

# INŻYNIERIA OPROGRAMOWANIA

MGR. ZBIGNIEW KASPRZYK LAB.

prof. KRZYSZTOF SACHA {poki 42 14<sup>25</sup>-15<sup>00</sup>

SALA 210 11<sup>30</sup>-14<sup>25</sup>

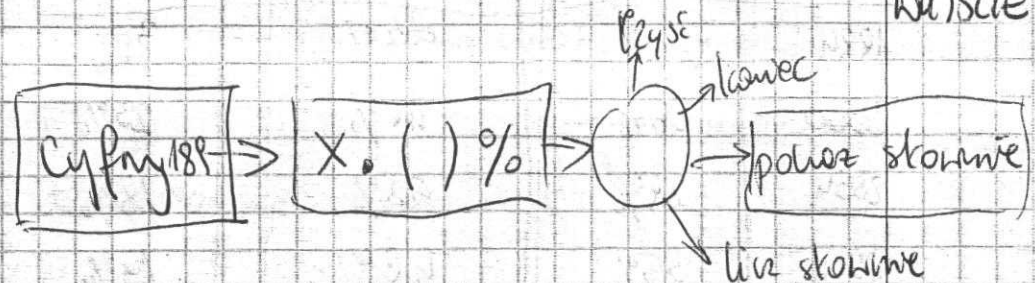
T

## WYKŁAD 1 LAB

2009/02/21

## UML - UNIFIED MODELING LANGUAGE

WYŚCIE



## WYKŁAD 1

2009/02/22

Inżynieria oprogramowania definiuje się metodami  
wytworzenia oprogramowania na szkieletach programów

dużo więcej wykładów 15%

projektowanie programów 25%

programowanie 20%

weryfikacja i ocena 40%

wzrost, i konsolidacja

- oprogramowanie
- celowy projekt programistyczny
- metody
- cel inżynierii oprogramowania

## Stan wiedzy (Chaos report, Standish Group)

Rok	Success	Preliminary	Porazka
1994	16%	53%	31%
2000	28%	44%	28%
2004	28%	53%	19%
2006	35%	46%	19%

## PROCESY PROJEKTOWE

- systemy zorientowane
- produkty otwartego rynku
- Cel: wytworzenie systemu informatycznego
- przygotowanie decyzji
  - studium wykonalności
  - specyfikacja wymagań
  - plan
- Wytworzenie systemu  $\longleftrightarrow$  podpisywany kontrakt
  - projektowanie systemu
  - opracowanie oprogramowania
  - przygotowanie infrastruktury przez klienta
  - testowanie  $\longleftrightarrow$  odbiór

## Wdrożenie

- przeniesienie danych
- szkolenie użytkowników
- runienie pojem biznesowego
- $\longleftrightarrow$  odbiór ostateczny
- ~~testowanie~~ i konserwacja
  - używanie systemu
  - konserwacja
  - modyfikacje i ewolucja

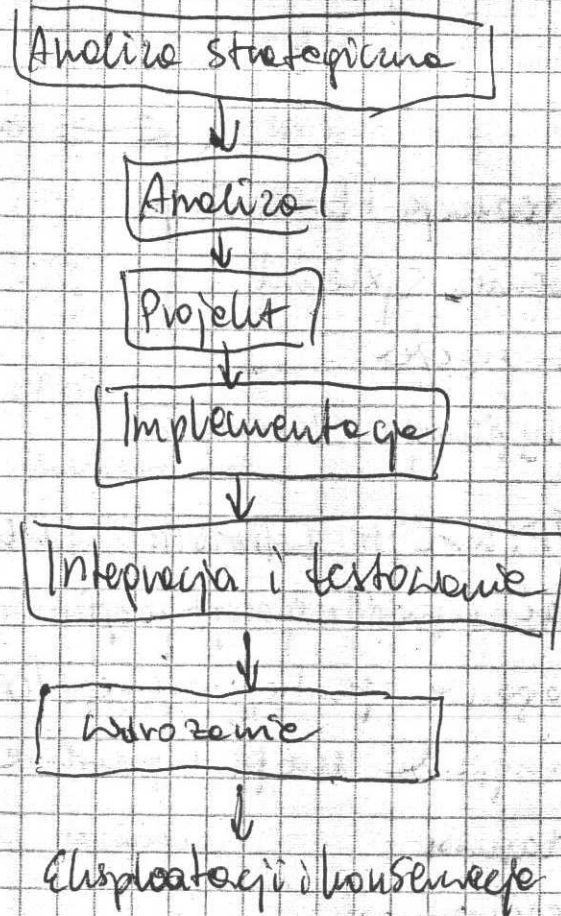
## SPECYFIKACJA WYMAGAŃ (dokument techniczny)

- postać i treść specyfikacji
- wymagania funkcjonalne (co system ma robić)
- wymagania niefunkcjonalne
  - wydajność
  - niezawodność
  - bezpieczeństwo
  - zgodność
- cechy specyfikacji
- zrealizowane wymagania



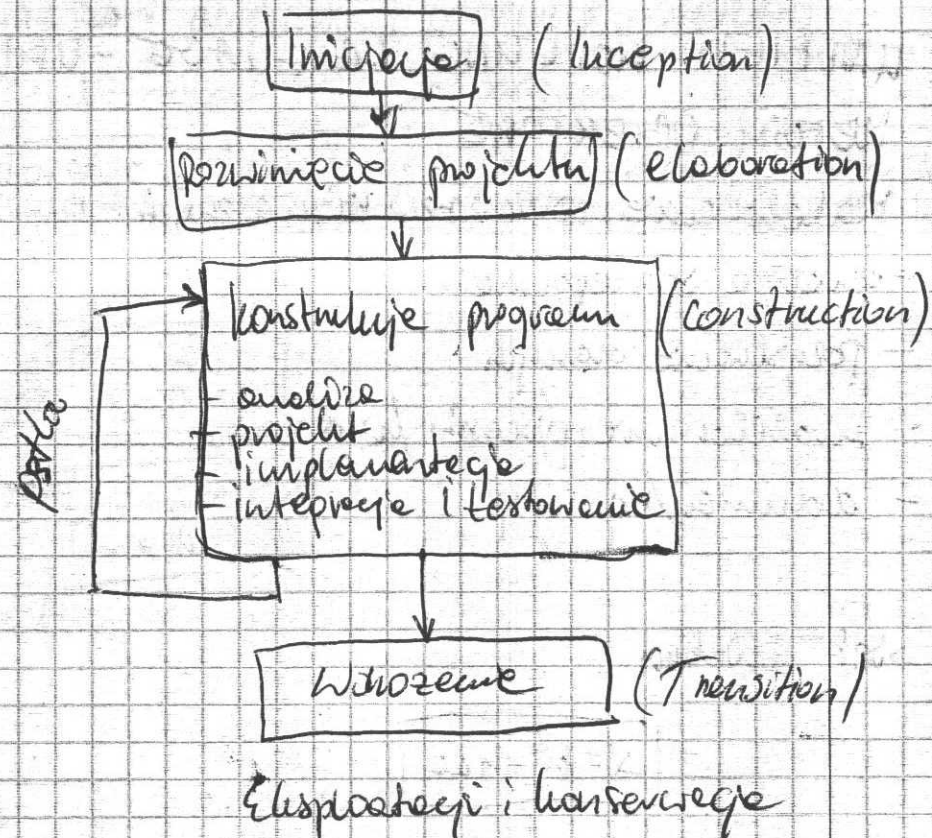
# KASKADOWY MODEL WYTWARZANIA OPROGRAMOWANIA (WATERFALL MODEL)

nie ma  
wrócić do  
wspierających  
elementów projektu



Iteracyjny model wytwarzania  
oprogramowania (Iterative model)

Proces RUP (Rational Unified Process)



## METODY I NARZĘDZIA

- Metody i modele
  - strukturalne
  - obiektowe, język UML
  - metodyki lekkie
- Narzędzia
  - systemy upper CASE
  - system lower CASE
  - środowiska uruchomieniowe testowania

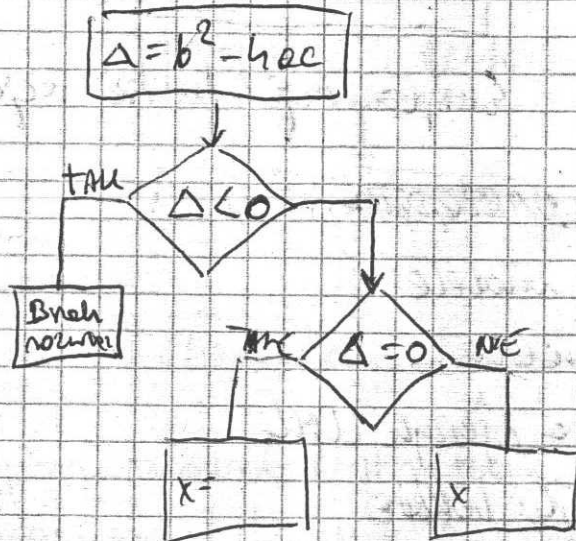
# UNIFIED MODELING LANGUAGE - UML

## METODY OBIEKTOWE

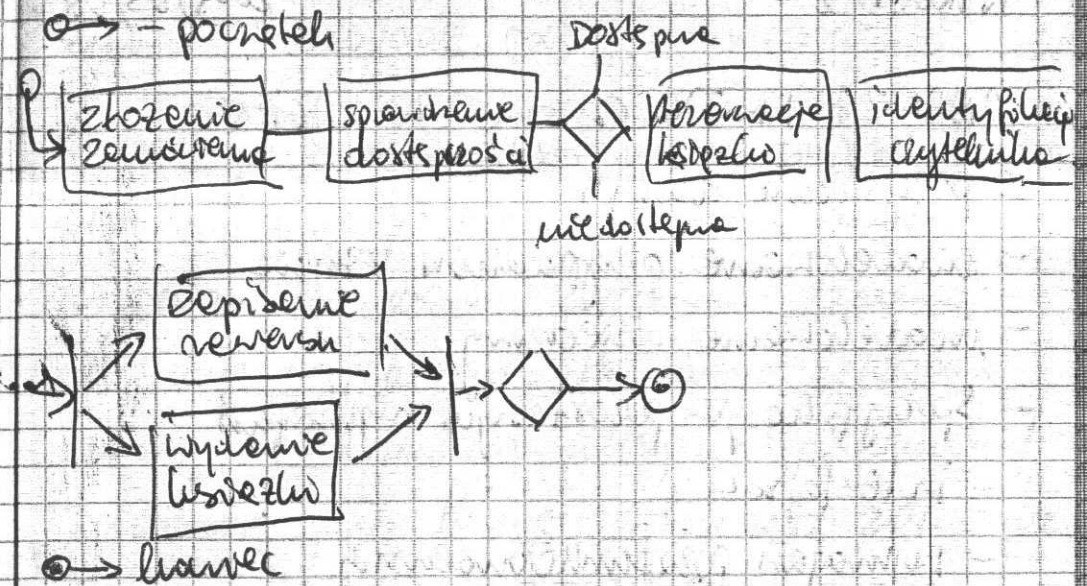
Modelowanie procesów biznesowych

- analiza strategiczna
- przetwarzanie danych
- modelowanie procesów biznesowych
- dokumenty wizji

Stać dwiekani



## DIAGRAM AKTYWNOŚCI (ACTIVITY DIAGRAM)



- zdawanie

- komentarz

- błąd, pomyłka, przerwanie

- obiekt

(NOTE)



WYKKAAD 2

2008/03/08

# ANALIZA I MODELOWANIE WYMAGAN

- porównanie danych
- modelowanie przypadków użycia
- modelowanie obwodów
- Specyfikacja prostych wymagań
- interfejsu
- wymagań poro funkcjonalnych
- drzaskin po ahroni
- p reguk biznesowych

USE CASE  $\rightarrow$  modelowanie  
wymagania

# PROBLAMY WZGLĘD

Zachowanie ubezpieczeniowe jest wykonywane  
na zlecenie ubezpieczyciela, któremu przysług  
mu korzyść w postaci ochrony przed  
odpowiedzialnością cywilną lub rozbrojeniem  
własnego samochodu. Wykonawcą zadania  
obejmuje wykonanie planu na formularzu

dokrećenie kopii dokumentów semiotyki, określenie  
składów i odebranie polityki ubezpieczeniowej

Likwidacja służby następuje na wniosek klienta  
litewum popysi komisji w postaci zwrotu  
pauzowanych kosztów. Wyliczenie zadane  
obejmuje wypełnienie formularza zrzeczenia  
służby, przedstawienie petycji ubezpieczeniowej  
ocenionej przez firmę z siedzibą zrzeczenia  
i rozstrzygnięcie i wartości służby oraz  
wyliczenie premii na konto klienta.

Seemannsz lilimilegi! Silioty

1. Przyjmujący rejestruje zgłoszenie w Systemie Zgłoszenie obejmuje: dane zgłaszającego, data wypadku i data zgłoszenia
2. System tworzy spisanie historii śladu i nadaje jej unikalny numer identyfikacyjny
3. Przyjmujący sprawczo dane śladu, obejmujące opis śladu i <sup>opis</sup> ~~innych~~ uszkodzeń oraz podpisuje dokument zgłoszenia



4. Diagnostyka dostępności zasobów
5. System przypisuje licencje do zasobów

Relacje łączące przypadki użycia

1. Generalizacja
2. Rozszerzenie
3. Zawieszenie

<<include>> włączyć  
<<extend>>

Przykład

System biblioteczny

Wymagania

1. System ma umożliwić przeglądanie katalogu oraz rezerwowanie książek dostępnych i niedostępnych przez internet
2. System ma rejestrować operacje wypożyczenia i zwrotu książek
3. System ma wspierać bieżące politykę biblioteki w zakresie prawa do wypożyczenia i zwrotu

• Rezerwacja exemplara

Aktory: czytelnik "Scenariusz główny"

1. czytelnik przegląda katalog biblioteki i wybierze wybrane pozycje
2. system sprawdza dostępność
3. system sprawdza dostępność czytelnika
4. System zapisuje rezerwacje i potwierdza wykonanie operacji

"Scenariusz alternatywny - braku wolnej kopii"

- 1-2 jak w scenariuszu głównym
3. Czytelnik wybiera opcję oczekiwania
4. System sprawdza dostępność czytelnika
5. System zapisuje oczekiwanie i potwierdza wykonanie operacji

• Wypożyczenie exemplara

Aktory: czytelnik, bibliotekarz

Scenariusz główny

1. bibliotekarz identyfikuje czytelnika w systemie
2. System sprawdza stan konta czytelnika
3. Bibliotekarz identyfikuje wypożyczenie z systemu



4. System zapisuje rekord wypożyczenia i potwierdza wykonanie operacji

5. Bibliotekarz preliminarz egzemplarz cyfrowy (ani)

Scenariusz alternatywny 1 - preliminarz limit wypożyczeń

1-2 jch w scenariuszu głównym

3. System porównanie o preliminarzu limitu wypożyczeń

Scenariusz alternatywny 2 - nieopłacone konto za preliminarz termin

1-2 jch w scenariuszu głównym

3. System powiadomienie o nieopłaconym koncie i wysłanie jej listy do opłacenia

4. Cyfelnik opłaca konto

• Zwrocone egzemplarze

Scen. główny

1. Bibliotekarz identyfikuje zwrocony egzemplarz w systemie

2. System zwraca rekord wypożyczenia i sprawdza termin zwrotu

3. System sprawdza kolejkę i zwraca odczytujący cyfelnik

4. System zwraca zwrocony egzemplarz dla odczytującego

5. System usuwa rekord i potwierdza zwrot egzemplarza

- preliminarz termin zwrotu

• Przeprowadzenie kontu

1. System sprawdza dostępność cyfelnika

2. System wysłanie stanu konta cyfelnika "liste wypożyczeń, liste rezerwacji, liste opłat i uchylnych kont"

3. Cyfelnik usuwa niepotrzebne rezerwy i odczytuje

## REGUŁY BIZNESOWE

RB1. dane bibliograficzne powinny obejmować:  
dokładny frequent tytułu lub numeru  
autora albo Nr ISBN, miejsce wydania  
pisać rok wydania

RB2. porządek danych się nie ma być konieczny  
podkreślenie, wyrażenie ma ~~możliwość~~ zawierać  
ogólna - ma mieć, określenie - precyzacja  
ma mieć

RB3, RB4, RB5

## DOCUMENTATION MODEL

- nazwa i identyfikacja modelu
- krótki opis problemu
- powstanie
- cel
- zakres - stan początkowy i końcowy
- scenariusz podstawowy i scenariusz alternatywny
- reguły biznesowe
- dokumenty przetwarzane i wytworzone
- uwagi i otwarte pytania (bardzo ważny punkt)

- diagram alityczności
- prototyp interfejsu użytkownika

## ZASTOSOWANIE MODELA PRZYPADKÓW UŻYTKA

- opis wymagań funkcjonalnych
- planowanie fazy konstrukcji
- rozumienie zachowanie przy projektowaniu aplikacji
- scenariusze testowania
- scenariusze podsumowania

## WILKARD 3

2009/03/22

## MODELOWANIE DŁUGOTERMINOWYCH PROBLEMÓW

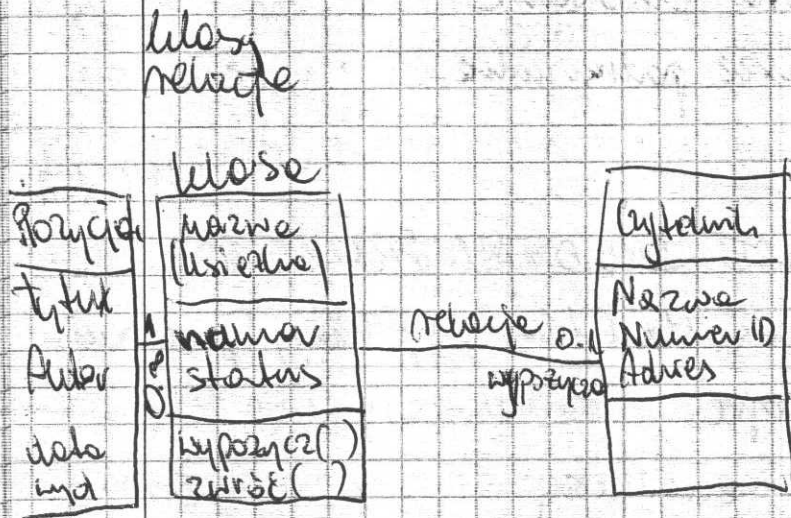
- wyodrębnienie danych określonych problemem  
analiza potrzeb  
analiza scenariuszy  
ocena skutków
- porównanie danych  
klasyfikacja danych  
identyfikacja zmian w klasie  
dokumentowanie



- określenie typów zachowań
- identyfikacja stanu obiektów
- identyfikacja zmian
- dokumentowanie

## DIAGRAM KLAS

- Model statycznej struktury problemu
- model



## PERSPEKTYWA WIZJENIA MODELU

- model pojęciowy
- model analizy (połączenie klas i atrybuty, relacje klas)
- bieżące sregulacje implementacyjnych

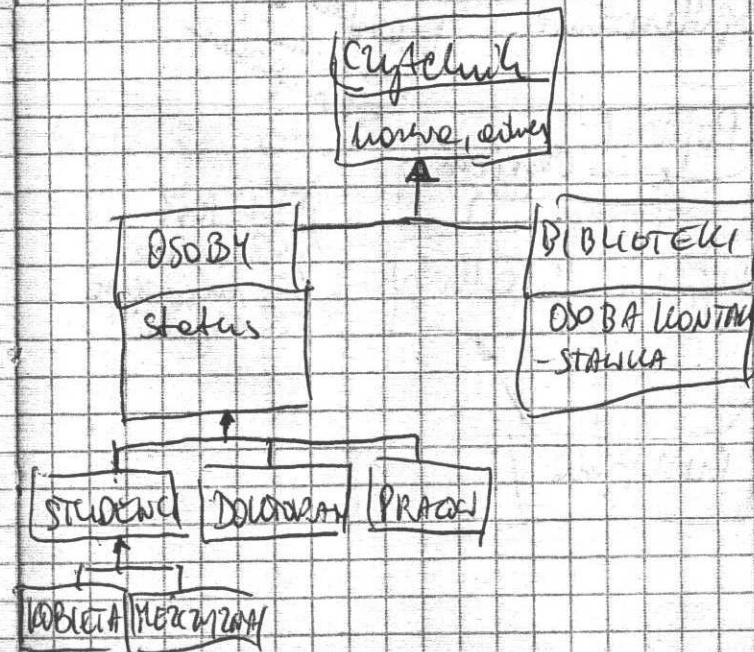
- Model projektowy (programistyczny)
- model projektu i implementacji
- określone struktury danych (pole)

## RELACJA WLOGICZENIA - klasyfikacja

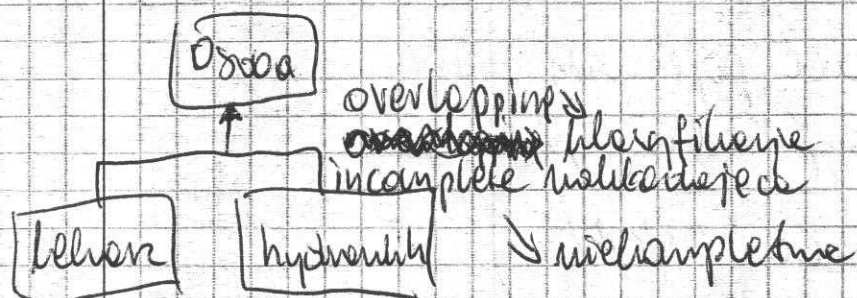
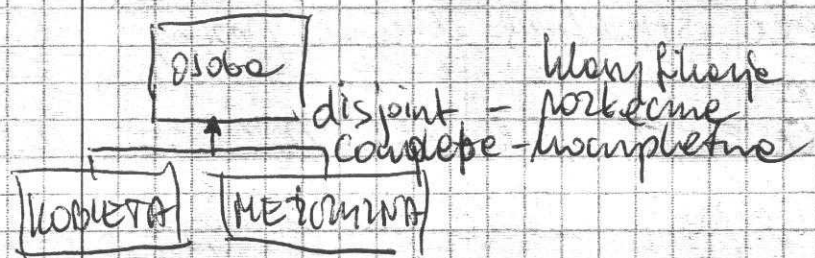
- klasyfikacja, modele na różnych poziomach abstrakcji

- model pojęciowy
- klasyfikacja bytów

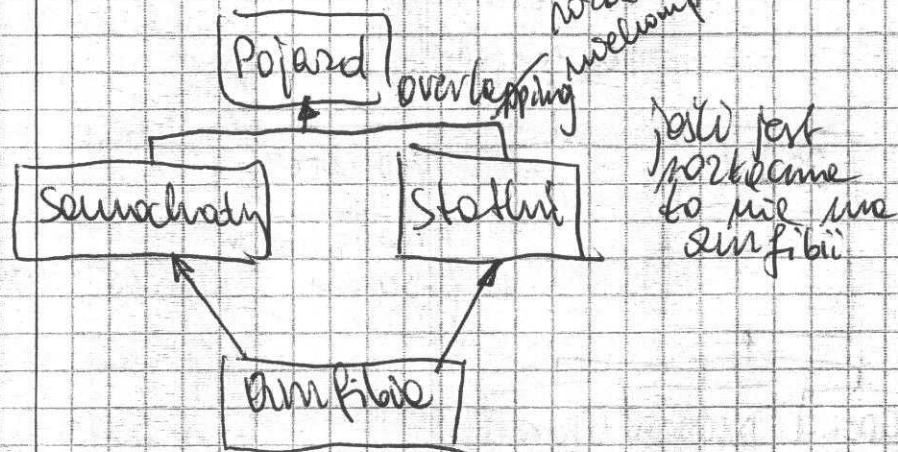
- model projektowy
- hierarchiczne interfejsy i drzewiaste



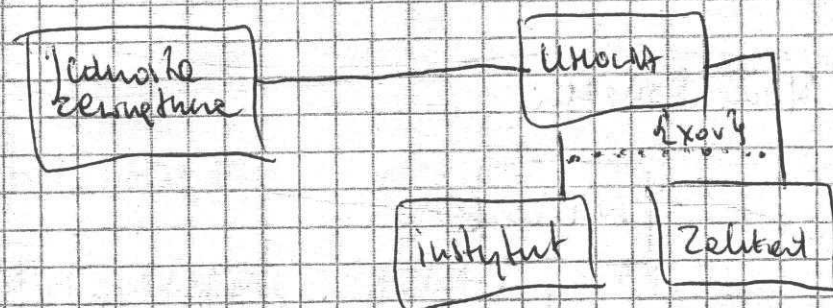
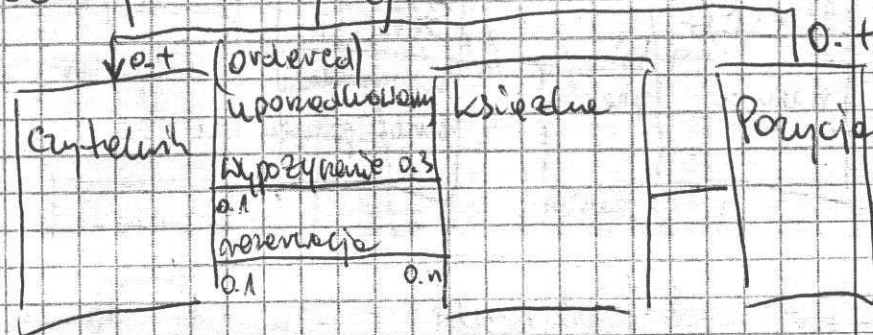
## OGRA NICZENA KLAS YFIKAC YJ



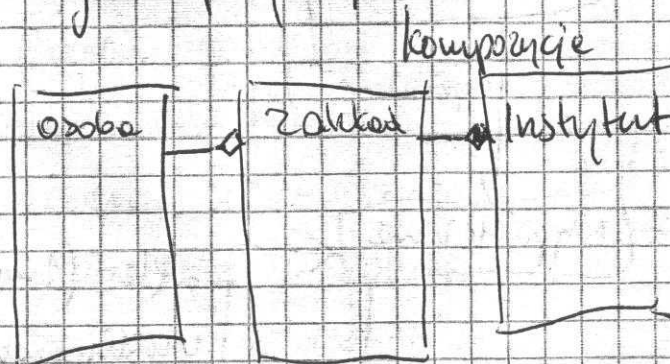
## Relacje wypełniania



## Relacje asocjacji

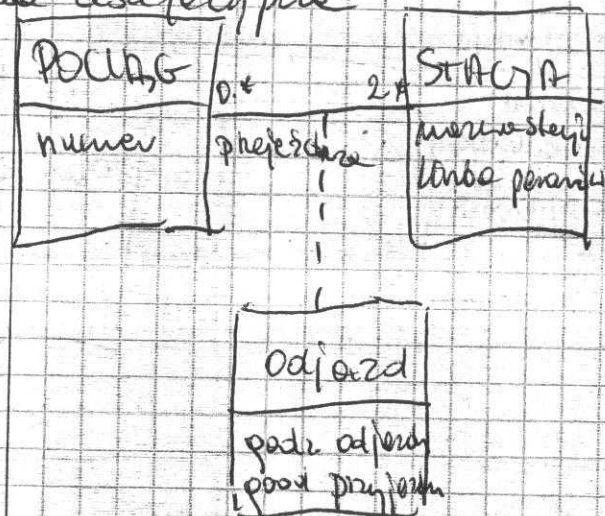


## Relacje operacji

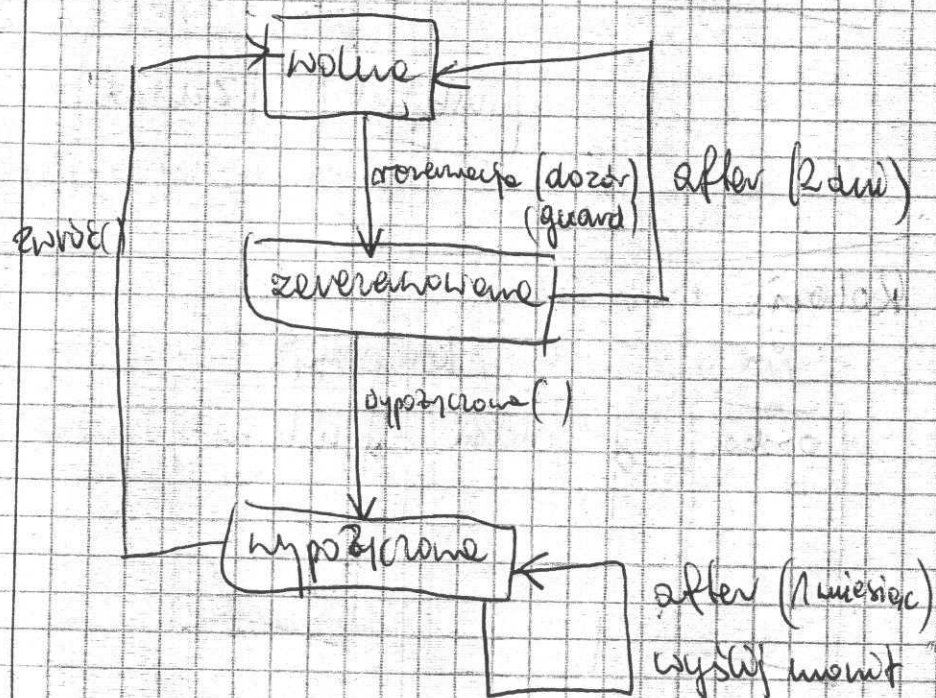




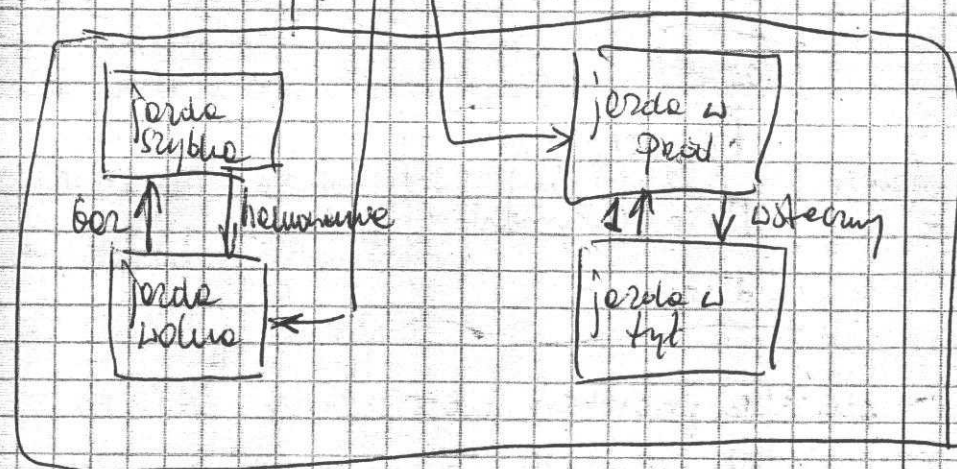
## Klasa asystenta



## Stan Kierownika



Stany  
Rozmowa



## WUKRAD 4

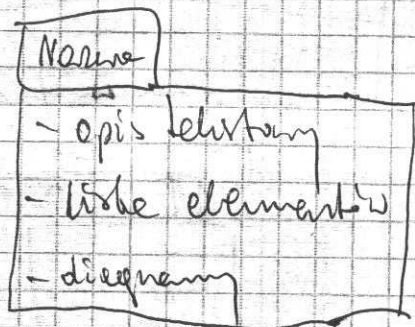
2008/04/05

## MODELOWANIE ARCHITEKTURY

- Modelowanie struktury oprogramowania
  - diagram komponentów
  - diagram pakietów
- Modelowanie fizycznej struktury systemu
  - diagram rozmieszczenia
- Modelowanie współpracy
  - diagram sekwencji
  - diagram komunikacji

## DIAGRAM PAKIETÓW

Pakiet - zbiór powiązanych elementów



- opis pakietu

... pakiet - zawierający ...

... elementy

- opis relacji między pakietami

Diagram sekwencji

Diagram komunikacji

Zastosowanie modeli

- Modelowanie architektury w formie schematów

• logiczne struktury danych (diagramy pakietów)

• fizyczne struktury systemów (diagramy rozmieszczenia)

• sposób drukowania (diagramy sekwencji)

- Modelowanie programów w formie konstrukcji

• Struktura programu

• Zmienne

• Dokumentacja

Projektowanie architektury oprogramowania

Analiza uciążliwości (Robustness Analysis)

- obciążenie graniczne

- obciążenie awaryjne

- obciążenie sterujące

Obciążenie graniczne (przebiegiem pracy)

Obciążenie sterujące

Obciążenie awaryjne

Projektowanie programów

Warstwa danych

Warstwa logiki biznesowej

(procedury biznesowe użytkownika)



# Reprezentacja wyposazenia (Scenariusz projektowy)

1. Biblioteczka wyboru opisu wyposazenia
2. Biblioteczka wprowadze kod literalamy karty bibliotecznej cyfelnika
3. System sprawozdanie saldo opkt cyfelnika
4. System zwojdzije rezerwacje cyfelnika i wyswietlenie ich opis
5. Biblioteczka wprowadze kod literalamy kodu
6. System dolina terminu zwrotu
7. System zapisuje rewers i bawe danych
8. System potwierdzenie wyposazenie

Wskazano

2009/04/26

Nastepne lapiki bismorowej

- I wariant - procedury transzacji uzytkownikow
- wypozycz() → kontroler wyposazenia
- podaj rezerwacje()
- cyfel kod kroskow() → cyfelnik
- sprawdz saldo (cyfelnik) → kontroler wyposazenia
- dolice okres (cyfelnik, wolunin) - RB2

- zwojdz Cyfelnika (idc)
- zwojdz rezerwacje (idc)
- zwojdz wolunin (idw)
- zwojdz rewers (rewers)

→ bawe danych

## Program (kod JAVA)

```

public class kontroler wyposazenia {
    private BaweDanych b = new BaweDanych();
    private Rezerwacja rezerwacja;
    private Cyfelnik cyfelnik;
    public Rezerwacja Rezerwacja podaj Rezerwacje (long idc) {
        cyfelnik = b.zwojdz Cyfelnika (idc);
        if (sprawdz saldo (cyfelnik) != OK)
            return null;
        rezerwacja = b.zwojdz Rezerwacje (idc);
        return rezerwacja;
    }
}
    
```

```
public int wypozyca (long int idk)
```

```
{
```

```
    long int idp;
```

```
    int okresWypozyczenia;
```

```
    idp = b. znajdzWolumin (idk)
```

```
    if (rezerwacja.idp != idp)
```

```
        return ERROR
```

```
    okres = obliczOkres (rezerwacja, kategoria,  
                        cytelnik, status)
```

```
    b. aktualizujRezerw (rezerwacja, idC, idP,  
                        idk, okres);
```

```
    return OK;
```

```
}
```

```
private int obliczOkres (int kategoria, int  
                        status)
```

```
{
```

```
    Switch (kategoria) {
```

```
        case patrewnik : return 100;
```

```
        case ogólny : return 30;
```

```
    default Switch (status)
```

```
{
```

```
    case przeornik : return 30;
```

```
    case doktorant : return 1;
```

```
    default : return 0;
```

```
}
```

```
}
```

```
}
```

Warstwa logiki biznesowej

II Warstwa

Program UWD

// Program kłuszenia obywateli, oparte wypożyczenia  
// użytkownika systemu biurowego

```
private PortelDanych p = new PortelDanych();
```

```
private Cyfelnik c = new Cyfelnik();
```

```
private Rezerw rezerw;
```

```
private Cytelnik cytelnik;
```

```
private Wolumin wolumin;
```

```
idC = c. cyfelKod();
```



```

czytelnik = p. znajdzCzytelnika (idC);
if (czytelnik.sprawdzSeldo() == OK) {
    newers = p. znajdzRenewacje (idC);
    newers.pozycja, wyswietl (okno);
    idW = c.czytajKod();
    wolucmin = p. znajdzWolucmin (idW);
    newers.wypozycz (wolucmin);
    p.sztuklizuj (newers);
}
...

```

```

Public class Renew {
    public int wypozycz (Wolucmin Wolucmin)
    {
        if (pozycja != wolucmin.pozycja)
            return ERROR;
        this.wolucmin = wolucmin;
        oknoWypozyczenie = pozycja.obliczOkno
        (czytelnik, status);
        return OK;
    }
}

```

```

Public class Podswami extends Pozycja
{
    public int obliczOkno (int status)
    {
        return 100;
    }
}

```

```

public class Opreniwanie extends Pozycja
{
    public int obliczOkno (int status)
    {
        switch (status)
        {
            case Pracownik Pracownik = return 30;
            case Student Student = return 7;
            default: return 0;
        }
    }
}

```

```

public class Stanodruk extends Pozycja
{
    public int obliczOkno (int status)
    {
        return 1;
    }
}

```

WYKŁAD

2008/05/17

## Formy konstruowania

- model implementacyjny (doprecz. <sup>składowczy</sup> list)
- model danych (doprecz. <sup>składowczy</sup> list, listy)
- pliki źródłowe / konfiguracyjne
- scenariusze przypadków testowych oraz raporty testowania
- dokumentacja, administracja

## Faza Prechodzenia

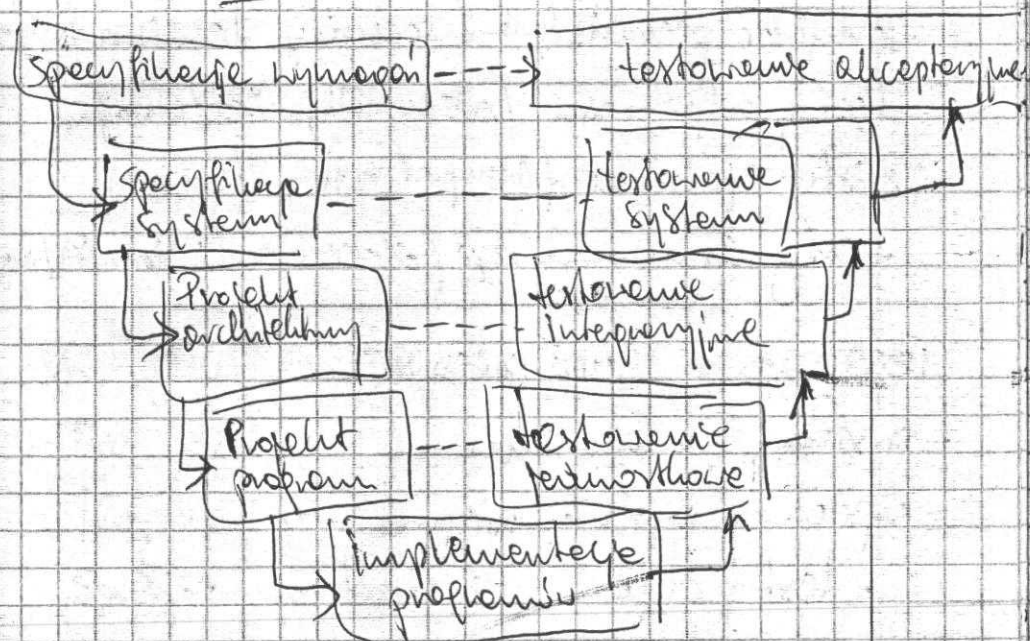
- zdefiniowanie oprogramowania (baza testy lub testowanie akceptacyjne)
- gotowy produkt
- docelowy system
- szkodliwe użytkownika

## Testowanie oprogramowania

- poprawność programu
- weryfikacja i zdefiniowanie
- testowanie
- tryb testowania (ciężki kontynuacyjny test)  
dane wej. - sposób postępowania

- projektowanie testów
- błędy i defekty programu
- testy regresji

## "POZIOMY TESTOWANIE"



## TESTOWANIE JEDNOSTKOWE I INTEGRACYJNE

- sterowanie testowe
- narzędzia
- strategie integracji
  - parametryczne
  - warstwowe
  - 2-stopniowe



## TESTOWANIE SYSTEMOWE

- testy funkcjonalne (functional testing)
- testy wydajnościowe (performance testing)
- testy przeciążeniowe (stress testing)
- testy bezpieczeństwa (security testing)
- testy odporności (recovery testing)
- testy zgodności (compatibility testing)
- testy dokumentacji (documentation testing)

## TESTOWANIE AKCEPTACYJNE

- systemy zamknięte
- systemy otwarte
  - testy alfa - testowanie u dostawcy
  - testy beta - testowanie u odbiorcy (u zmiennika za informacje o błędach)
- testy funkcjonalne
- testy operacyjne
- testy niefunkcjonalne

## ORGANIZACJA PROCESU TESTOWANIA

### Plan testów

- zakres testów

## Strategia testowania i raportowanie błędów

- metadane testowania
- kryteria zakończenia
- kryteria zakończenia testów

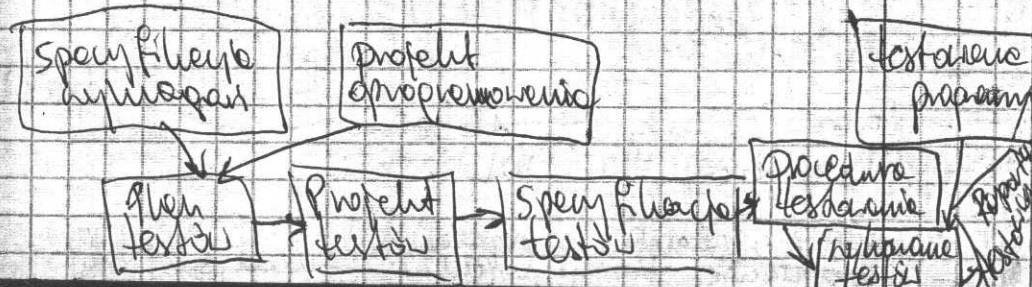
### Zarządy

- śledztwo testowe
- ludzie i odpowiedzialność

### Harmonogram

## PRZYGOTOWANIE TESTÓW

- Plan testów
- (Projekt testów)
- Specyfikacja testów
  - przypadki testowe
  - Wynagrodzenia i opiewanie
  - scenariusze przypadków testowych
- Procedura testowania



## METRYKI TESTOWANIA

Metryki pokrycia kodu

- pokrycie bloków
- pokrycie decyzji
- pokrycie składek

testowanie czarnego skrzynki (black-box)

testowanie szklanego skrzynki (white-box)

Blok - linie przedstawionych bloków  
do linii wszystkich bloków

## METRYKI POKRYCIA WYMAGAŃ

- pokrycie wymagań
- pokrycie błędów
  - pokrycie przypadków biznesowych
  - pokrycie przypadków systemowych
  - pokrycie scenariuszy

## INNE METRYKI

- linie wykrytych defektów
- linie defektów komponentów
- częstość defektów
- procent defektów poprawianych
- szacowanie liczby defektów (fault seeding)

## WNIOSKI

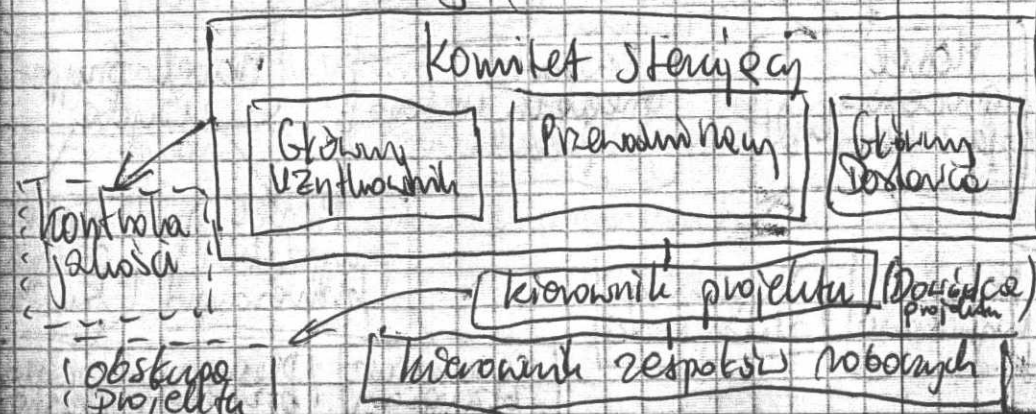
31/05/2008

Zarządzanie projektem

Zakres:

- Struktura organizacyjna
- Planowanie projektu
  - określenie zakresu pracy
  - tworzenie harmonogramu
- Plan projektu
- Planowanie kosztów
  - szacowanie
  - tworzenie budżetu
- Nadzorowanie biegu projektu
- Zarządzanie ryzykiem

## Schemat struktury (PRINCE 2)



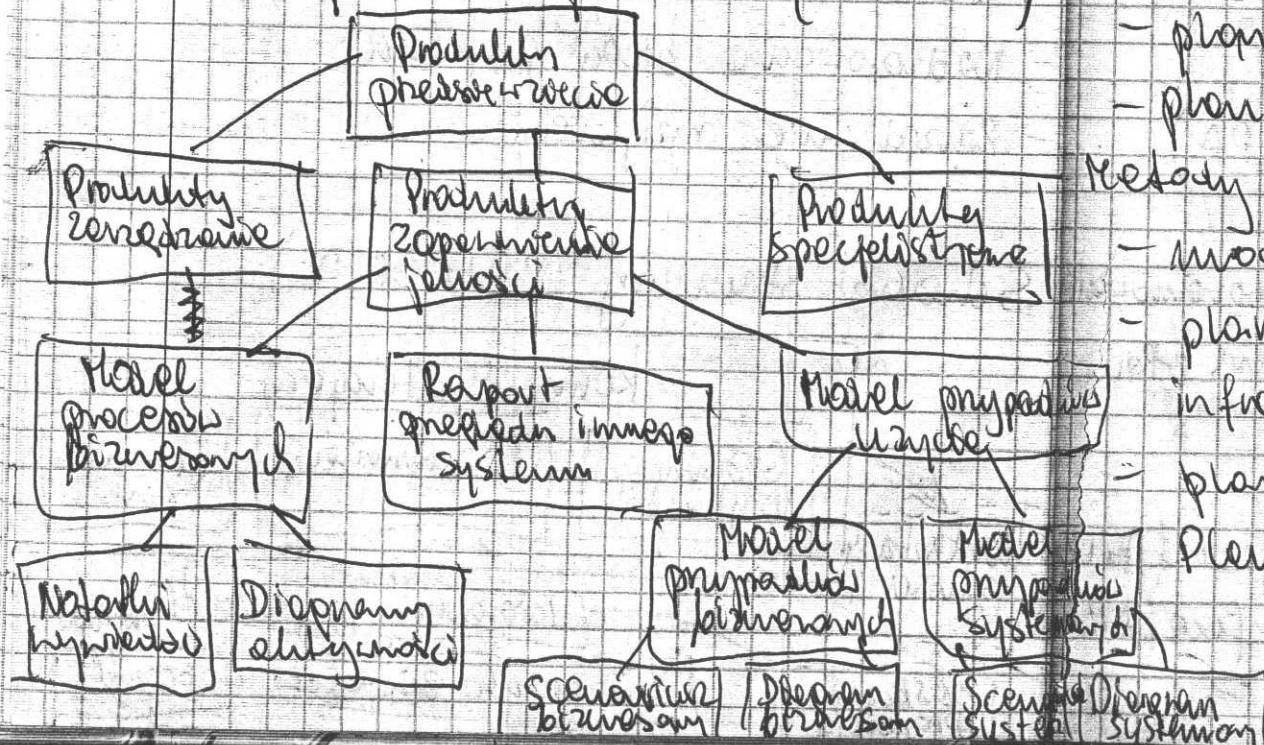


# Phylosofia projektu (PLANNING PROJECT)

## Modelowanie przypadków użycia

- 1) przeprowadzenie wywiadu (3 dni)
- 2) przejrzenie systemu innej biblioteki (2 dni)
- 3) modelowanie procesu biznesowego (2 dni)
- 4) tworzenie modelu biznesowego (5 dni)
- 5) tworzenie modelu systemowego (12 dni)

## Struktura podziału produktu (PRINCE2)



## Metoda Szwedzkiej inteligencji

### PLAN PROJECT

#### Wstęp

#### Organizacja projektu

- opis otoczenia
- opis struktury organizacyjnej
- podział ról i odpowiedzialności

#### Plan zarządzania projektem

- plan rozkładu projektu
- plan pracy
- plan monitoringu projektu
- plan zakończenia projektu

#### Metody i techniki realizacji projektu

- model procesu wytwórczego
- plan pozyskania i utrzymania infrastruktury
- plan zrealizowania produktów (kostowniki, jakość)

#### Plan procesów wspierających

## Planowanie kosztów

- szacowanie kosztów
- ukazanie budżetu

## Heurystyczne metody szacowania

- metoda analogii
  - średnia i zakres zastosowania
  - rozmiar
  - metody i narzędzia
  - wymagania specjalne
- metoda oceny podobieństwa pracy
- metoda oceny podobieństwa pracy

## Metoda punktów funkcyjnych (FPA) (Function Point Analysis)

- linowa ocena złożoności oprogramowania
- International Function Point Users Group

## Ocena złożoności zbiorów

- zbiory wewnętrzne (Internal Logical File ILF)
  - zbiory zewnętrzne (External Interface File EIF)
- REI  
DET

## Ocena złożoności przetwarzania

- wejście zewnętrzne (External Input - EI)
  - wyjście zewnętrzne (External Output EO)
  - zapytanie (External Inquiry EQ)
- ETR - linowa różnorodność zbiorów ILF, EIF  
DET - linowa ilość rekordów

## Niezależne dochodzących czynników kosztu (N)

- złożona komunikacja zewnętrzna
- przetwarzanie na bieżąco
- złożony interfejs użytkownika
- złożone algorytmy przetwarzania
- wysokie wymagania wydajnościowe
- uśrednionym obciążeniem
- przetwarzanie rozproszone
- ograniczenie zasobowe

$$VAF = (0,01 \times TDI) / 0,65$$

~~AFR = UFR \* VAF~~  
 $AFP = UFP * VAF$

$$SLOC = AFP * LM$$



Source  
Lines  
of  
Code } SLOC

## Model COCOMO (Constructive Cost Model)

$$\text{Pracochodność} = A * (\text{rozmiar})^B$$

$$\text{Czas realizacji} = 2,5 * (\text{pracochodność})^C$$

	Projekt prosty	Projekt złożony
A	2,4	3,6
B	1,05	1,20
C	0,38	0,32

## THORNTON BUDŻET

- bezpośrednie prace własne (analiza, projekt, implementacja, testowanie, zarządzanie, dokumentacja, administracyjne)
- prace własne innych firm

- dodatkowe usługi (konsultingowe, audytowe, szkolenia)
- sprzęt zakupiony lub dzierżawiony na potrzeby projektu
- oprogramowanie zakupione na potrzeby projektu (licencje)
- powołanie biura (wynajmowanie lub własne)
- telekomunikacja
- szkolenie
- wypłaty skarbce

## Zarządzenie biurem projektu

- kontrola postępu pracy
  - punkty kontrolne
  - punkty kontrolne zadań
  - szacowanie czasu do zakończenia
- kontrola budżetu
  - nakłady zewnętrzne
  - koszty pracy

↓ Verta

## — Rozporządzenie trudności

- błąd pojemnych etapów
- zbyt małe siły
- braki innych zasobów
- błędne oszacowanie potrzeb
- braki drabek
- dodatkowe zasoby
- nieopracowane zespoły
- zwiększenie nadzoru
- zmiana kolejności pracy
- zmiana metod lub narzędzi
- opracowanie kontroli jakości
- zmniejszenie zakresu
- zatrzymanie projektu