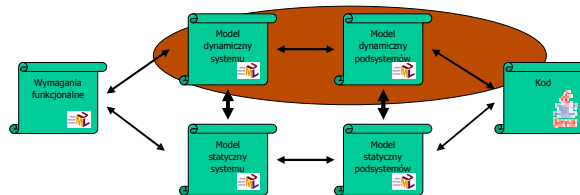


Michał Śmiałek
ZETiIS, PW



Model interakcji

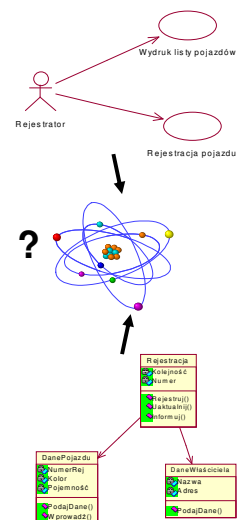
- Dynamika systemu – interakcje obiektów
- Diagramy sekwencji i diagramy współpracy
- Dynamika dla systemu i podsystemów
- Generacja kodu

<http://www.iem.pw.edu.pl/~smialek/apo>

Michał Śmiałek
ZETiIS, PW

Modelowanie dynamiki systemu

- Model przypadków użycia wymienia różne sposoby zachowania się systemu w stosunku do jego użytkowników.
- Model klas lub model komponentów określa statyczne zależności między elementami całego systemu lub jednego z podsystemów.
- Elementem łączącym obydwa modele jest model interakcji.
- Dzięki diagramom interakcji możemy zachować ścisłą relację między dwoma podstawowymi modelami. Są one spoiwem, dzięki któremu możliwy jest stopniowa, równoległa rozbudowa modelu systemu.

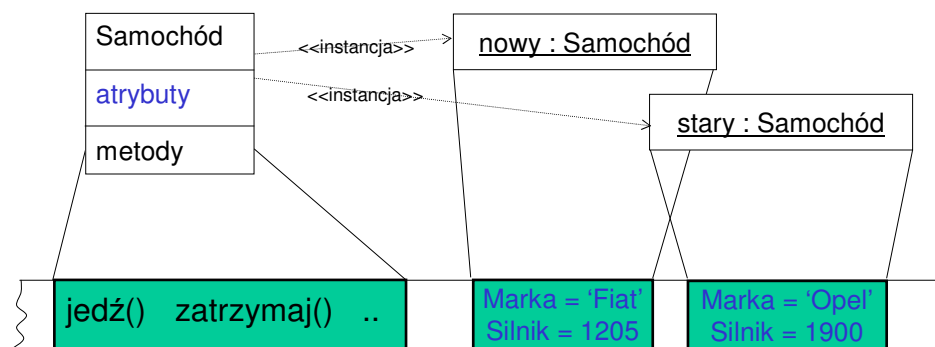


<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Obiekty - elementy modelu dynamicznego

- Obiekt jest elementem istniejącym podczas wykonywania się programu. Stan obiektu przechowywany jest w pamięci.
- Notacja obiektów: prostokąt z nazwą obiektu i nazwą klasy. Nazwa obiektu jest podkreślona dla odróżnienia od nazwy klasy.

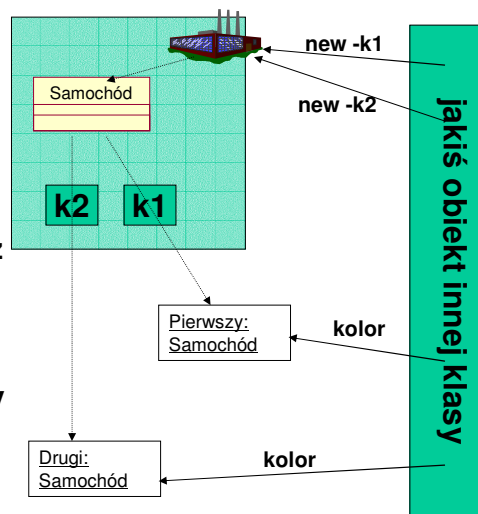


<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Dynamika: konstrukcja obiektów

- Konstrukcja obiektu wykonywana jest podczas jego tworzenia.
- Konstrukcja polega na nadaniu wartości atrybutom obiektu, na utworzeniu obiektów agregowanych oraz na powiązaniu obiektu z innymi obiektami.
- Konstrukcja obiektu może przebiegać na różne sposoby - w zależności od życzenia klienta. Klasa może zatem dostarczać kilku różnych metod konstrukcji.

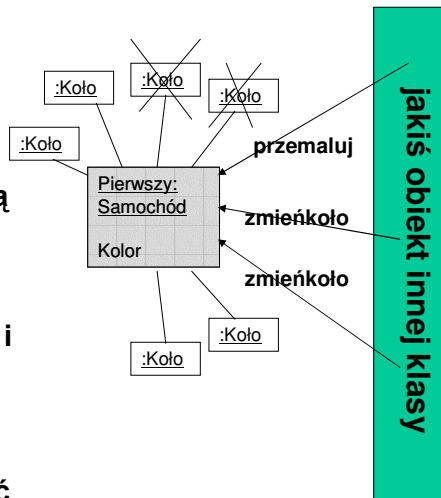


<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Dynamika: zmiany stanu podczas życia obiektu

- Po wyprodukowaniu, obiekt ma zainicjowaną strukturę danych - znajduje się w stanie początkowym.
- Podczas swego życia, obiekt podlega zmianom stanu. Zmieniają się wartości atrybutów oraz powiązania z innymi obiektami. Elementy składowe obiektu (agregowane) mogą być tworzone i niszczone.
- Wszystkie możliwe zmiany stanu obiektu określają jego dynamikę.
- Zmiany stanu obiektu powinny być realizowane przez wąski interfejs.

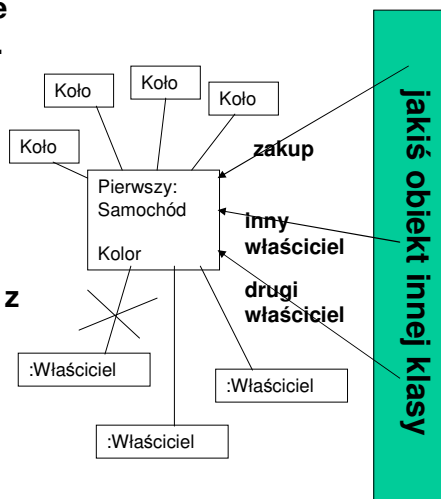


<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Dynamika: zmiana związków z innymi obiektami

- Podczas swego życia, obiekt może mieć związek z różnymi obiektami. Wszystkie możliwe związki i ich krotności uwzględnione są na diagramie klas.
- Powiązania z innymi obiektami mogą się zmieniać w sposób dynamiczny. Obiekt w pewnym momencie może być np. związany z dwoma, a w innym momencie - z pięcioma obiektami składowymi.
- Zmiana powiązania następuje w wyniku wywołaniu metody, która usuwa je lub tworzy powiązanie z obiektem podanym np. jako parametr.

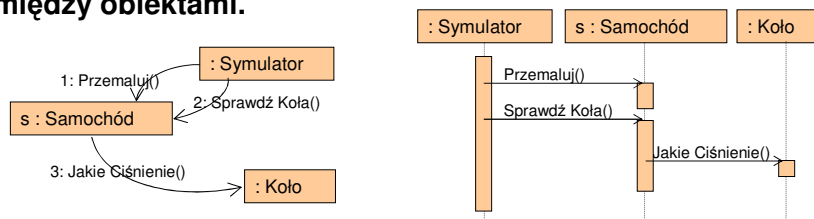


<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Opis dynamiki systemu - diagramy interakcji

- Język UML umożliwia prezentację dynamiki działania systemu (patrz poprzednie slajdy). Służą temu diagramy interakcji.
- Dwa typy diagramów interakcji - diagramy sekwencji oraz diagramy współpracy. Obydwa typy prezentują tę samą informację, jednak w innej formie (dalej opiszemy tylko diagramy sekwencji).
- Diagramy sekwencji przedstawiają wprost kolejność przesyłania komunikatów między obiektami. Diagramy współpracy większy nacisk niż na porządek komunikatów kładą na fakt ich wymiany pomiędzy obiektami.



<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Co to jest diagram sekwencji?

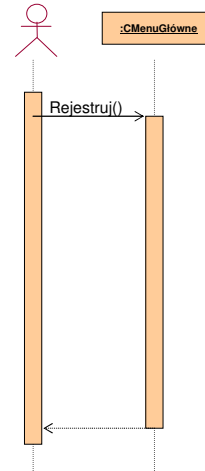
- Prezentacja komunikacji pomiędzy obiektami klas współpracującymi przy realizacji przypadku użycia.
- Komunikacja jest inicjowana przez aktora.
- Na diagramie widać kolejność (z góry na dół) przekazywanych komunikatów rozpoczynających wykonywanie przez obiekty stosownych operacji.
- Z reguły prezentowany jest tylko jeden scenariusz, ale standard notacji UML dopuszcza stosowanie warunków.
- Diagram sekwencji ściśle wiąże się z konkretnym scenariuszem. Zazwyczaj tworzy się po jednym diagramie sekwencji dla każdego scenariusza przypadku użycia.

<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Elementy składowe diagramu sekwencji

- **Pionowe kolumny to kolumny obiektów.**
- **Ludzik - obiekt aktora.**
 - Aktor inicjujący rysowany jest najczęściej po lewej stronie.
- **Prostokąt - obiekt klasy.**
 - Zawiera nazwę obiektu i nazwę klasy.
- **Przerywana pionowa linia - linia życia obiektu.**
- **Strzałka - komunikat.**
 - Komunikat zazwyczaj posiada nazwę i parametry.
- **Rozszerzona belka - belka operacji.**
 - Po skończeniu operacji następuje powrót (przesyłany jest komunikat zwrotny); może być zaznaczony przerywaną strzałką.

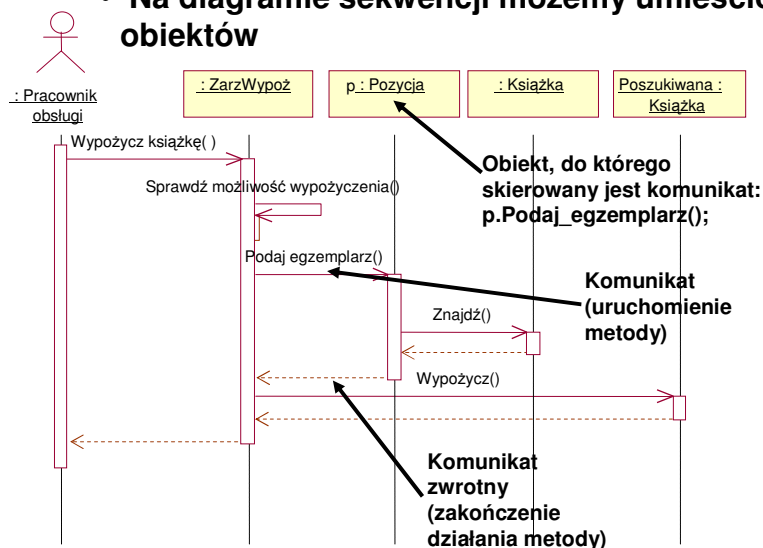


<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Przykładowy diagram sekwencji

- Na diagramie sekwencji możemy umieścić wiele obiektów



<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Konwencje dla diagramie sekwencji

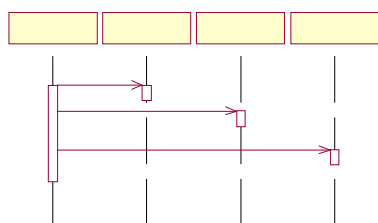
- Pionowa kolumna na diagramie sekwencji może oznaczać zarówno obiekt o konkretnej nazwie lub nieokreślony obiekt danej klasy.
- Na pierwszych etapach modelowania nie określa się nazw obiektów. W fazie implementacji obiekty powinny mieć swoje nazwy, gdyż przekładają się bezpośrednio na pisany kod.
- Na jednym diagramie sekwencji może występować kilka obiektów jednej klasy - powinny jednak być wtedy nazwane.
- Linie odpowiadającą aktorowi inicjującemu akcję umieszcza się po lewej stronie diagramu.
- Jeżeli w realizacji przypadku bierze udział więcej aktorów, to należy umieszczać ich na skrajach diagramu.

<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

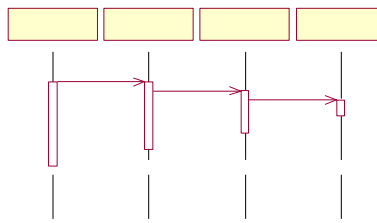
Sposoby współpracy obiektów

Scentralizowany



i

Zdecentralizowany



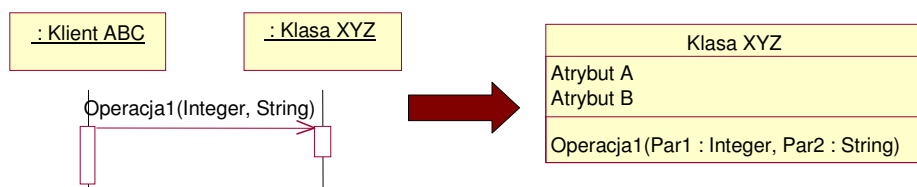
- Przy scentralizowanym sposobie wymiany komunikatów jeden z obiektów kontroluje cały przebieg przypadku - steruje operacjami i pośredniczy w wymianie danych.
- Przy podejściu zdecentralizowanym nie ma obiektu kontrolującego - obiekty komunikują się ze sobą bezpośrednio.

<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiełek
ZETiIS, PW

Modelowanie sekwencji i modelowanie klas

- Modelując realizację przypadków użycia określa się komunikaty wymieniane pomiędzy obiektami (klasami). Komunikaty te powinny odpowiadać operacjom udostępnianym przez te klasy.
- Specyfikując te operacje uzupełniamy utworzony wcześniej statyczny model klas.
- Dla operacji definiuje się typ, parametry wejściowe i wyjściowe.



<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiełek
ZETiIS, PW

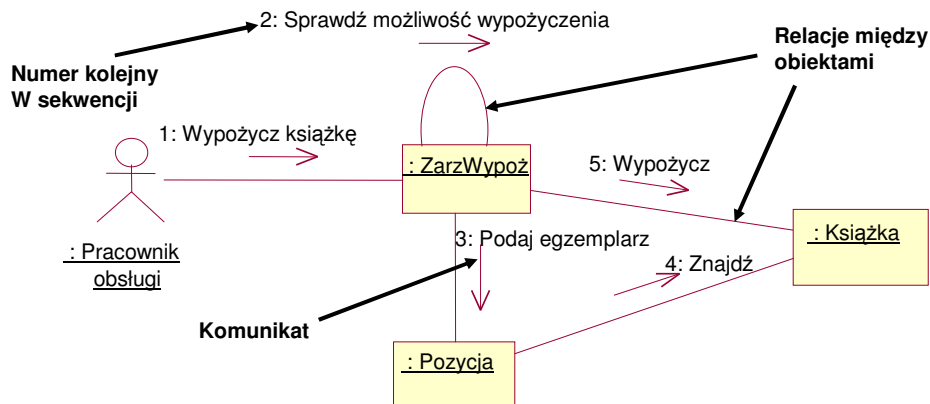
Diagram współpracy obiektów (kolaboracji)

- Diagram współpracy ma podobną zawartość informacyjną co diagram sekwencji.
- Różnica polega na sposobie prezentacji - na diagramie współpracy nacisk jest położony na to jakie role pełnią dane obiekty, zaś na diagramie sekwencji na kolejność przesyłania komunikatów przy realizacji scenariusza przypadku użycia.
- Diagram współpracy może dotyczyć jednego lub wielu scenariuszy jednego przypadku użycia.
- Można agregować diagramy współpracy dla wielu przypadków użycia.
- Diagram współpracy dla wszystkich przypadków użycia pozwala oceniać stopień zależności pomiędzy poszczególnymi klasami obiektów.

<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Przykładowy diagram współpracy



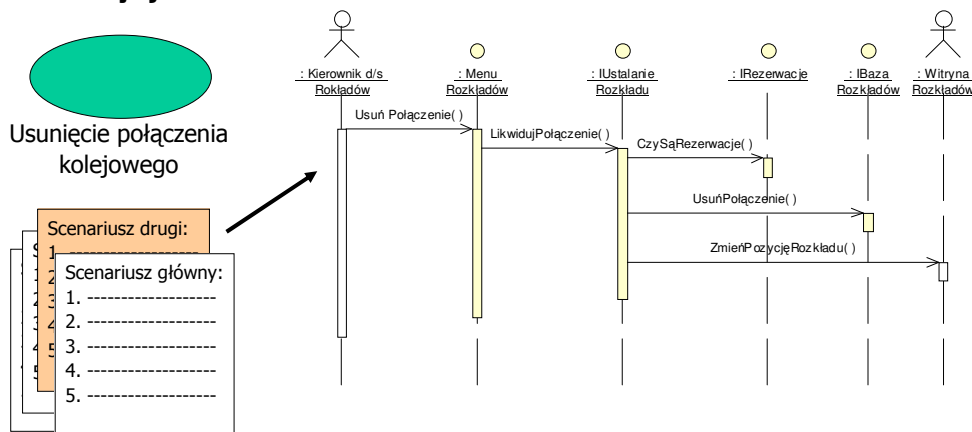
- Diagram współpracy odpowiadający informacyjnie diagramowi sekwencji

<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Diagramy interakcji dla przypadków użycia

- Diagramy interakcji na poziomie architektury: aktorzy i interfejsy.

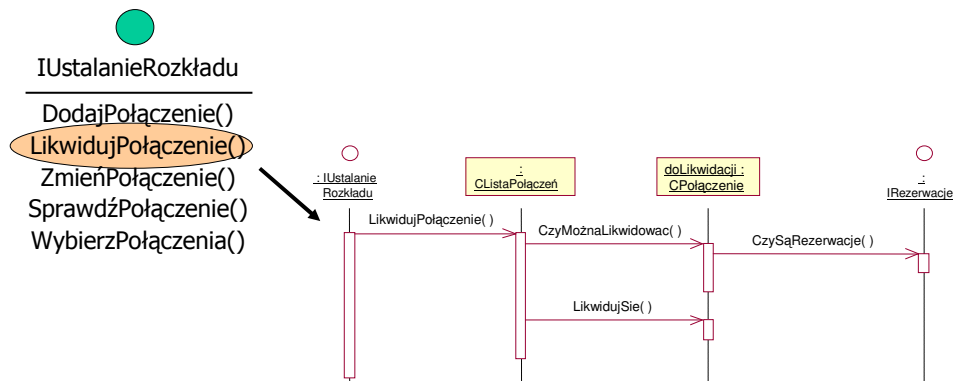


<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Diagramy interakcji dla usług

- Diagramy interakcji na poziomie projektu podsystemu (komponentu): interfejsy i obiekty klas.



<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Weryfikacja dynamiki - granie ról

- Bardzo dobrą metodą konstrukcji diagramów interakcji są sesje „grania ról”. Podczas takiej sesji korzystamy z interfejsów lub klas utworzonych w modelu statycznym.
- Każdy uczestnik otrzymuje po kilka klas i podczas sesji gra role obiektów otrzymanych klas.
- W trakcie sesji odczytywane są scenariusze przypadków użycia. Po odczytaniu każdego zdania, uczestnik, którego klasa jest odpowiedzialna za odczytane zdarzenie informuje o tym fakcie.
- W razie wątpliwości, która klasa powinna być za co odpowiedzialna, dyskutowany jest model statyczny. Podczas przechodzenia przez kolejne scenariusze uzupełniany i zmieniany jest zatem zakres odpowiedzialności klas i komponentów.

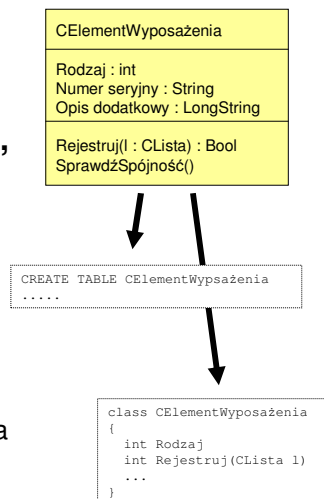
<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Generacja kodu z modeli w UML

- **Większość narzędzi CASE umożliwia generację kodu dla różnych języków programowania, baz danych czy technologii komponentowych (np. Java, C++, Visual Basic, Oracle 8, CORBA, COM, EJB).**

- Generacja dla języków obiektowych: jednej klasie w modelu klas odpowiada jedna klasa w języku obiektowym.
- Generacja dla bazy danych: zazwyczaj jednej klasie odpowiada jedna tabela.
- Generacja dla technologii komponentowych: dla klas interfejsowych tworzone są opisy w językach opisu interfejsów.



<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Klasy, obiekty, komunikaty - jak wyprodukować kod?

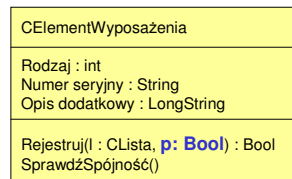
- **Na projekt systemu składają się model klas i model interakcji (diagramy sekwencji). Wszystkie elementy tych modeli mogą być bezpośrednio lub pośrednio przełożone na kod.**
- **Klasa projektowa ==> szkielet kodu klasy**
 - Wszystkie atrybuty, operacje, sygnatury operacji są bezpośrednio realizowane w kodzie.
- **Obiekt ==> zmienna lokalna operacji lub atrybut klasy**
 - Każdy nazwany obiekt na diagramie sekwencji może być zadeklarowany w kodzie operacji jako zmienna lokalna.
- **Komunikat ==> zawsze odpowiada operacji klasy obiektu**
 - Wykonanie operacji w wyniku przesłania komunikatu znajduje odzwierciedlenie w kodzie tej operacji dla odpowiedniej klasy.

<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Utrzymywanie spójności modelu i kodu

- **Dobra zasada: wszelkie modyfikacje atrybutów, nazw i sygnatur metod dokonywane są tylko w modelu graficznym, nie w kodzie.**
- **Po zmianie jednego z elementów opisu klasy generowany jest kod. Współczesne narzędzia CASE umożliwiają zachowanie kodu pisanego w ramach środowiska deweloperskiego.**
- **Praca programisty-projektanta: jednoczesna budowa modelu klas, modelu interakcji i pisanie kodu.**



```

class CElementWyposażenia
{
    int Rodzaj
    int Rejestruj(CLista l, int p)
}

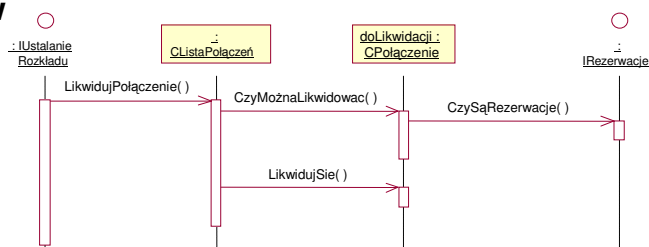
int CElementWyposażenia::
    Rejestruj(CLista l, int p)
{
    // początek kodu metody
    CLista n; CZbior z;
    n.inicjuj(); z.inicjuj();
    if (p) n.kopiuj(l);
    else z.zastap(l);
    // koniec kodu metody
}
  
```

<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiałek
ZETiIS, PW

Realizacja usług i metod klas

- **Zestaw diagramów interakcji jest podstawą do określenia struktury komunikacji (wywołań metod) między obiektami. Struktura ta powinna być odzwierciedlona w kodzie metod. Zadanie to należy do programisty.**



```

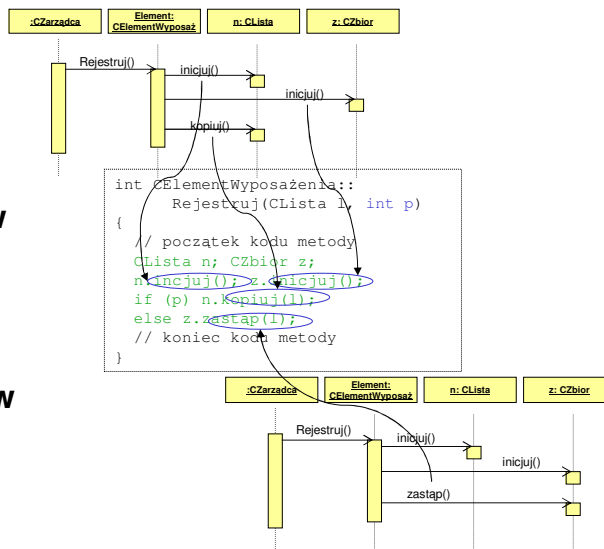
void CListaPołączeń::LikwidujPolaczenie() {
    CPolaczenie doLikwidacji;
    // jakiś kod
    doLikwidacji.CzyMożnaLikwidowac();
    // jakiś inny kod
    doLikwidacji.LikwidujSie();
    // jeszcze inny kod
}
  
```

<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiełek
ZETiIS, PW

Jak przetłumaczyć diagramy sekwencji?

- Diagramy sekwencji ułatwiają tworzenie kodu operacji. Kod operacji stanowi podsumowanie treści wszystkich diagramów sekwencji, na których występuje odpowiedni komunikat.
- Rolą programisty jest umiejętnie połączenie w kodzie wszystkich możliwych wywołań operacji.

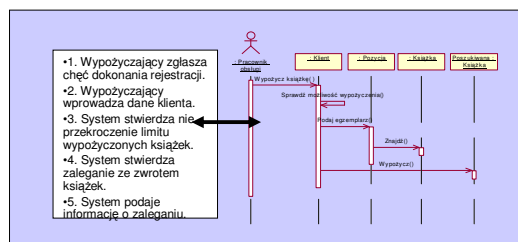
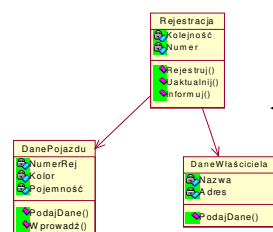
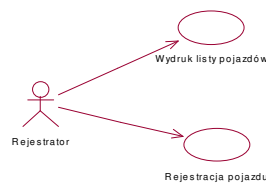


<http://www.iem.pw.edu.pl/~smialek>

Michał Śmiełek
ZETiIS, PW

Podsumowanie – od przypadków użycia do klas

- Przypadki użycia =>
- => scenariusze =>
- => diagramy interakcji =>
- => diagramy klas



<http://www.iem.pw.edu.pl/~smialek>