
2. Booch Grady, Rumbaugh James, Jacobson Ivar, „The Unified Modeling Language User Guide”, Addison Wesley Longman Inc., 1999
3. <http://bdn.borland.com/article/0,1410,31863,00.html>



Diagram przypadków użycia, to graficzne przedstawienie przypadków użycia, aktorów oraz związków między nimi, występujących w danej dziedzinie przedmiotowej

Definicja

Diagram przypadków użycia, choć jest zbudowany z kilku elementów, odgrywa najważniejszą rolę w procesie modelowania biznesowego.

Diagram przypadków użycia zawiera następujące główne kategorie kojęciowe:

- Przypadek użycia
- Aktor
- Związek

Przypadek użycia



Przypadek użycia (ang. use case) to zbiór scenariuszy powiązanych ze sobą wspólnym celem użytkownika.

1. Przypadki użycia są stosowane w całej analizie modelowania biznesowego
2. Przypadek użycia musi być w interakcji chociaż z jednym aktorem. Wyjątek stanowi sytuacja, gdy przypadek użycia jest połączony z innym przypadkiem użycia

Każdy przypadek użycia można opisać za pomocą takich cech jak:

- Nazwa;
- Opis;
- Przepływ zdarzeń (scenariusz);
- Zależności i relacje;

Jedną z najważniejszych cech, jaką opisuje przypadek użycia, jest przepływ zdarzeń - scenariusze, które wskazują zestaw, sekwencji kolejno wykonywanych czynności.

Nazwę przypadku użycia stanowi zwięzłe polecenie wykonania funkcji w modelowanym systemie biznesowym, sformułowane w trybie rozkazującym. Na Rysunku 1. pokazano oznaczenie przypadku użycia zgodne ze standardem UML 2.1.



Aktor

Z każdym modelowanym systemem komunikują się związani z nim aktorzy.



Aktor (ang. Actor) to spójny zbiór ról odgrywanych przez użytkowników przypadku użycia w czasie interakcji z tym przypadkiem użycia.

Aktorzy mogą być osobowi lub nieosobowi:

- Rolę aktora osobowego może pełnić osoba, zespół, dział, instytucja, organizacja, zrzeszenie. Nazwy aktorów osobowych często pokrywają się z nazwami stanowisk pełnionych w organizacji, projekcie czy przedsięwzięciu.
- Aktorami nieosobowymi są systemy zewnętrzne, urządzenia oraz czas.

Na Rysunku 2 pokazano przykładowych aktorów zgodnie ze standardem UML 2.1.



Nazwę aktora wyraża się rzeczownikiem lub określeniem rzeczownikowym w liczbie pojedynczej.

Aktor może użytkować jeden lub więcej przypadków użycia w danym procesie biznesowym, natomiast przypadek użycia może być użytkowany przez jednego bądź wielu aktorów.

Podsumowując:

1. Aktor nie jest częścią systemu
2. Reprezentuje rolę w jaką może wcielić się użytkownik
3. Może reprezentować człowieka, urządzenie bądź inny system

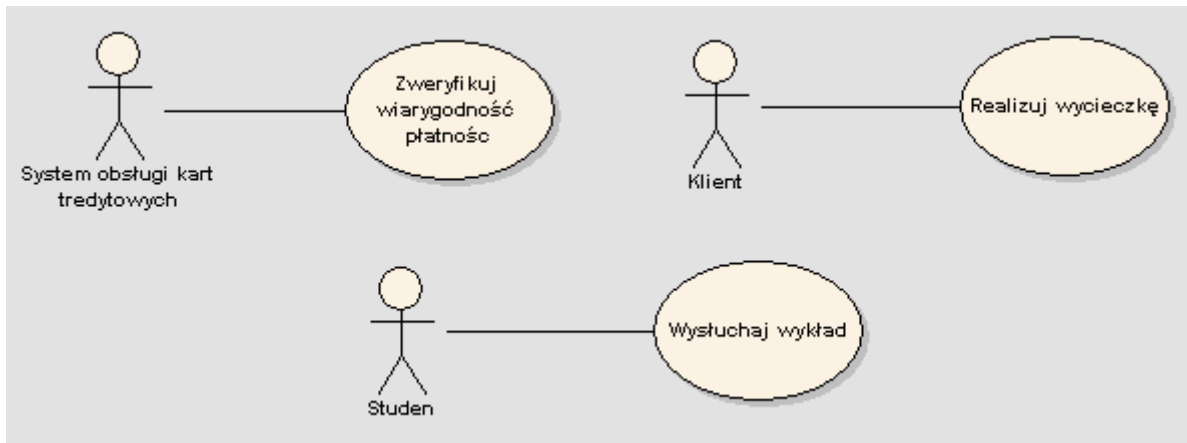
Związek

1. Każdy aktor umieszczony w diagramie przypadków użycia powinien być bezpośrednio powiązany z co najmniej jednym przypadkiem użycia
2. Każdy przypadek użycia powinien być użytkowany przez co najmniej jednego aktora.

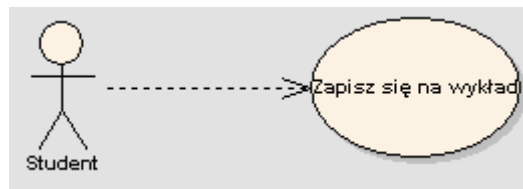


Związek (ang. relationship) stanowi semantyczne powiązanie pomiędzy elementami modelu.

Do łączenia diagramów z aktorami najczęściej stosuje się powiązania poprzez asocjację. Na rysunku 3 pokazano tego typu powiązania.



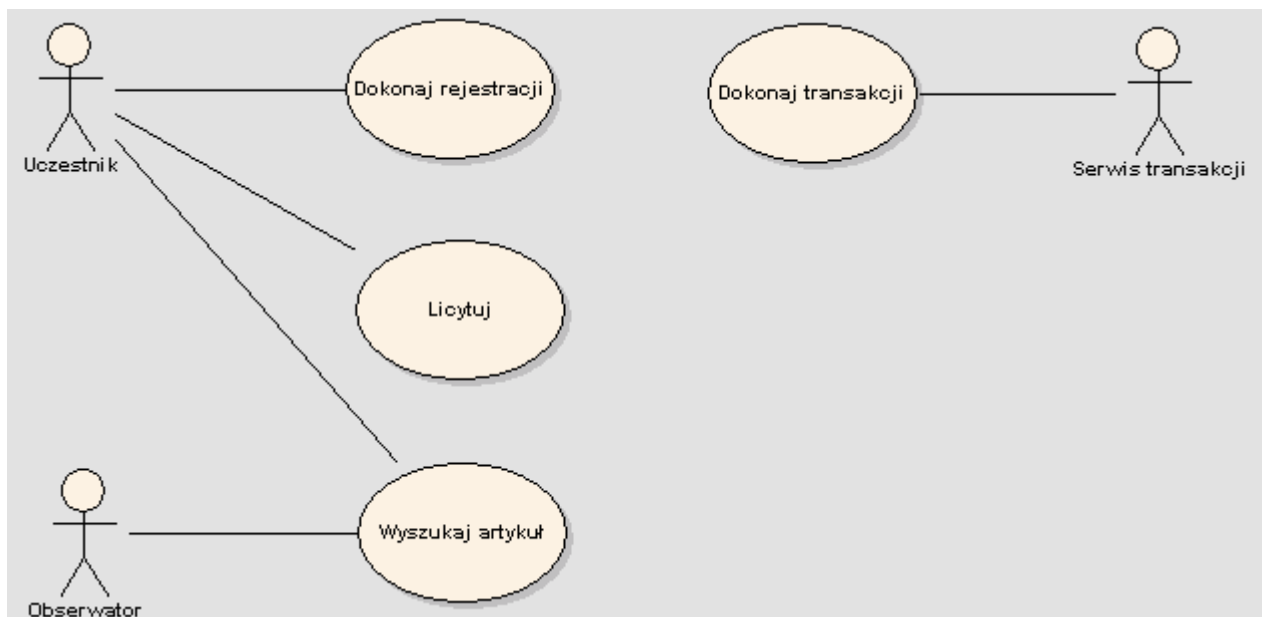
Bardzo często jest to asocjacja skierowana, która wskazuje, kto inicjuje usługę. Taki typ Asocjacji pokazano na Rysunku 4.



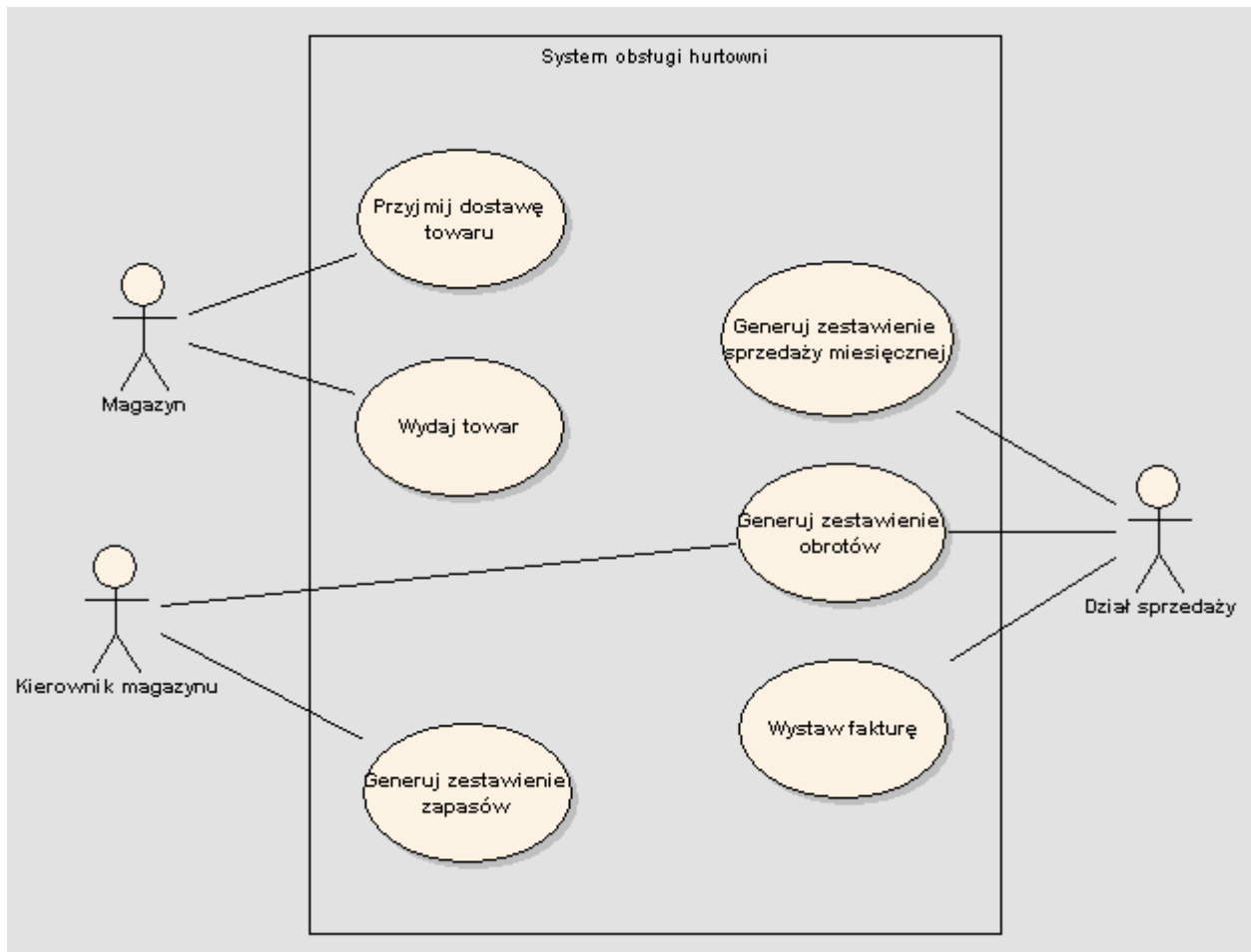
- Student - inicjator usługi
- Element inicjujący (Student) zna element inicjowany (Wykład)
- Element inicjowany (Wykład) nie na elemencie inicjującego (Studenta)

Zatem: Student wie o możliwości zapisania się na wykład natomiast rejestracja na wykład nie wie o istnieniu Studenta.

Przykład 1 - Aukcja internetowa



Przykład 2 - System obsługi hurtowni



Strukturalne związki zawierania i rozszerzenia

Istnieją dwa strukturalne związki:

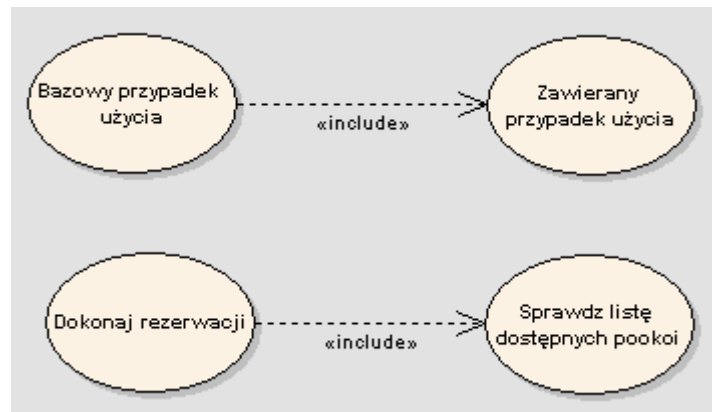
1. Zawieranie
2. Rozszerzenie

Zawieranie



Zawieranie (<<include>>) powiązanie pomiędzy przypadkiem zawierającym tj. bazowym przypadkiem użycia, a przypadkiem zawieranym

- Związek zawierania umożliwia uniknięcie sytuacji, w której ta sama funkcjonalność będzie opisywana wielokrotnie
- Zawierany przypadek użycia stanowi tzw. blok wielokrotnego użycia, który wskazuje tę część rozwiązania, do której można będzie odwołać się wielokrotnie.
- Związek zawierania skierowany jest góram w stronę zawieranego przypadku użycia



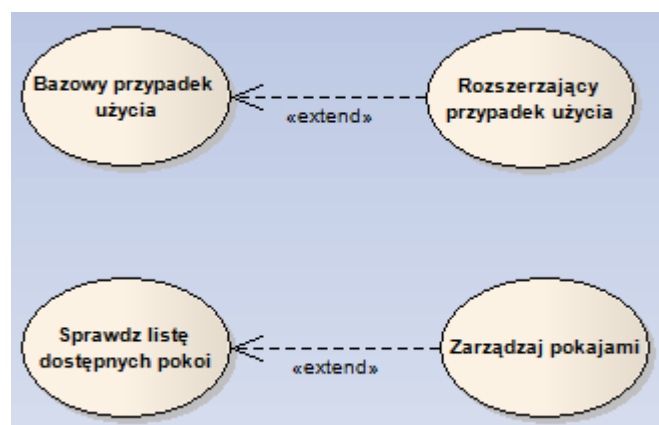
- Dokonanie rezerwacji pociąga za sobą konieczność zweryfikowania dostępności pokoi
- Po wykonaniu przypadku "Sprawdz listę dostępnych pokoi" następuje wykonanie funkcjonalności przypadku "Dokonaj rezerwacji".
- "Sprawdz listę dostępnych pokoi" może być wykorzystywany przez inne przypadki zawierające, które go przywołują.

Rozszerzenie



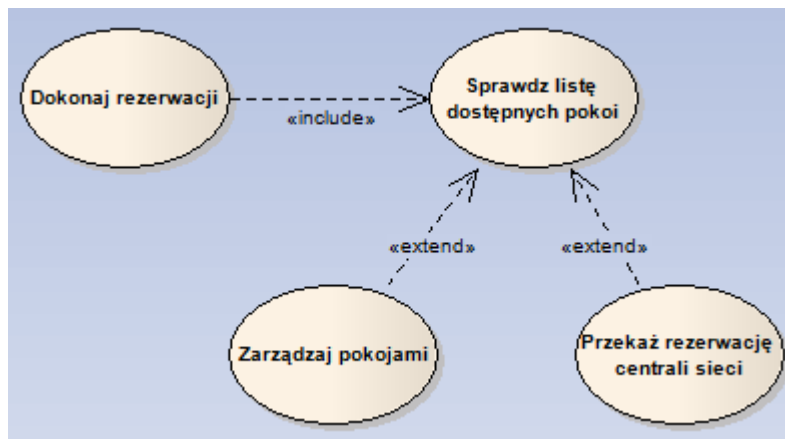
Rozszerzenie (<<extend>>) wskazuje że dany przypadek użycia opcjonalnie rozszerza funkcjonalność bazowego przypadku użycia.

- Funkcjonalność bazowego przypadku użycia jest rozszerzana o inny przypadek użycia.
- Tworzenie zależności rozszerzania znajduje zastosowanie o ile funkcjonalność przez rozszerzony przypadek użycia ma zostać uzupełniona o kilka dodatkowych kroków
- Związek rozszerzenia skierowany jest góram w stronę bazowego przypadku użycia



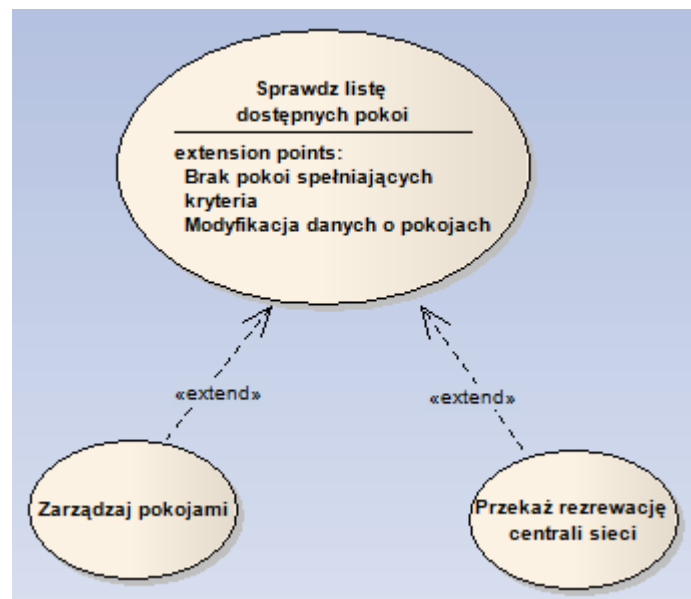
- "Sprawdzenie listy pokoi" nie jest jednoznaczne z "Zarządzaniem pokojami".
- Zarządzanie pokojami uwzględnia:
 - Przypisanie pokojowi innej kategorii,
 - Zmiane ceny wynajmu
 - Zmiane statusu pokoju na "dla niepalących"
 - Czasowe wyłączenie pokoju z użytkowania.

Przykład 3 - Zarządzanie rezerwacją w hotelu



Miejsca rozszerzenia

Przypadek bazowy, może być rozszerzony o przypadki poszerzające pospełnieniu określonych warunków. Warunki te określa się w przypadku bazowym jako miejsca rozszerzenia bądź punkty rozszerzenia.

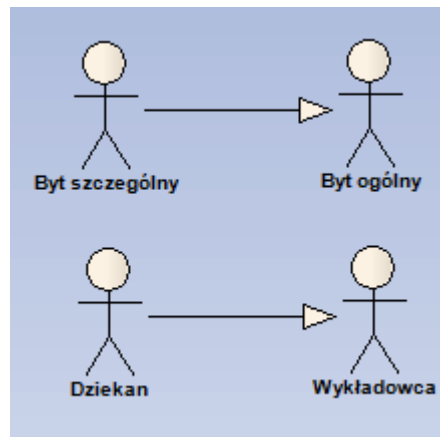


Generalizacja (uogólnienia)



Związek pomiędzy aktorami lub przypadkami użycia, w którym jeden z nich jest bytem ogólnym a drugi bytem szczególnym.

Byt szczególny może zastąpić byt uszczegóławiany.



- Byt ogólny -przodek - Wykładowca
- Byt szczególny - potomek - Dziekan
- Każdy Dziekan może być wykładowcą.
- Wykładowca nie może zastąpić Dziekana.

Przykład 4 - Wykłady

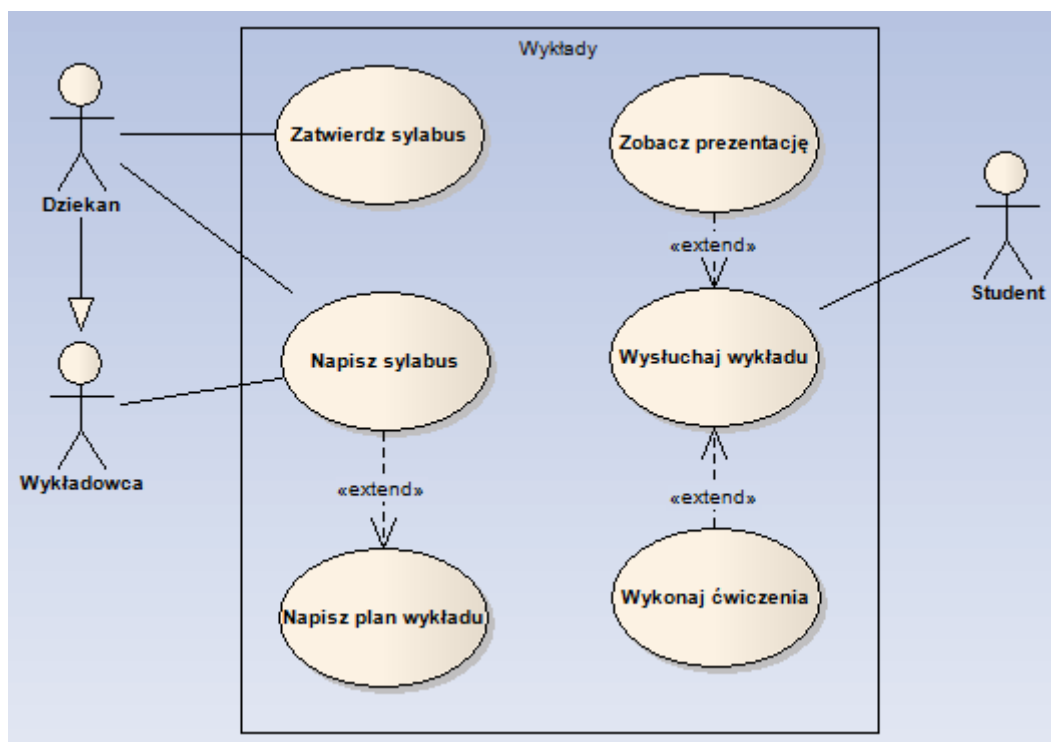


Diagram sekwencji

Konrad Markowski

Diagram sekwencji

- Najczęściej wykorzystywany
- Dokumentuje funkcjonalność przypadków użycia
- Rodzaj diagramu interakcji
- Opisuje interakcję pomiędzy instancjami klasyfikatorów systemu w postaci sekwencji komunikatów wymienianych między nimi

Diagram sekwencji

- Przedstawia koncepcję obiektowego modelowania systemów informatycznych.

Interakcja w dwóch wymiarach:

- **Poziomym** – instancje klasyfikatorów biorące udział w interakcji, ma charakter statyczny
- **Pionowa** – chronologiczne ułożenie komunikatów, ma charakter dynamiczny

Diagram sekwencji

Wyróżniamy **3 rodzaje diagramów sekwencji** w zależności od stopnia abstrakcji:

- koncepcyjny
- implementacyjny
- wystąpieniowy

Diagram konceptualny – ogólny zakres i zawartość interakcji dla danej aplikacji, nie zdefiniowany ściśle w UML 2.0

Diagram implementacyjny – obejmuje przepływy główne oraz alternatywne z odpowiedniego diagramu przypadków użycia, stanowi wystarczającą podstawę do opracowania specyfikacji programistycznej, zazwyczaj odpowiada mu wiele diagramów wystąpieniowych

Diagram wystąpieniowy – odnosi się do ustalonego scenariusza

Diagram sekwencji

Podstawowe pojęcia:

- Klasyfikator (obiekt)
- Komunikat
- Linia życia
- Ośrodek sterowania

Diagram sekwencji

Obiekt (instancja klasyfikatora)

– uogólnia kolekcję instancji o tych samych cechach

Linia życia – powiązanie z konkretną instancją klasyfikatora na diagramie sekwencji, wskazująca okres istnienia tej instancji

Komunikat – specyfikacja wymiany informacji między obiektami, zawierające zlecenie wykonania określonej operacji

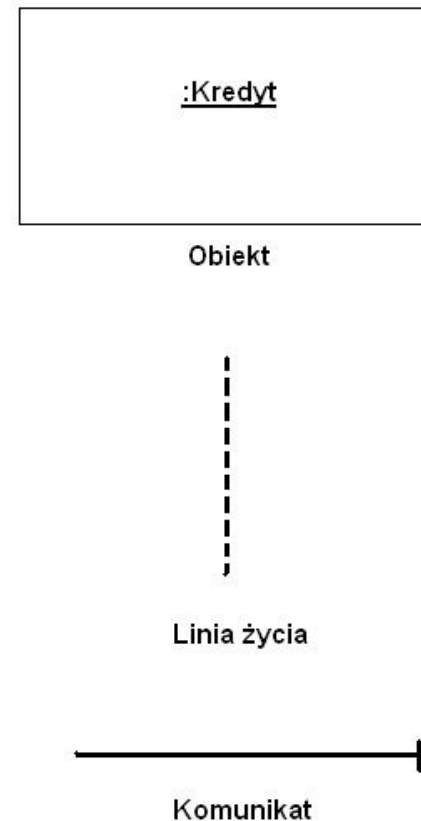


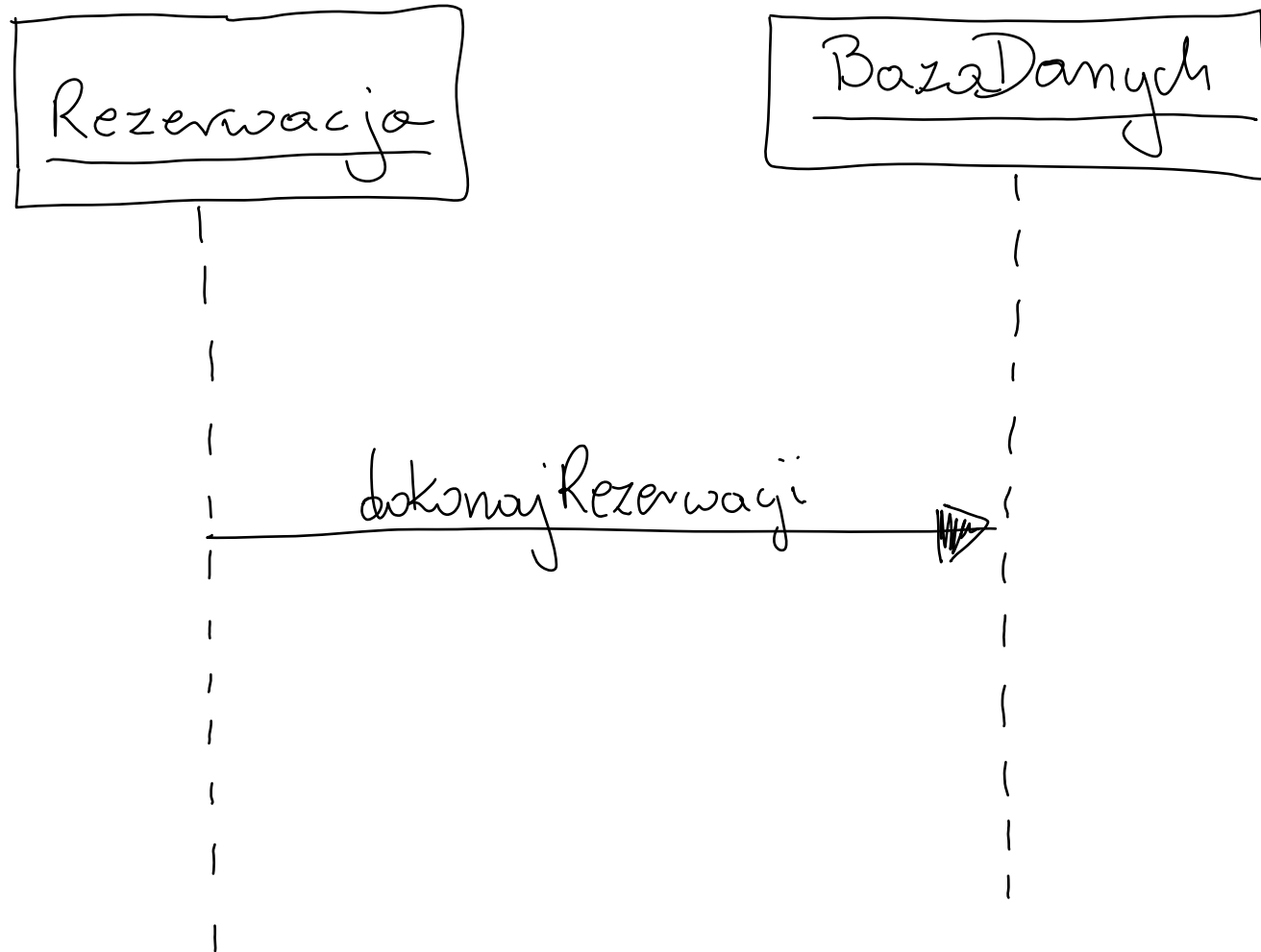
Diagram sekwencji

- Instancje klasyfikatorów ilustrowane są w postaci obiektów.
- Instancje klasyfikatorów umieszczamy na diagramie sekwencji w kolejności ich wystąpienia.
- Z każdej instancji klasyfikatora wyprowadzana jest linia życia.

Diagram sekwencji

- Komunikaty rozmieszcza się na osi pionowej diagramu, pomiędzy liniami życia instancji klasyfikatorów.
- Komunikaty porządkowane są według kolejności ich wystąpienia, im wyżej tym wcześniej występuje.
- Komunikaty można numerować oraz umieszczać przy nich parametry operacji
- Komunikaty można przesyłać między dowolnymi liniami życia.

PRZESYŁANIE KOMUNIKATU



SPRAWDZANIE STANU KONTA

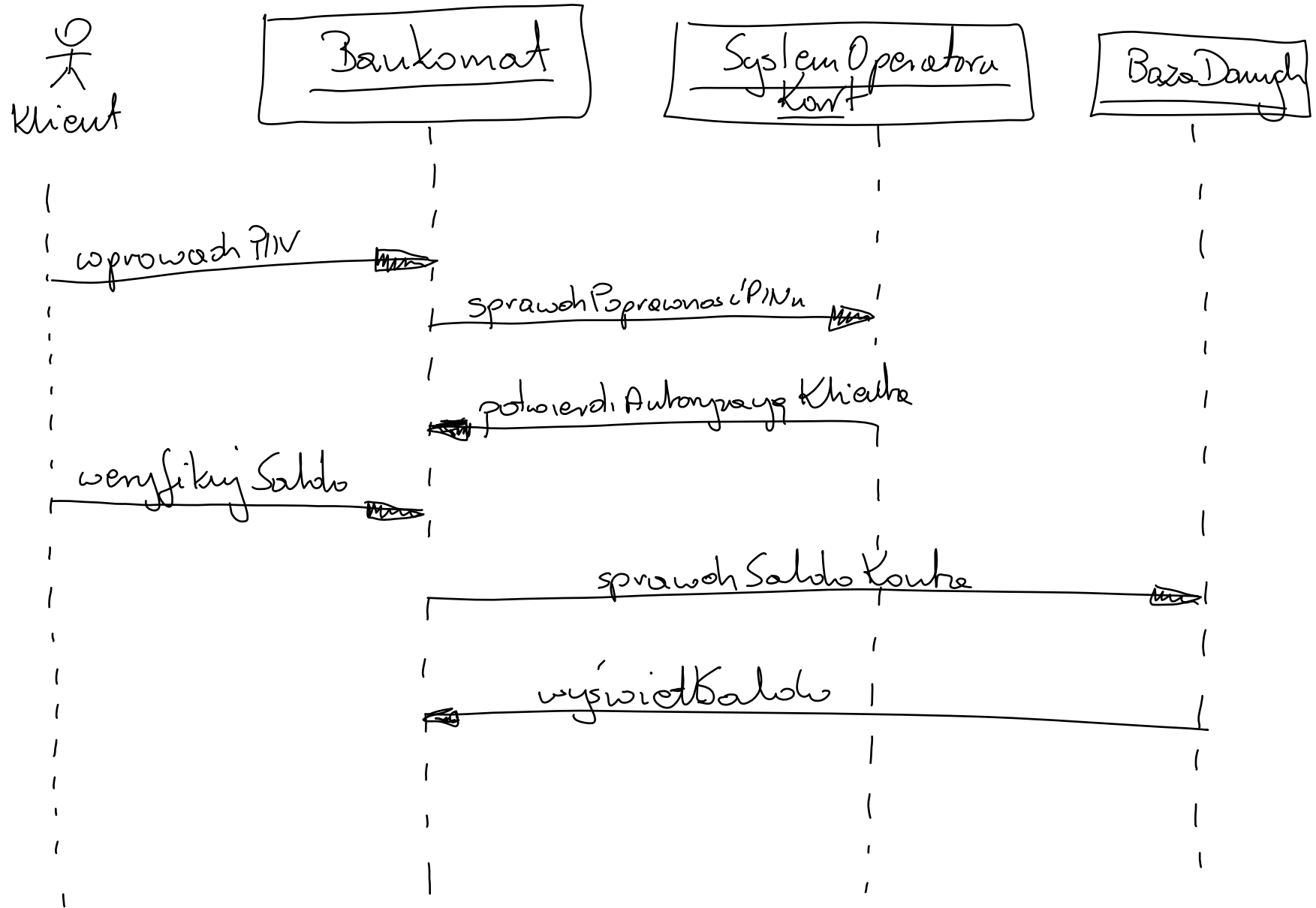
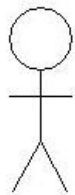
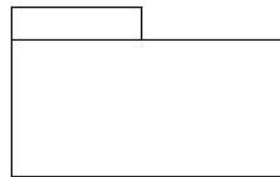


Diagram sekwencji

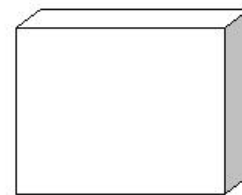
Rodzaje klasyfikatorów



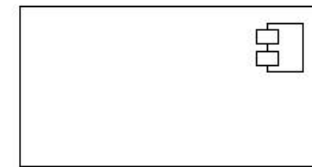
Aktor



Pakiet



Węzel



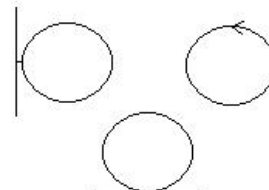
Komponent



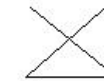
Przypadek użycia



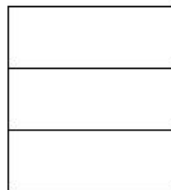
Współdziałanie



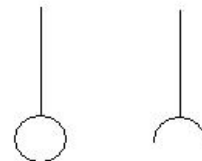
Klasa analityczna



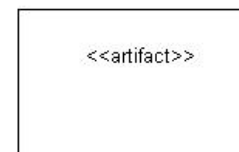
Sygnal



Klasa



Interfejs



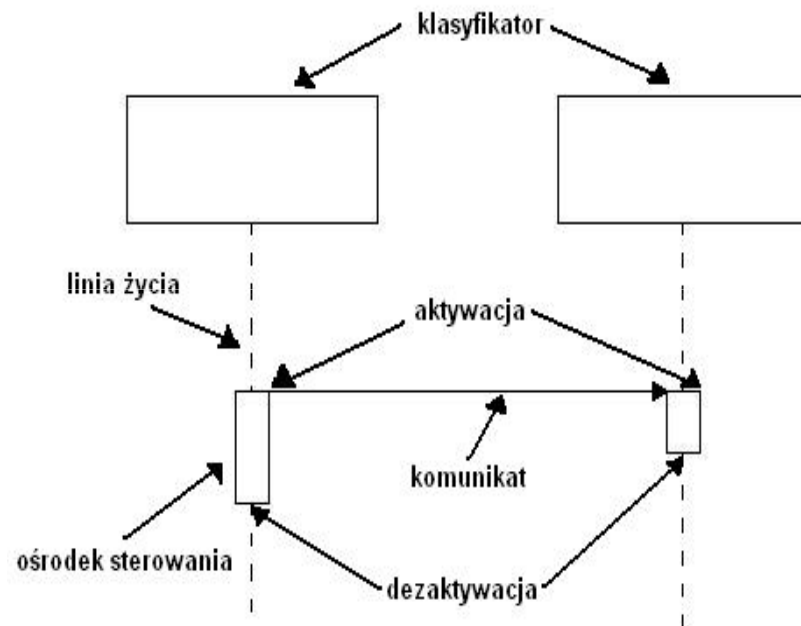
Artefakt

Diagram sekwencji

Ośrodek sterowania jest to specyfikacja wykonania czynności, operacji lub innej jednostki zachowania w ramach interakcji.

Jest inicjowany aktywacją, a przejście w stan czuwania dezaktywacją.

Może istnieć kilka niezależnych ośrodków sterowania związanych z daną instancją klasyfikatora.



REZERWACJA HOTELOWA

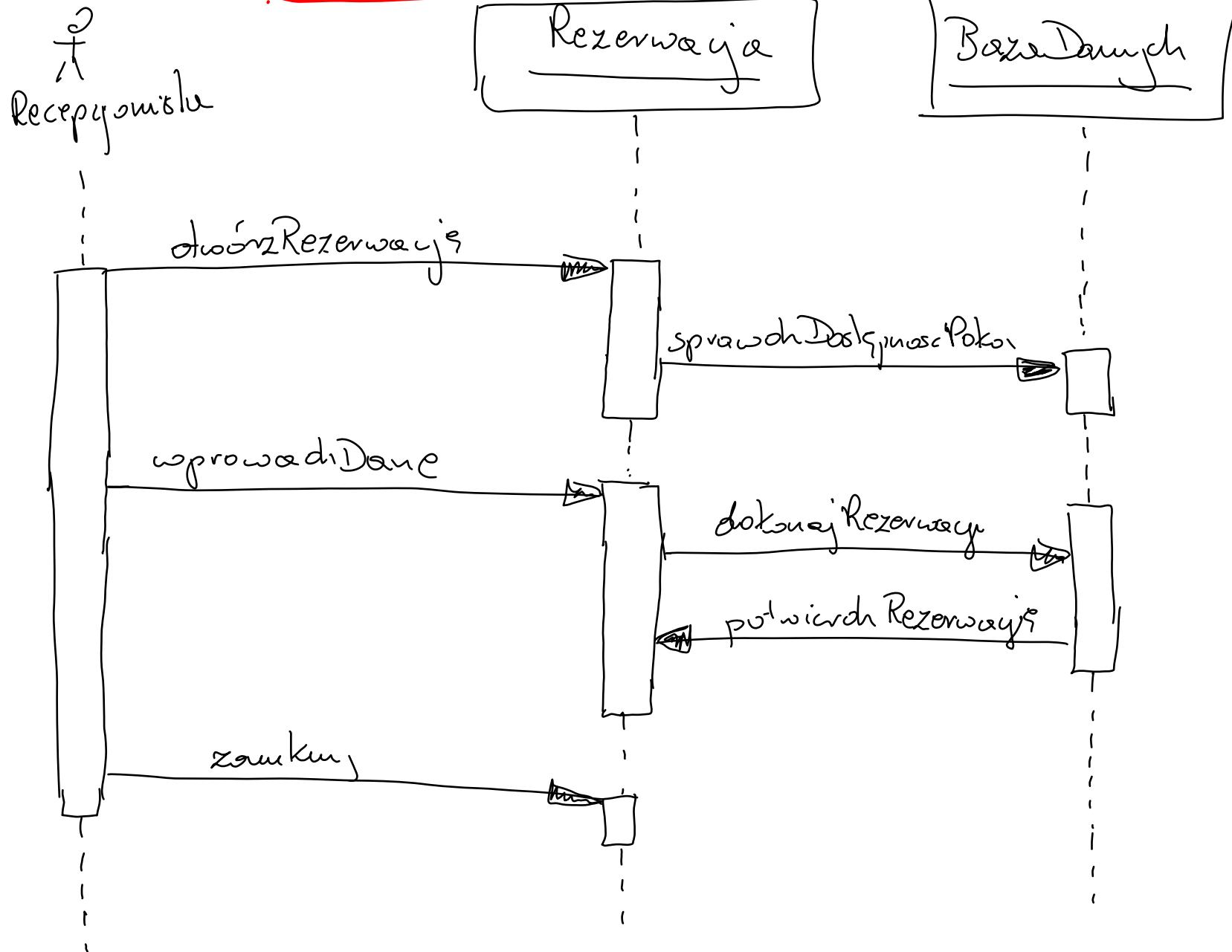


Diagram sekwencji

- Pojęcia zaawansowane:
 - rodzaje komunikatów
 - tworzenie i niszczenie obiektów
 - warunki
 - samo wywoływanie
 - iteracja
 - rozgałęzienie
 - fragmenty wyodrębnione i operatory interakcji
 - przywoływanie wystąpienia interakcji
 - bramy

Diagram sekwencji

Rodzaje komunikatów:

- synchroniczny
- asynchroniczny
- zwrotny
- utracony
- znaleziony
- opcjonalny
- oczekujący

Diagram sekwencji

Podział rodzajów komunikatów:

- kompletne – synchroniczny, asynchroniczny, zwrotny, opcjonalny, oczekujący
- niekompletne – utracony, znaleziony

Komunikat synchroniczny powoduje przerwanie przepływu sterowania u klasyfikatora-nadawcy, jest wznowiany po wykonaniu operacji przez klasyfikator-odbiorcę.



Komunikat asynchroniczny nie powoduje przerwania aktualnego przepływu sterowania nadawcy.



Komunikat zwrotny wskazuje powrót sterowania do instancji klasyfikatora po wykonaniu komunikatu synchronicznego.



Diagram sekwencji

Komunikat utracony – kiedy odbiorca jest nieznany.



Komunikat znaleziony – kiedy nieznany jest nadawca.



Komunikat opcjonalny – komunikat po wysłaniu do odbiorcy, jeśli nie zostanie niezwłocznie obsłużony przez odbiorcę, to nadawca więcej nie próbuje go wysłać, nie jest elementem standardu UML 2.0



Komunikat oczekujący – komunikat nadany do odbiorcy, albo jest obsłużony natychmiast lub czeka odpowiedni okres czasu na wykonanie, po upływie tego czasu nadawca rezygnuje z danej interakcji, nie jest elementem standardu UML 2.0.



Magazyn

Artykuł

zmień Cenę
Komunikat oczekuje
na odpowiedź

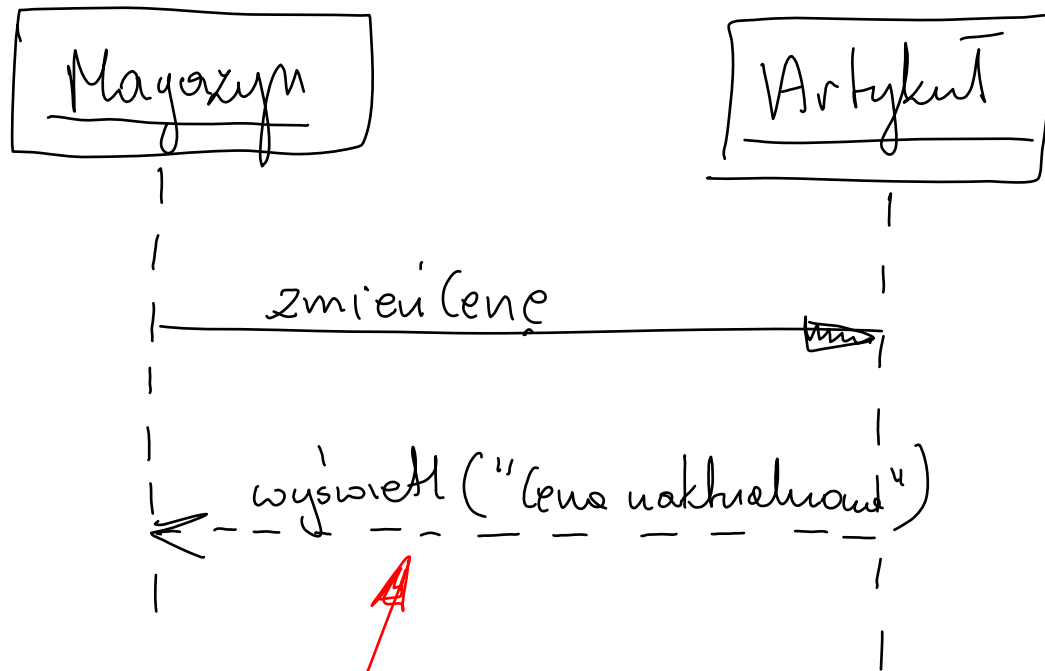
Komunikat
Synchroniczny

Nadawca

Satelita

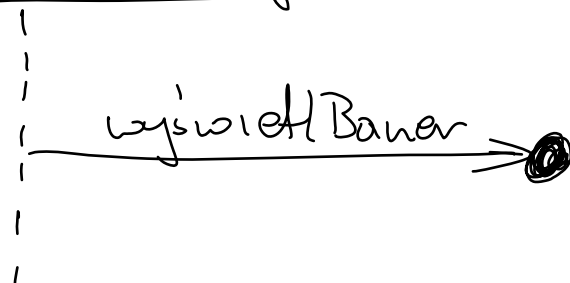
odbierz Sygnał
Komunikat nie
oczekuje na odpowiedź

Komunikat
asynchroniczny



powrót sterowania do instancji
klasy Kalora

Portal Internetowy

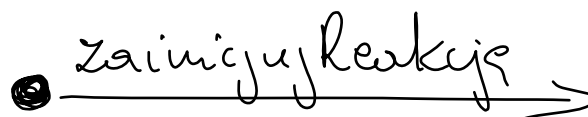


Komunikat
utrącony

Nadawca jest znany,
odbiorca nieznany

Komunikat
znaleziony

Oznajmk



Odbiorca jest znany,
nadawca nieznany

Magazyn

Baza Danych

Komunikat
opcjonalny

sortuj Artykuł (indeks) ←

Nadawca wysyła do odbiorcy komunikat, oczekując gotowości do jego niezwłocznej obsługi przez odbiorcę

Magazyn

Baza Danych

Komunikat
oczekujący

na wcześniejsze Potwierdzenie

Nadawca wysyła komunikat do odbiorcy i jest gotów czekać na jego potwierdzenie przez określony odstęp czasu

Diagram sekwencji

Warunek komunikatu

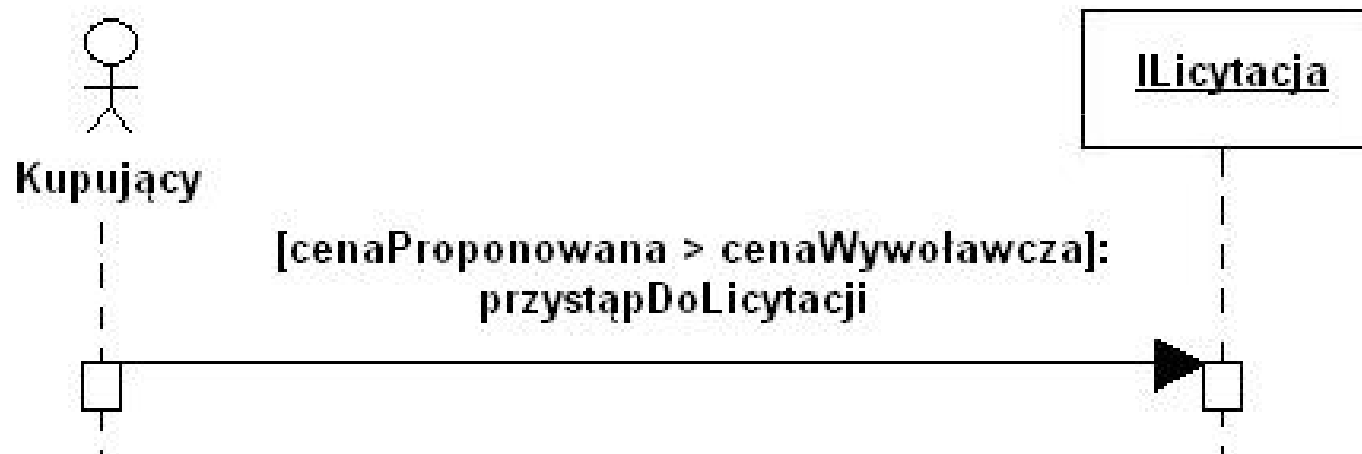


Diagram sekwencji

Samowywołanie – dana instancja klasyfikatora
wywołuje własną operację.

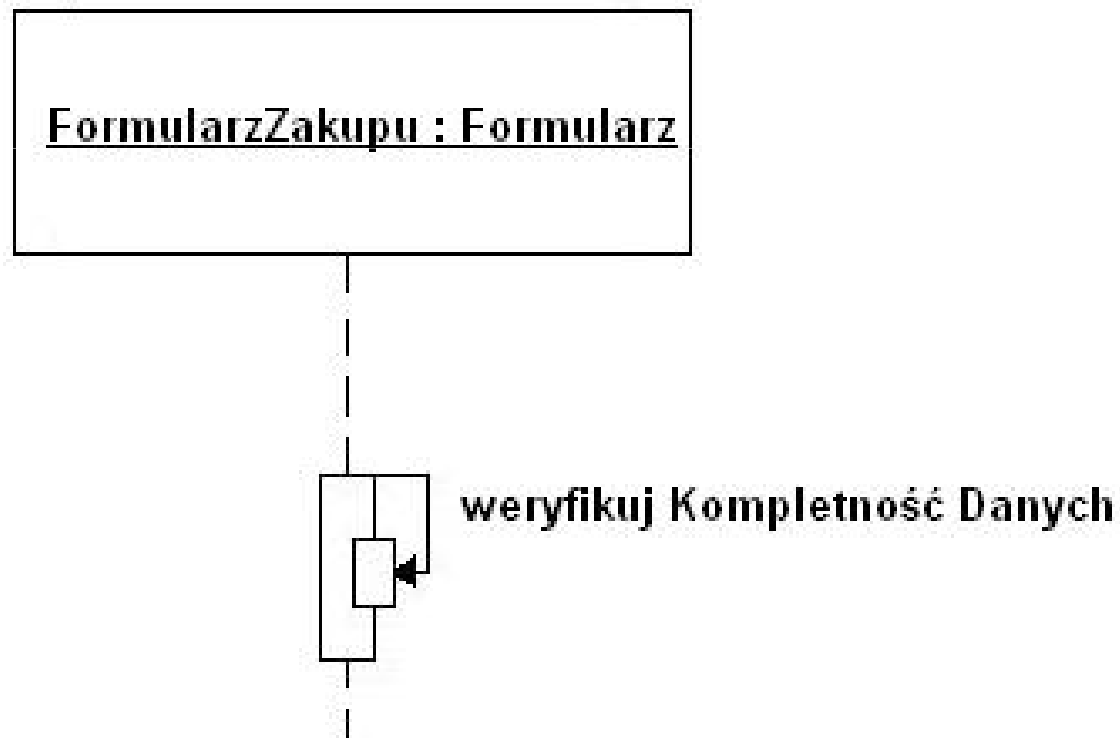


Diagram sekwencji

Iteracja – wielokrotne, policzalne powtórzenie jednego komunikatu

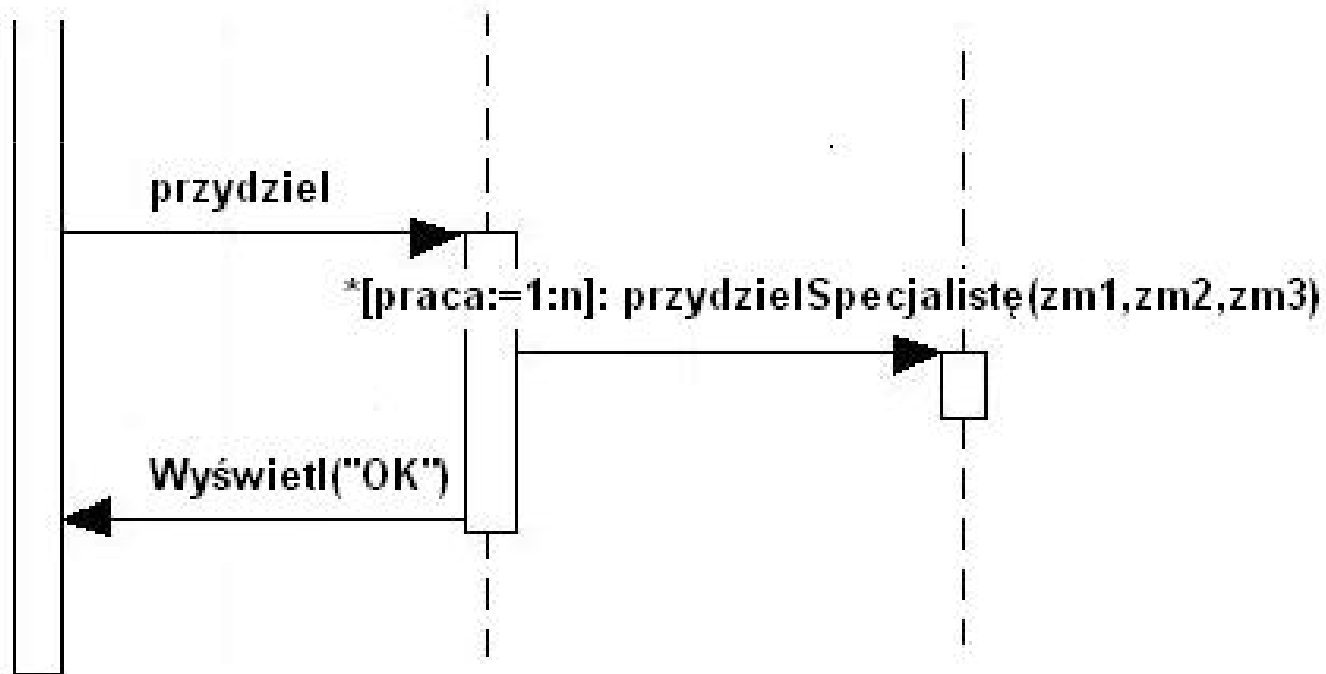
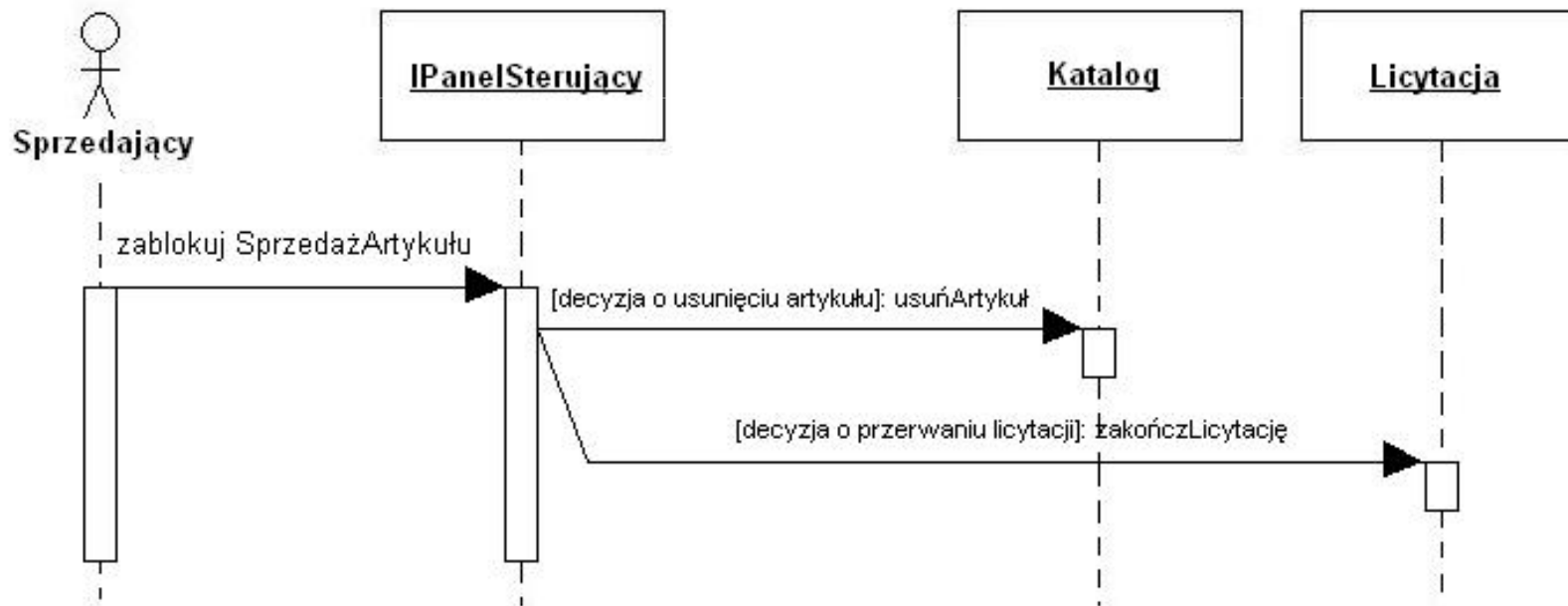


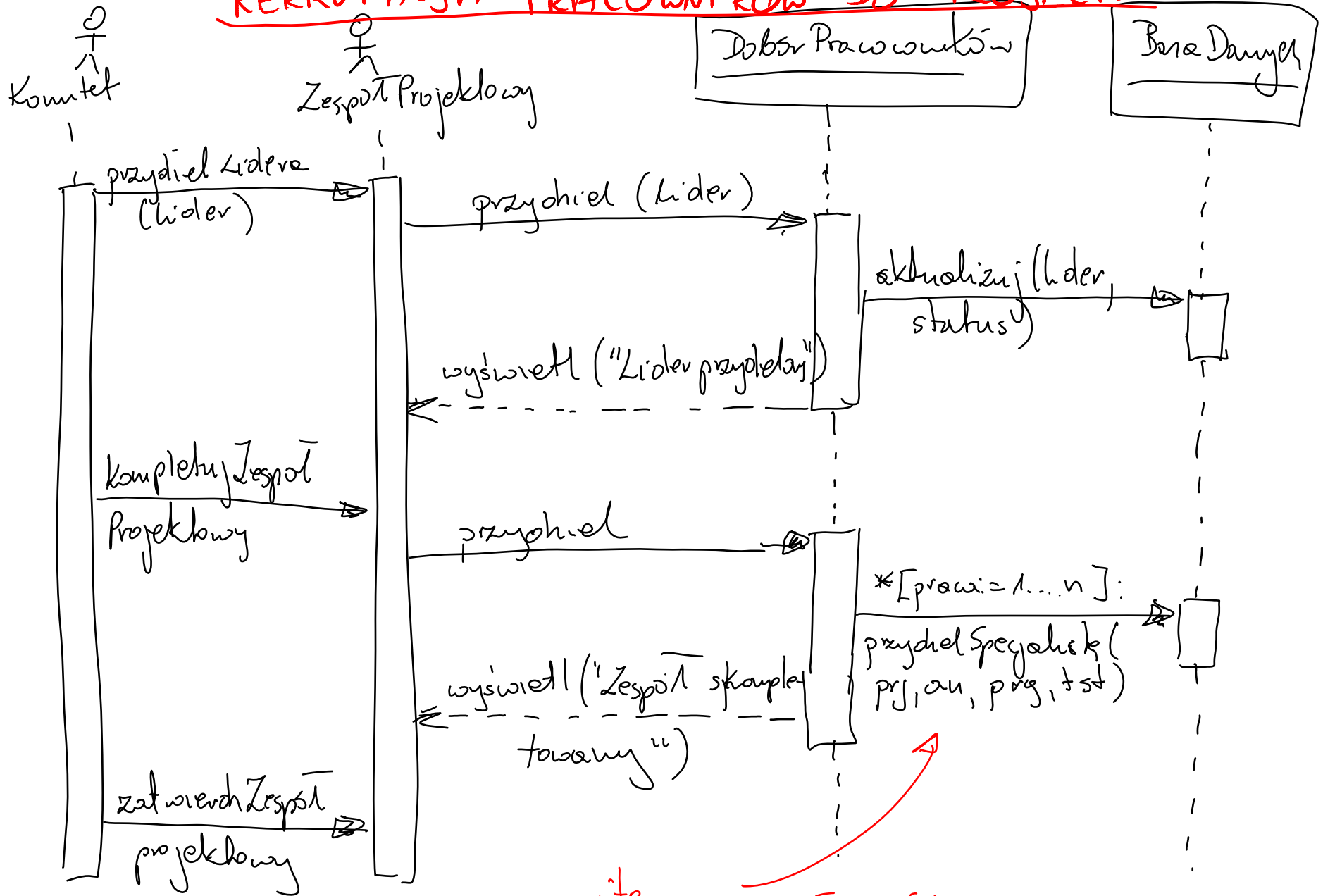
Diagram sekwencji

Rozgałęzienie

- Klasyfikator – nadawca → kilka klasyfikatorów –odbiorców
- Klasyfikator - nadawca → jeden klasyfikator - odbiorca



REKRUTACJA PRACOWNIKÓW DO PROJEKTU



iteracja * [specyfikacja iteracji] : nazwa operacji

Diagram sekwencji

Proces tworzenia diagramu sekwencji

1. Analiza przypadku użycia oraz jego scenariuszy
2. Identyfikacja klasyfikatorów uczestniczących w interakcji
3. Konceptualny diagram sekwencji
4. Implementacyjny diagram sekwencji

DIAGRAM KLAS

Konrad MARKOWSKI

Znaczenie diagramu klas

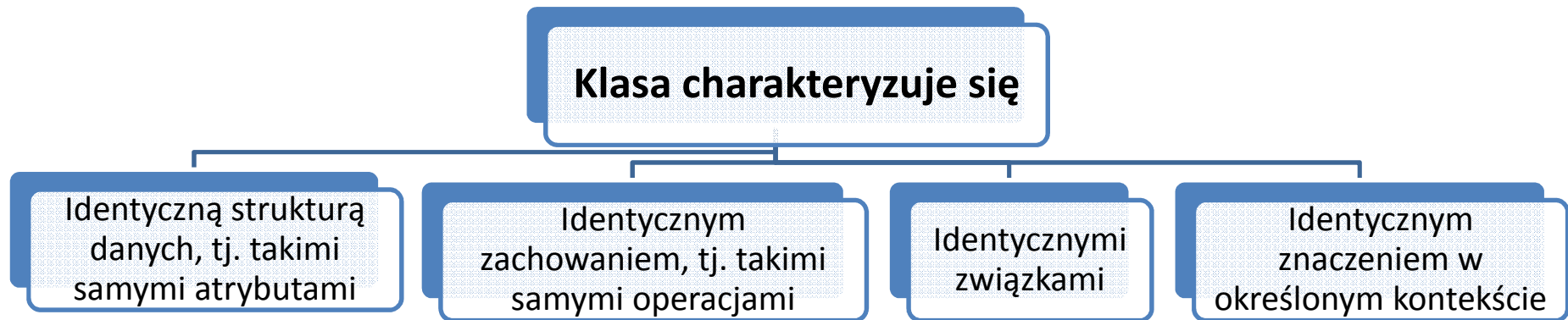
Diagram klas to graficzne przedstawienie statycznych, deklaratywnych elementów dziedziny przedmiotowej oraz związków między nimi

Podstawowe kategorie pojęciowe oraz notacja graficzna

Obiektem jest każdy byt – pojęcie lub rzecz – mający znaczenie w kontekście rozwiązywania problemu w danej dziedzinie przedmiotowej.

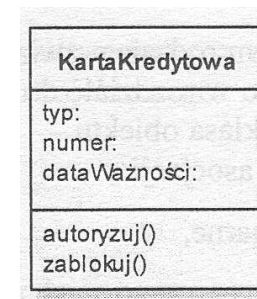
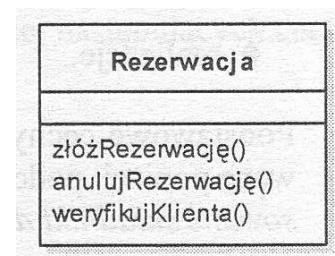
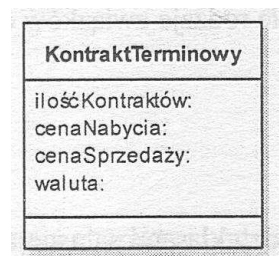
1. Wszystko, co wiadomo o obiekcie, jest reprezentowane przez wartości atrybutów – czyli cech statycznych tego obiektu
2. Zachowanie obiektu wyrażone jest w operacjach określających usługi, które oferuje obiekt
3. Obiekt charakteryzuje się unikalną tożsamością

Dowolny obiekt jest instancją abstrakcyjnego pojęcia, jakim jest **klasa** (z ang. class) **obiektu**.

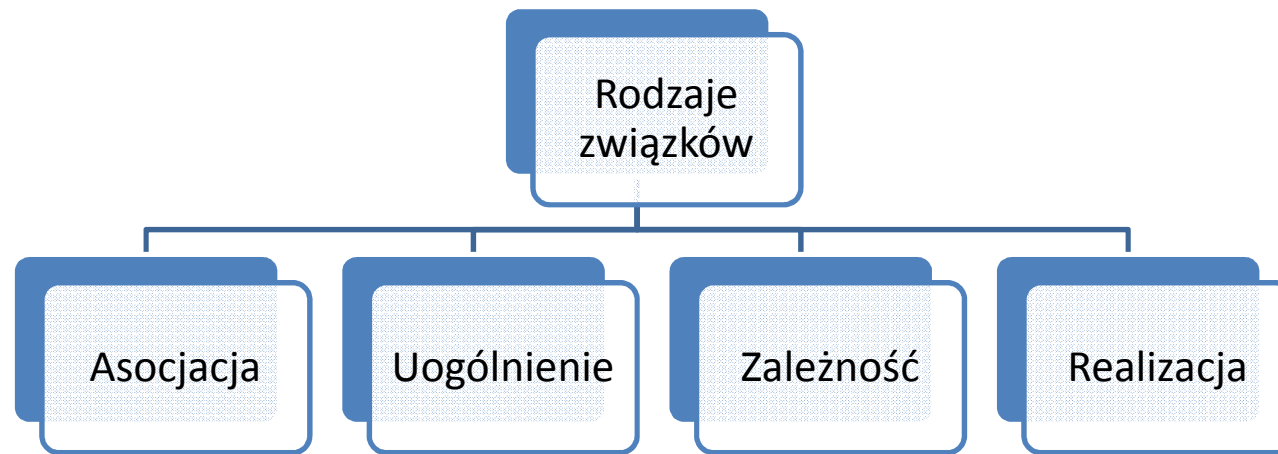


Klasa jest uogólnieniem zbioru obiektów, które mają takie same atrybuty, operacje, związki i znaczenie.

1. Każda klasa zawiera zestaw informacji istotny z punktu widzenia kontekstu modelowanego procesu.
2. Diagram klas standardowo przedstawia się jako prostokąt złożony z trzech sekcji:
 - Nazwy klasy
 - Zestawu atrybutów
 - Zestawu operacji
3. Możliwe kombinacje graficznej prezentacji klas:
 - Sama nazwa klasy
 - Nazwa klasy z zestawem atrybutów
 - Nazwa klasy z zestawem operacji
 - Nazwa klasy z zestawem atrybutów i operacji

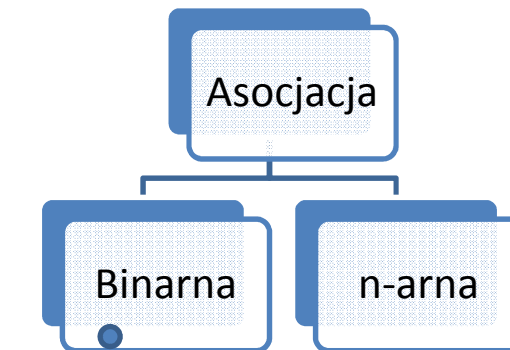


1. Klasy obiektów powiązane są różnego rodzaju związkami.
2. W diagramie klas stosuje się powszechnie **cztery rodzaje związków**.



Asocjacja

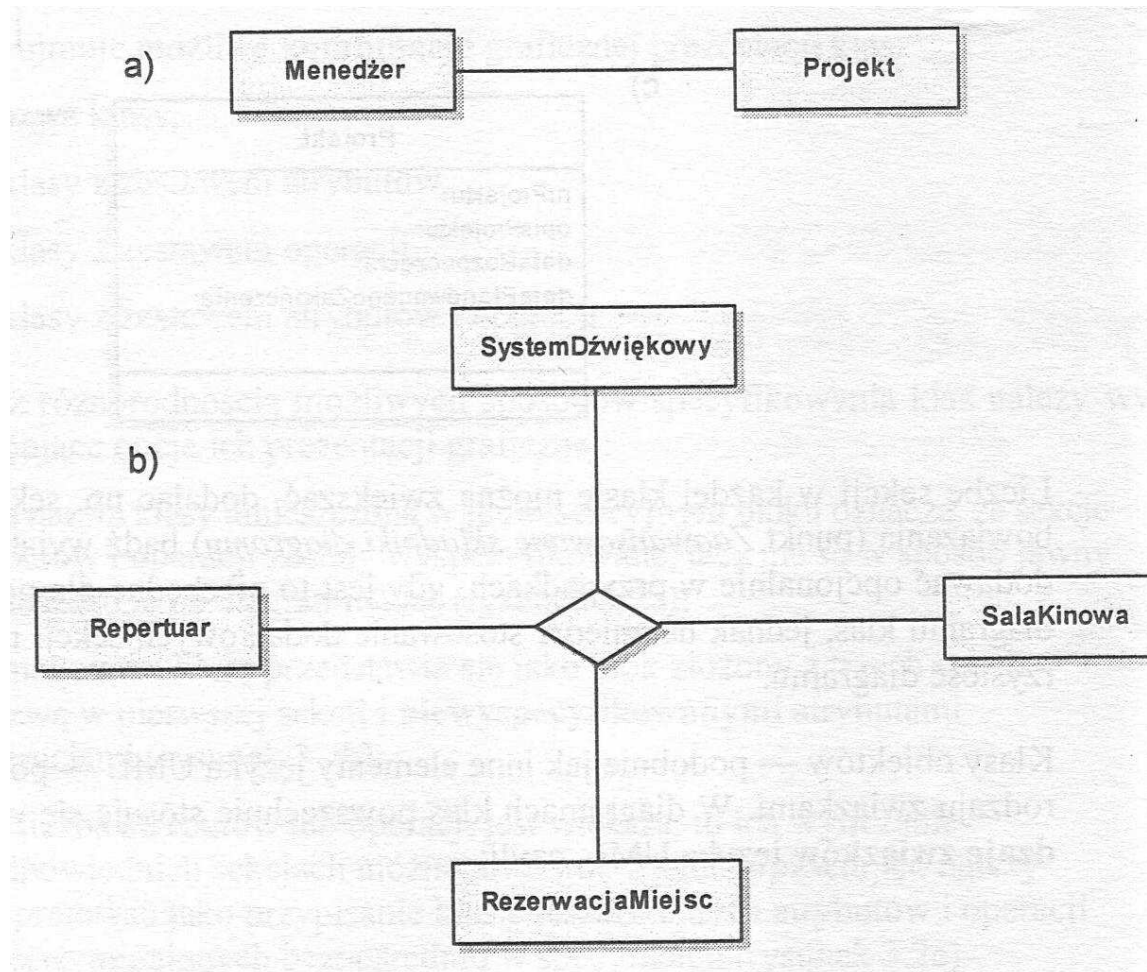
Asocjacja opisuje zbiór powiązań pomiędzy obiektami

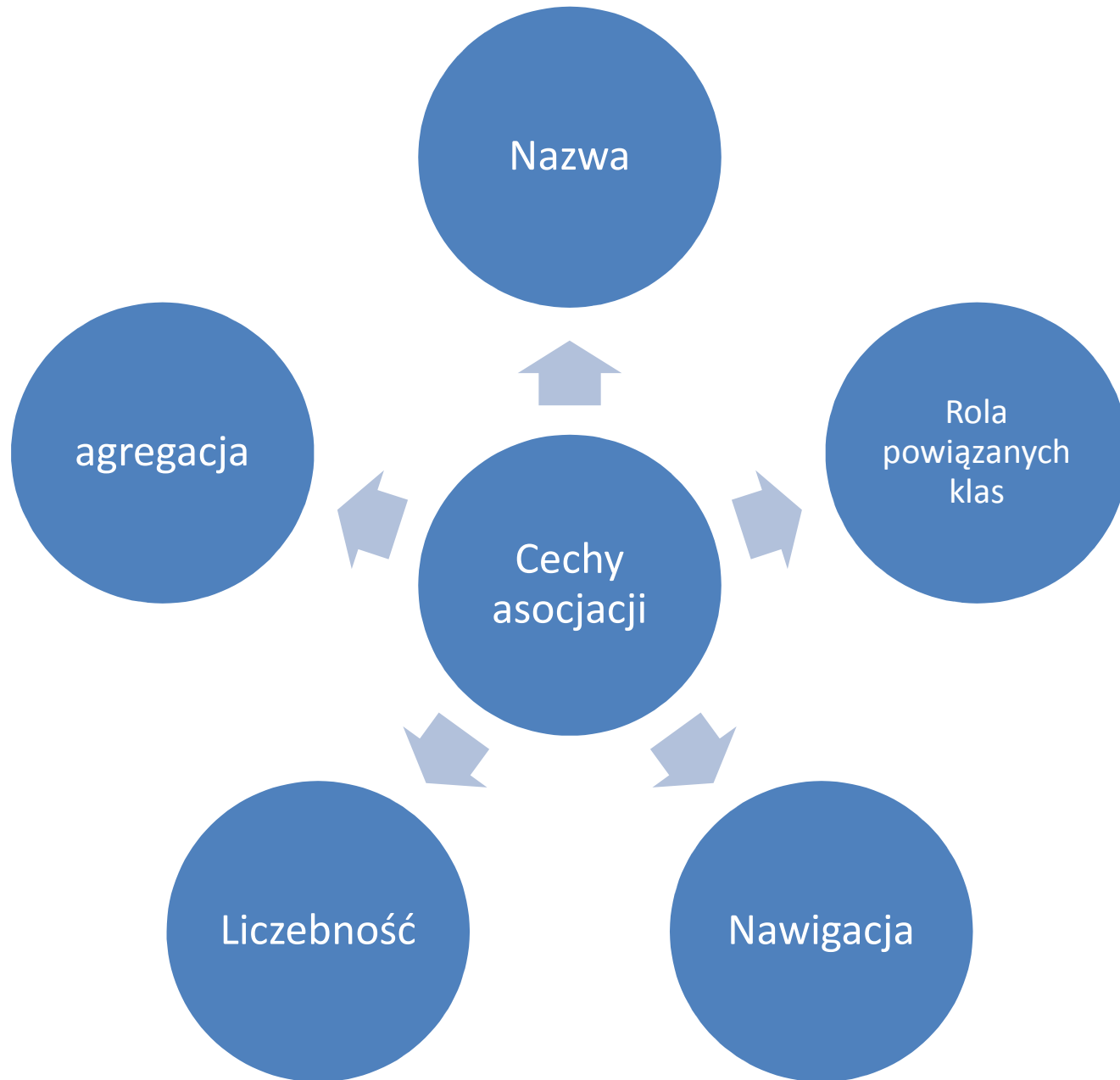


W praktyce
dominuje
asocjacja binarna

Przykład:

- a. Asocjacja binarna na przykładzie harmonogramowania projektu
- b. Asocjacja n-arna na przykładzie systemu kinowego

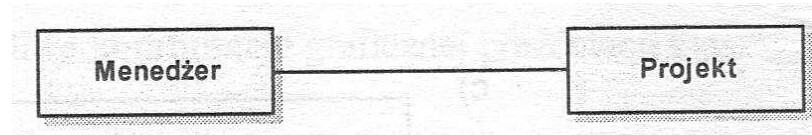




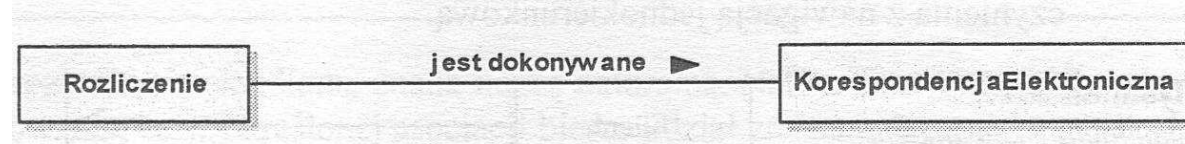
Nazwa asocjacji, role

Istnieje szereg możliwości oznaczania nazw asocjacji. Mogą one być:

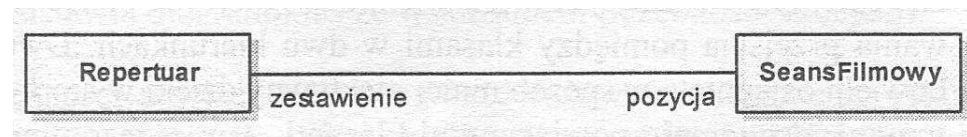
1. Nienazwane



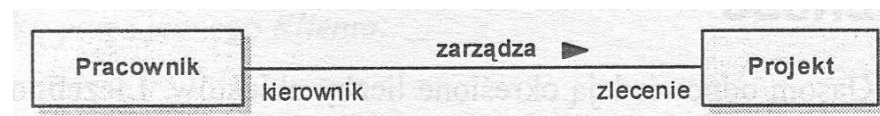
2. Nazwane z opcjonalnym zamieszczeniem znacznika wskazującego kierunek interpretacji asocjacji



3. Scharakteryzowane poprzez role klas pełnione w asocjacji

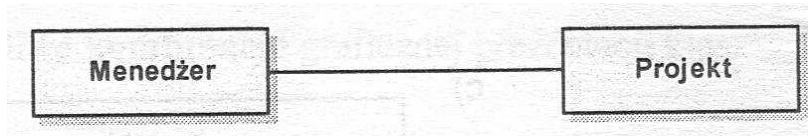


4. Nazwane i równocześnie scharakteryzowane przez role

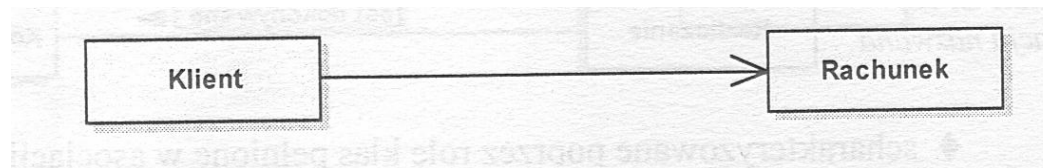


Nawigacja

1. Do wykonywani operacji na obiektach poszczególnych klas niezbędne jest przesyłanie komunikatów pomiędzy nimi.
2. Nawigacja:
 - Asocjacja

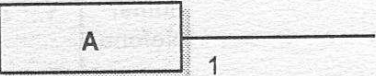

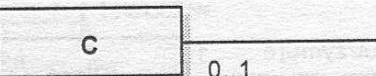


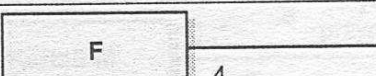

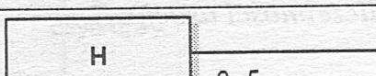
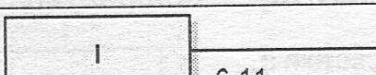




- Asocjacja z kierunkiem nawigacji



Liczebność

Liczebność to specyfikacja zakresu dopuszczalnej liczby obiektów danej klasy biorącej udział w danym związku

Oznaczenie	Opis	Przykład
1	dokładnie jeden	
1..*	jeden lub wiele	
0..1	zero lub jeden	
*	wiele	
0..*	zero lub wiele	
n	dokładnie n (n > 1)	
1..n	od 1 do n	
0..n	od 0 do n	
n..m	od n do m (n, m > 1)	
n..*	więcej niż n	
n, m, o..p, q	liczebność złożona	

Przykład. Operator komórkowy

