Metryki

Pomiar złożoności modułowej i międzymodułowej **oprogramowania**

autor: Zofia Kruczkiewicz

Metryki złożoności modułowej i międzymodułowej **Chidamber & Kemerer** (CK)

Metryki złożoności modułowej i międzymodułowej **Chidamber & Kemerer** (CK), uzupełnione przez innych autorów

1.1. Podstawowe metryki CK:

- międzymodułowe CBO, RFC
- modułowe WMC, DIT, NOC, LCOM1.

1.2. Uzupełniony zbiór metryk przez innych autorów:

- międzymodułowe CA
- modułowe NPM

Strona autorów narzędzia ckjm: http://www.spinellis.gr/sw/ckjm/

ckjm — Chidamber and Kemerer Java Metrics - Windows Internet Explorer

http://www.spinellis.gr/sw/ckjm/

Plik Edycja Widok Ulubione Narzędzia Pomoc

Google extended ckjm

Szukaj Więcej » Zaloguj się

Ulubione Sugerowane witryny Pobierz więcej dodatk...

ckjm — Chidamber and Kemerer Java ...

Strona Bezpieczeństwo Narzędzia

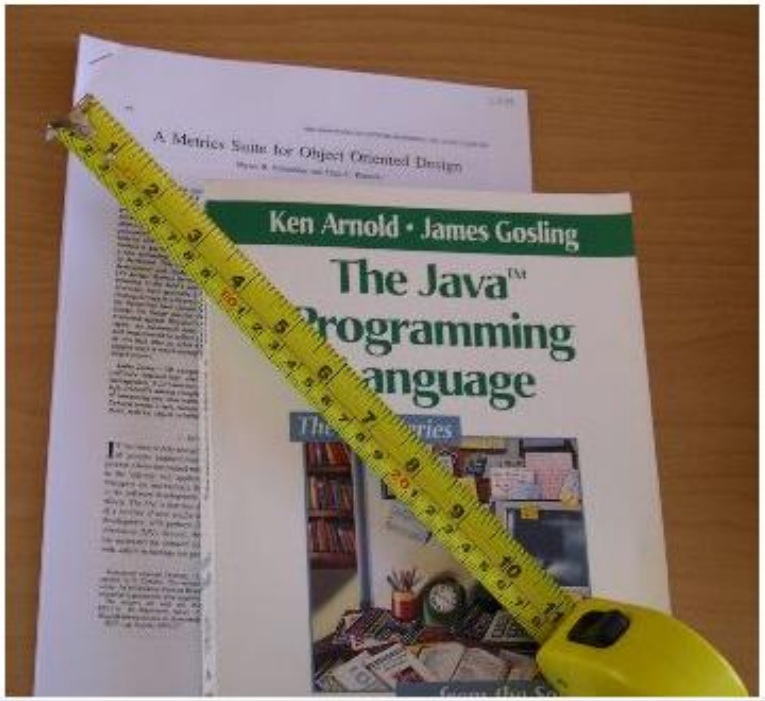
ckjm — Chidamber and Kemerer Java Metrics

The program *ckjm* calculates Chidamber and Kemerer object-oriented metrics by processing the bytecode of compiled Java files. The program calculates for each class the following six metrics proposed by Chidamber and Kemerer.

- WMC: Weighted methods per class
- DIT: Depth of Inheritance Tree
- NOC: Number of Children
- CBO: Coupling between object classes
- RFC: Response for a Class
- LCOM: Lack of cohesion in methods

In addition it also calculates for each class

- Ca: Afferent couplings
- NPM: Number of public methods



Internet | Tryb chroniony: włączony

100%

Metrics information

Name	Range
V(G) (cyclomatic-complexity)	1.0 ... 10.0
LOC (total-lines-of-code)	5.0 ... 1000.0
NCLOC (noncomment-lines-of-code)	0.0 ... 0.0
CLOC (comment-lines-of-code)	0.0 ... 0.0
DC (density-of-comments)	0.2 ... 0.4
NP (number-of-parameters)	0.0 ... 4.0
EXEC (executable-statements)	0.0 ... 20.0
WMC (weighted-methods-per-class)	1.0 ... 50.0
RFC (response-for-class)	0.0 ... 50.0
DIT (depth-in-tree)	0.0 ... 5.0
NOC (number-of-children-in-tree)	0.0 ... 10.0
Ce (efferent-coupling)	0.0 ... 20.0
A (abstractness)	0.0 ... 0.5
NOT (number-of-types)	0.0 ... 80.0
NOTa (number-of-abstract-types)	0.0 ... 20.0
NOTc (number-of-concrete-types)	0.0 ... 80.0
NOTe (number-of-exported-types)	3.0 ... 50.0
LSP (limited-size-principle)	0.0 ... 10.0
DIP (dependency-inversion-principle)	0.3 ... 1.0
MQ (modularization-quality)	0.0 ... 1000.0
NT (number-of-tramps)	0.0 ... 1.0
LCOM (lack-of-cohesion)	0.0 ... 0.2

Zakresy wartości metryk, między innymi metryk CK

CKJM Chidamber and Kemerer Java Metrics - Windows Internet Explorer

Przykład metryk CK wyznaczonych porzez program **ckjm 1.8 extended** aplikacji typu Java Application z modelem obiektowym opartym na klasach zdefiniowanych przez użytkownika **(część warstwy biznesowej)** oraz klasach typu Controller technologii JPA **(warstwa integracji)**

Top 25: lcom3

[wmc] [dit] [noc] [cbo] [rfc] [lcom] [ca] [npm] [lcom3] [explanations]

name	wmc	dit	noc	cbo	rfc	lcom	ca	npm	lcom3
wypozyczalnia1app.TFabryka	3	1	0	4	20	3	2	3	2.0
wypozyczalnia1app.TTytul_ksiazki	22	1	1	2	36	189	6	22	0.9047619047619048
wypozyczalnia1app.TEgzemplarz	12	1	1	1	20	42	5	12	0.8409090909090909
wypozyczalnia1app.TTytul_ksiazki_na_kasecie	4	2	0	1	9	4	1	4	0.8333333333333334
wypozyczalnia1app.TEgzemplarz_termin	5	2	0	1	12	0	1	5	0.75
wypozyczalnia1app.TEgzemplarzController	9	1	0	2	29	34	0	8	0.125
wypozyczalnia1app.TTytul_ksiazkiController	9	1	0	1	34	34	0	8	0.125
wypozyczalnia1app.TAplikacja	11	1	0	3	29	13	0	11	0.0

Explanations

WMC - Weighted methods per class
A class's *weighted methods per class* WMC metric is simply the sum of the complexities of its methods. As a measure of complexity we can use the cyclomatic complexity, or we can abritrarily assign a complexity value of 1 to each method. The *ckjm* program assigns a complexity value of 1 to each method, and therefore the value of the WMC is equal to the number of methods in the class.

DIT - Depth of Inheritance Tree
The *depth of inheritance tree* (DIT) metric provides for each class a measure of the inheritance

Komputer | Tryb chroniony: wyłączony

100%

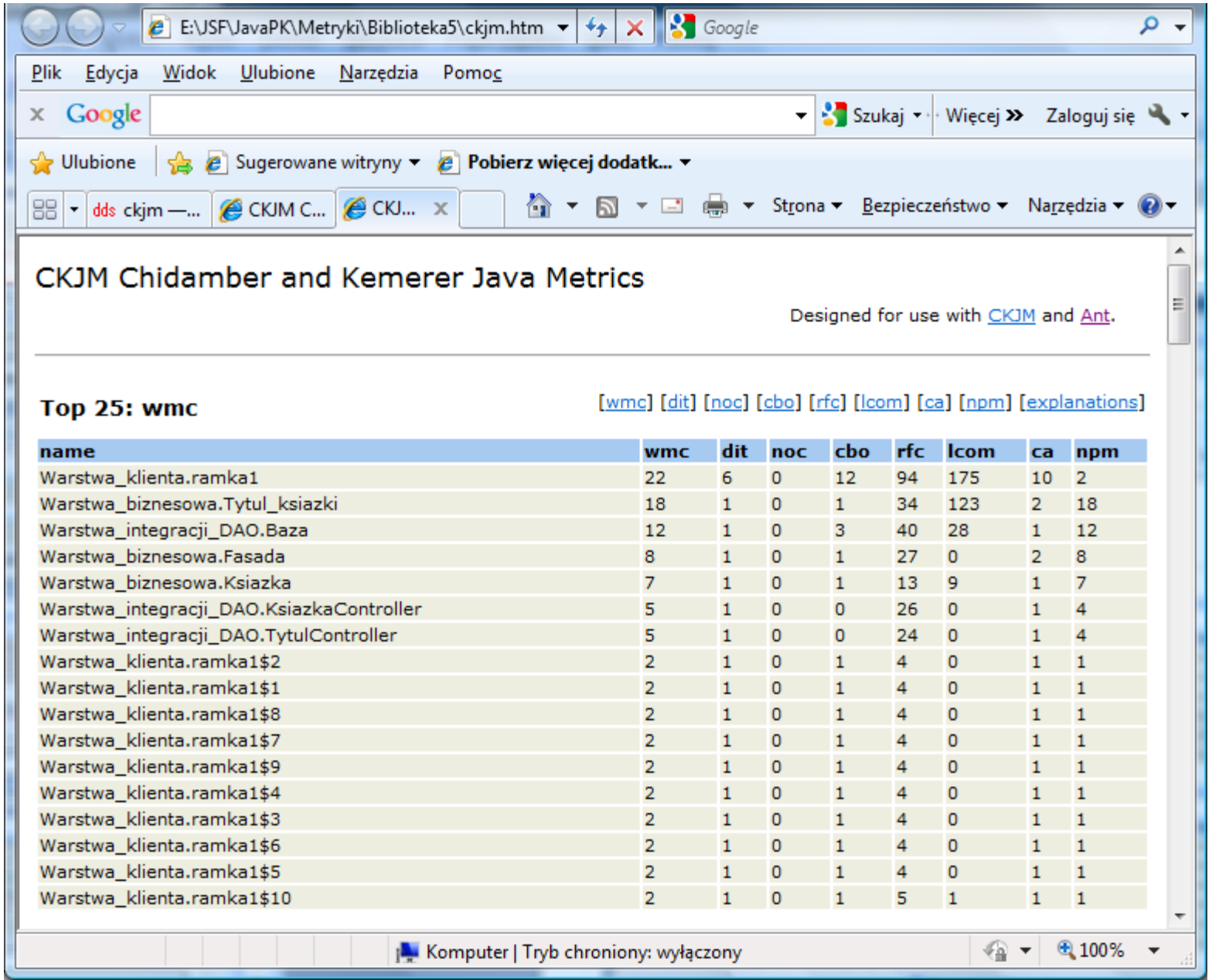
Przykład metryk CK wyznaczonych porzez program **ckjm 1.8 extended** aplikacji typu Visual Web Java Server Faces, zawierającej **warstwy prezentacji (internetowej)** i **część biznesowej** (obejmującą RequestBean1, SessionBean1 oraz ApplicationBean1), jako pozostała część pełnej aplikacji internetowej (slajd poprzedni zawiera pomiar metryk pozostałych warstw tej aplikacji - biznesowej i integracji).

Top 25: lcom3

[wmc] [dit] [noc] [cbo] [rfc] [lcom] [ca] [npm] [lcom3] [explanations]

name	wmc	dit	noc	cbo	rfc	lcom	ca	npm	lcom3
webwypozyczalnia2.RequestBean1	6	3	0	3	11	15	14	3	2.0
webwypozyczalnia2.Tytulybaza	35	3	0	8	44	511	0	31	0.9411764705882353
webwypozyczalnia2.Ksiazkibaza	27	3	0	8	36	291	0	23	0.9230769230769231
webwypozyczalnia2.Tytuly	26	3	0	15	44	277	0	22	0.92
webwypozyczalnia2.Menu	26	3	0	5	32	283	6	22	0.92
webwypozyczalnia2.Baza_tytul	26	3	0	15	45	277	0	22	0.92
webwypozyczalnia2.Baza_ksiazki	24	3	0	13	40	234	0	20	0.9130434782608695
webwypozyczalnia2.Baza_tytuly	24	3	0	13	40	234	0	20	0.9130434782608695
webwypozyczalnia2.Ksiazki	24	3	0	18	49	234	0	20	0.9130434782608695
webwypozyczalnia2.Page1	22	3	0	12	40	195	0	18	0.9047619047619048
webwypozyczalnia2.FormTytul	31	3	0	6	43	349	2	27	0.9
webwypozyczalnia2.FormKsiazka	19	3	0	6	31	115	1	15	0.8333333333333334
webwypozyczalnia2.Logo	11	3	0	6	18	43	0	7	0.8
webwypozyczalnia2.ApplicationBean1	25	3	0	7	52	234	16	24	0.7666666666666666
webwypozyczalnia2.SessionBean1	9	3	0	2	14	30	15	7	0.75
webwypozyczalnia2.Ksiazkiaplikacja	9	3	0	5	15	30	0	5	0.75
webwypozyczalnia2.Tytulyaplikacja	11	3	0	8	25	43	1	7	0.6

Przykład pomiaru metryk CK wyznaczonych za pomocą programu **ckjm 1.9** aplikacji typu Java Application, zawierającej warstwy: **klienta, biznesową i integracji**



name	wmc	dit	noc	cbo	rfc	lcom	ca	npm
Warstwa_klienta.ramka1	22	6	0	12	94	175	10	2
Warstwa_biznesowa.Tytul_ksiazki	18	1	0	1	34	123	2	18
Warstwa_integracji_DAO.Baza	12	1	0	3	40	28	1	12
Warstwa_biznesowa.Fasada	8	1	0	1	27	0	2	8
Warstwa_biznesowa.Ksiazka	7	1	0	1	13	9	1	7
Warstwa_integracji_DAO.KsiazkaController	5	1	0	0	26	0	1	4
Warstwa_integracji_DAO.TytulController	5	1	0	0	24	0	1	4
Warstwa_klienta.ramka1$2	2	1	0	1	4	0	1	1
Warstwa_klienta.ramka1$1	2	1	0	1	4	0	1	1
Warstwa_klienta.ramka1$8	2	1	0	1	4	0	1	1
Warstwa_klienta.ramka1$7	2	1	0	1	4	0	1	1
Warstwa_klienta.ramka1$9	2	1	0	1	4	0	1	1
Warstwa_klienta.ramka1$4	2	1	0	1	4	0	1	1
Warstwa_klienta.ramka1$3	2	1	0	1	4	0	1	1
Warstwa_klienta.ramka1$6	2	1	0	1	4	0	1	1
Warstwa_klienta.ramka1$5	2	1	0	1	4	0	1	1
Warstwa_klienta.ramka1$10	2	1	0	1	5	1	1	1

Metryki złożoności międzymodułowej

Osłabienie powiązań między-modułowych prowadzi do zmniejszenia oddziaływań między modułami oraz poprawy struktury oprogramowania.

Elementami łączącymi wyjściowymi z innymi modułami są:

- **Funkcja/metoda wywołująca funkcję z innego modułu**
- wszystkie elementy importowane z innych modułów
- każda informacja z poza modułu potrzebna do zdefiniowania ciała funkcji (np. obsługa błędów), definicji typu strukturalnego, definicji dowolnej zmiennej

Elementami łączącymi wejściowymi dany moduł z innymi modułami są:

- **funkcja/metoda danego modułu wywoływana przez funkcję/metodę z innego modułu**
- Wszystkie elementy modułu przekazywane w importowanych modułach
- informacja zawarta w module potrzebna w innych modułach do dowolnej definicji (np. obsługa błędów), definicji typu strukturalnego, definicji dowolnej zmiennej

RFC - metryki połączeń wyjściowych

RFC = M + R oraz RFC' = M + R'

Zakres wartości (1 – 50)

gdzie

M – liczba w danej klasie

R – liczba metod wywoływanych przez metody M z innych klas

R' – R + pozostałe metody wywoływane zgodnie z drzewem wywołań

R i R' są wywołanymi metody zwykłymi lub wirtualnymi (tyle razy liczonymi, ile klas przesłania metodę)

Uwagi:

1. Duża wartość metryki oznacza dużo błędów
2. Duża wartość metryki oznacza duży wysiłek przy testowaniu
3. Duża wartość metryki oznacza trudność w zrozumieniu klasy

CBO – metryka połączeń wyjściowych z innymi klasami, z którymi jest powiązana dana klasa Zakres wartości (0..14)

Wartość metryki oznacza liczbę klas powiązanych przez wywołanie metod zwykłej lub wirtualnej innych klas (tyle razy liczonej, ile klas przesłania metodę), zastosowanie odwołania do zmiennej (wzajemne powiązanie między klasami jest liczone tylko raz) własnej klasy i przez dziedziczenie, przez argumenty metody, przez typy danych zwracane przez return oraz powiązania za pomocą wyjątków– wartość do 14

Uwagi:

1. Zbyt duża wartość wymaga dużego wysiłku przy testowaniu
2. Ograniczone zastosowanie zbyt powiązanej klasy w innych programach – gorsza wielożywalność

Fan-out – metryka połączeń wyjściowych

Metryka *Fan-out* wyznacza liczbę połączeń *elementów wyjściowych* jednego modułu z *elementami wejściowymi* innych modułów. Uwzględnia się tylko jedno dowolne połączenie wyjściowe-wejściowe z każdym z modułów.

Fan-in – metryka połączeń wejściowych

Metryka *Fan-in* wyznacza liczbę połączeń *elementów wejściowych* jednego modułu z *elementami wyjściowymi* innych modułów. Uwzględnia się tylko jedno dowolne wejściowo-wyjściowe połączenie z każdym z modułów.

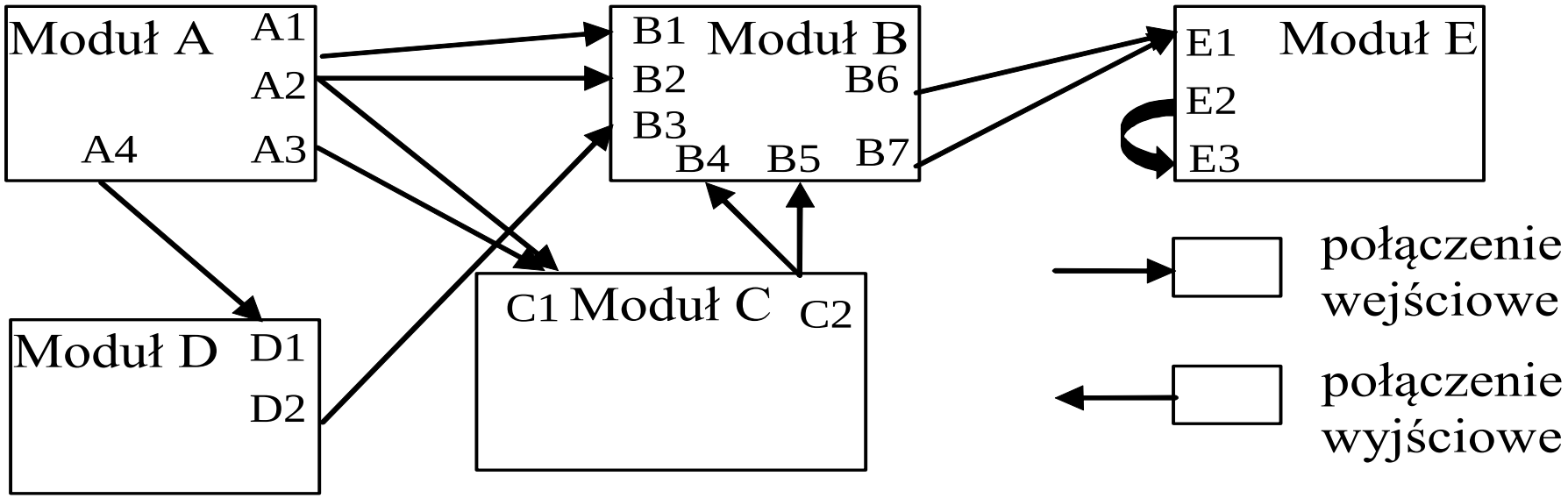
Ca - metryka połączeń wejściowych

Metryka CA wyznacza liczbę klas, które używają danej klasy przez wywołanie jej metod zwykłych lub wirtualnych (tyle razy liczonych, ile klas przesłania metodę), zastosowanie odwołania do zmiennej (wzajemne powiązanie między klasami jest liczone tylko raz) typu danej klasy i dziedziczonych przez nią atrybutów, przez argumenty metod typu danej klasy, wyniki typu danej klasy zwracane przez return oraz wyjątki– definicja powiązań wejściowych jest taka sama jak CBO.

Przykład rozwiązania dla modułu A (rysunek z następnego slajdu)

- Moduł A zawiera elementy łączące wyjściowe: A1, A2 ,A3 ,A4. Moduł B dla modułu A zawiera łączące elementy wejściowe B1, B2, moduł C zawiera łączący element wejściowy C1 oraz moduł D zawiera element wejściowy łączący D1 oraz:
- A1 łączy się z B1
- A2 łączy się B2, C1
- A3 łączy się C1
- A4 łączy się D1
- RS={A1,A2,A3,A4} $\cup$ {B1,B2} $\cup$ {D1} $\cup$ {C1} = {A1,A2,A3,A4,B1,B2,D1,C1}
- RFC= |RS|=8
- Fan-out= |{<A1,B1>, <A2,C1>, <A4,D1>}| = 3 *//dowolny element wejściowy*
- Fan-in=|{}|=0
- R={<A1,B1>, <A2,B2>, <A2,C1>,<A3,C1>,<A4,D1>}
- |R|=5

Przykłady metryk międzymodułowych dla modułów A, B, C, D, E cd.



	A	B	C	D	E
Fan-out	3	1	1	1	1
Fan-in	0	3	1	1	2
RFC	8	3	3	2	2
|R|	5	2	2	1	1

Metryki złożoności modułowej

Wzmocnienie ***powiązań wewnątrz-modułowych*** prowadzi do zmniejszenia oddziaływań między modułami oraz poprawy struktury oprogramowania.

Metryki rozmiaru

SLOC

- Jest to liczba wierszy kodu źródłowego programu liczona niezależnie od liczby instrukcji lub fragmentów instrukcji znajdujących się w każdym wierszu. Nie wlicza się wierszy z komentarzami lub pustych wierszy.
- SLOC jest powszechnie używaną metryką do szacowania nakładów pracy nad programem oraz jest mocno skorelowana z testowalnością, konserwowalnością i zrozumiałością.
- Zakres wartości 5 -1000 linii

S/C

- Metryka ta jest liczbą wszystkich elementów programu należących do bloków logicznych:
- inicjowanie zmiennych sterujących — **int** i=0
- porównanie — i <10
- zwiększanie zmiennej sterującej — i++
- liczba instrukcji w każdym bloku — **for** (;;) {...}

Żetony

- Jest to zbiór metryk, które określają liczbę:
- $\eta 1$ - *liczbę typów operatorów(słownik typów operatorów)*, czyli liczbę: operatorów predefiniowanych (logicznych, arytmetycznych, przypisania, relacyjnych itp.), słowa kluczowe instrukcji (**while, if, else, do**), nazwy funkcji
- $\eta 2$ - *liczbę typów argumentów(słownik typów argumentów)*, czyli liczbę: wszystkich symboli reprezentujących dane przy deklaracji i definicji
- $\eta 3$ - *liczbę wszystkich wystąpień operatorów*
- $\eta 4$ - *liczbę wszystkich wystąpień argumentów*

NPM - liczba metod publicznych

- Metryka wyznacza liczbę metod publicznych , która pozwala wyznaczyć miarę rozmiaru API pakietu, w którym znajduje się klasa.

WMC - Liczba metod w klasie
Zakres wartości (1 - 50)

- **Suma złożoności metod** w klasie (struktura logiczna i rozmiar)

$$WMC = \sum_{i=1}^{n} c_i$$

- gdzie *ci* jest statyczną złożonością każdej z *i* - metod (złożoność cyklomatyczna materiał podany dalej). Jeżeli *ci* jest równe 1, wtedy *WMC* jest równe liczbie metod *n*. *WMC* maleje przy wykorzystaniu polimorfizmu i dziedziczenia

Uwagi:

- Zbyt duża wartość metryki powoduje w klasie więcej błędów
- Zbyt duża wartość oznacza mniejszą wielożywalność klasy
- Zbyt duża wartość powoduje mniejsze zrozumienie odpowiedzialności klasy

DIT - Głębokość dziedziczenia
Zakres wartości (0 - 5)

- czyli liczba poziomów w drzewie dziedziczenia odniesiona do liczby klas, określająca zakres dziedziczenia (rozmiar)

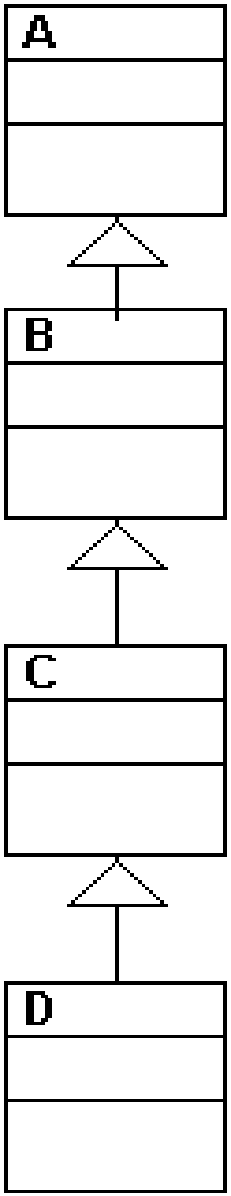
$$DIT = \frac{\sum \ glebokosc\ dziedziczenia}{calkowita\ liczba\ klas}$$

Uwagi:

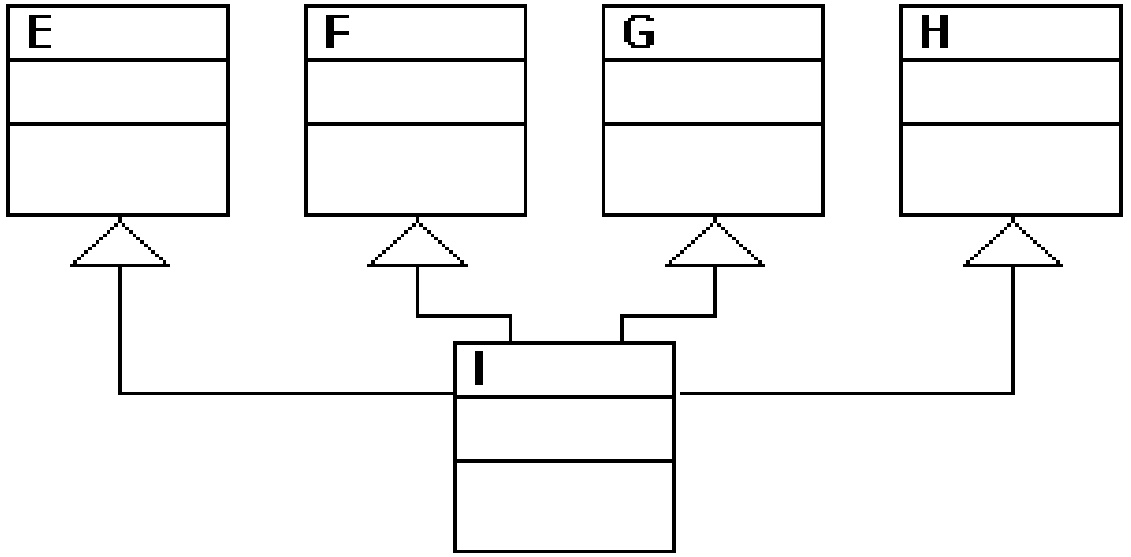
1. Przy głębokim drzewie dziedziczenia rośnie wieloużywalność
2. Przy głębokim drzewie dziedziczenia rośnie też liczba błędów, szczególnie w klasach należących do środkowych poziomów dziedziczenia

(1) Przykłady modeli do pomiaru metryk dziedziczenia

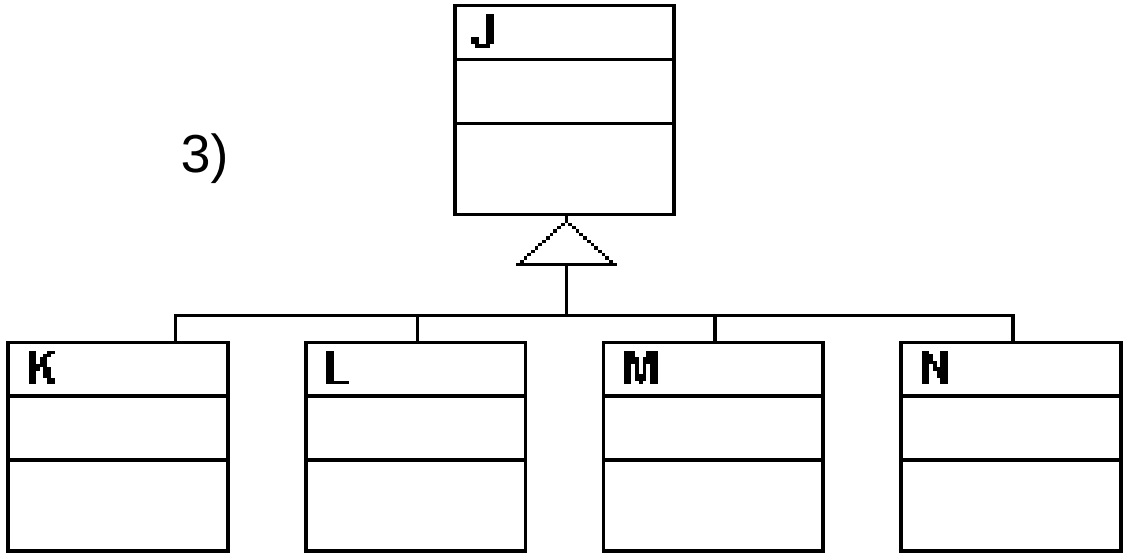
1)



2)

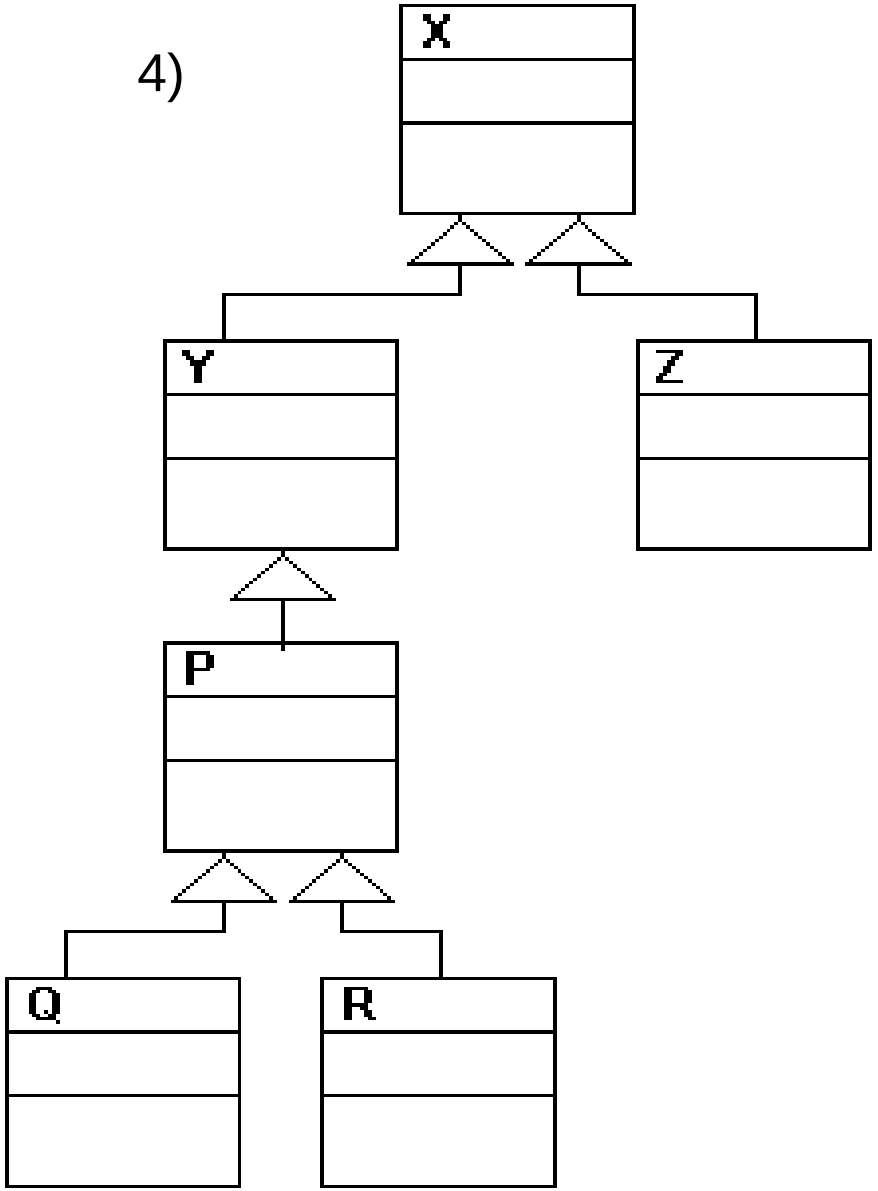


3)

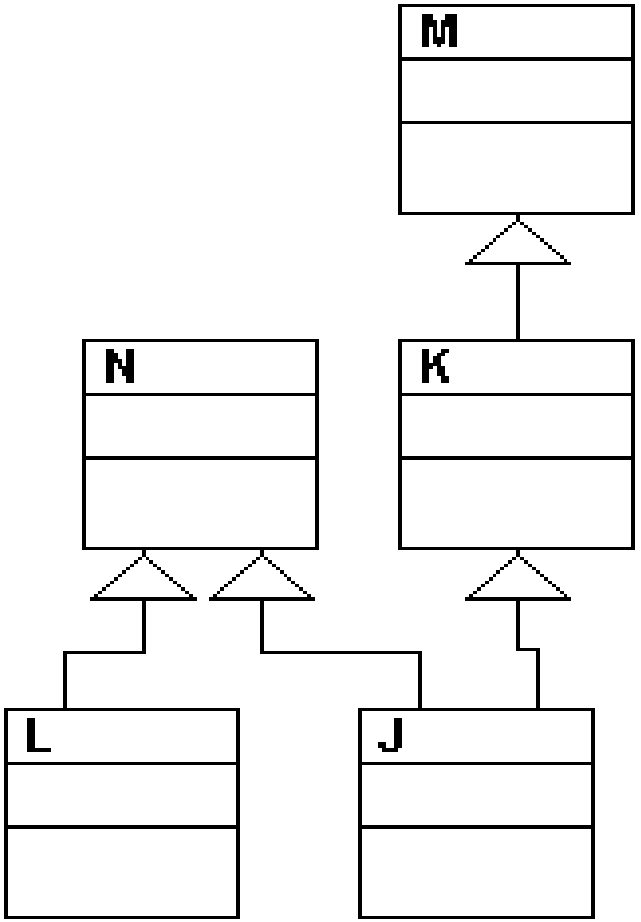


(2) Przykłady modeli do pomiaru metryk dziedziczenia

4)



5)



Metryka / Przykład	S (specialization)	U (reuse)	DIT
1*	1/3 $\rightarrow$ 0	3/4 $\rightarrow$ 1	(0+1+2+3)/4=1.5
2*	1/4 $\rightarrow$ 0	4/5 $\rightarrow$ 1	(0+0+0+0+(1+1+1+1)/4)/5=0.2
3	4/1 $\rightarrow$ μ	1/5 $\rightarrow$ 0	(0+1+1+1+1)/5=0.8
4	3/3	3/6	(0+1+1+2+3+3)/6=1.5
5	2/3	3/5	(0+0+1+(1+2)/2+1)/5=0.7

$$DIT = \frac{\sum \ glebokosc\ dziedziczenia}{calkowita\ liczba\ klas}$$

$$S = \frac{liczba\ \ PodTypów}{liczba\ \ \ NadTypów}$$

$$U = \frac{liczba\ \ NadTypów}{całkowita\ \ liczba\ \ klas}$$

Przykłady 1 i 2 reprezentują ubogi schemat dziedziczenia.

Wartości *U* bliska 1 oraz S bliską 0 określają liniowy model dziedziczenia. Wartości U<<1 oraz S >>1 oznaczają pożądaną wartość.

NOC – liczba klas dziedziczących
Zakres wartości (0..10)

Uwagi

1. Zbyt dużo podklas oznacza dużo testowania
2. Zbyt dużo podklas może powodować błędne użycie tych podklas

Metryki logicznej struktury programu, czyli przepływu sterowania

Liczby cyklomatyczne McCabe

Zakres wartości (1 -10)

VLI (G) = e – n +p+1

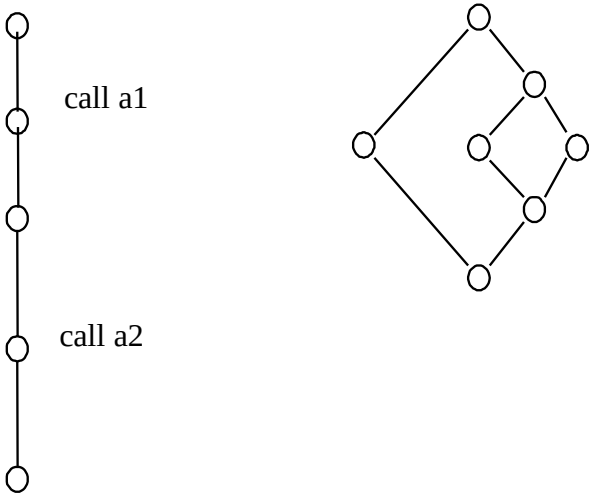
Liczba ta jest wyznaczana na podstawie grafu przedstawiającego drogi sterowania w programie, gdzie *n* jest liczbą wierzchołków grafu reprezentujących poszczególne instrukcje, w tym wywołania funkcji, *e* jest liczbą krawędzi grafu reprezentujących połączenia poszczególnych realizacji instrukcji, *p* jest liczbą podgrafów rozłącznych, a każda funkcja stanowi niezależny podgraf, którego wywołanie jako wierzchołek jest umieszczony w innym podgrafie.

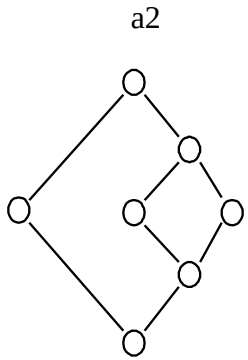
V(G) = e – n + 2*p

Metryka V(G) uwypukla istnienie funkcji za pomocą składnika 2*p, VLI (G) natomiast wywołanie funkcji traktuje na równi z innymi instrukcjami.

(1) Przykład prezentujący obliczenia metryk MC Cabe

a)





Cała aplikacja

a) e=20, n=19, p=3

V(G) = e-n+2*p =20– 19 + 2*3=7

$\mathbf{V_{LI}(G)}$ = e–n+p+1=20-19+3+1=5

b) e=23, n=20, p=1

$\mathbf{V(G) = V_{LI}(G)}$

= e – n + 2 = e – n + 2*p

= 23-20+2=5

Metody a1 i a2 (przypadki a i b):

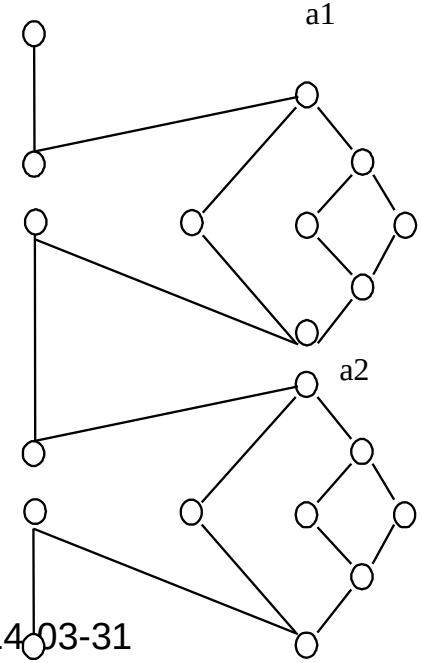
e=8, n=7, p=1

$\mathbf{V(G)= V_{LI}(G)}$ =

= e – n + 2 = e – n + 2*p

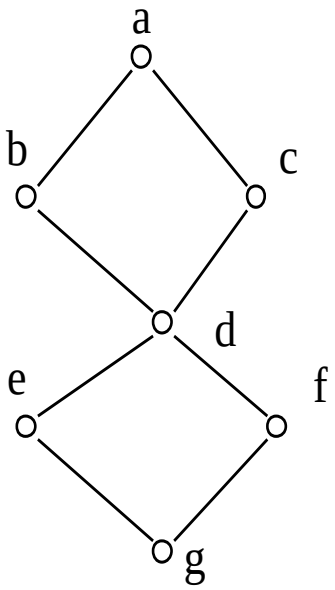
= 8-7+2=3

b)

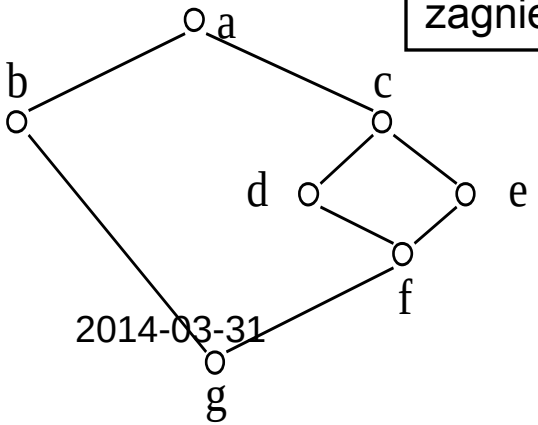


	Metoda a1	Metoda a2	Całość
a			
V(G)	3	3	7
$V_{LI}(G)$	3	3	5
b			
V(G)	3	3	5
$V_{LI}(G)$	3	3	5

(2) Przykład prezentujący obliczenia metryk MC Cabe



Zgodnie z aksjomatem 7, pętla zagnieżdżona powinna mieć złożoność różną od programu z dwiema sekwencyjnie wykonywanymi pętlami.
Jednak zarówno SLOC, V(G), VLI(G) są identyczne w obu rozwiązaniach, natomiast różne są wartości metryki S/C. Wg metryki S/C bardziej złożony jest program z zagnieżdżoną pętlą.



dwie pętle sekwencyjne

```
a: while (x>= 0)
c:  {x=x-y;   }          (gdy a==true)
b:                       (gdy a==false)
d: while (y>= 10)        (koniec a)
f: { x=x+1;              (gdy d==true)
     y=y-1;
   }
e:                       (gdy d==false)
g:                       (koniec d, koniec programu)
```

V(G)=e-n+2*p=3
VLI(G)=e-n+p+1=8-7+2=3
SLOC=7
S/C=7

b) podwójna pętla zagnieżdżona

```
a: while (x>= 0)
 { x=x-y;                (gdy a==true)
   c: while (y>= 10)     (gdy a==true)
    e:  { x=x+1;         (gdy c==true i a==true)
          y=y-1;}
    d:                   (gdy c==false i a==true)
     f:                  (koniec c i a==true)
  }
 b:                      (gdy a==false)
g:                       (koniec a, koniec programu)
```

V(G)=e-n+2*p=3
VLI(G)=e-n+p+1=8-7+2=3
SLOC=7
S/C=9

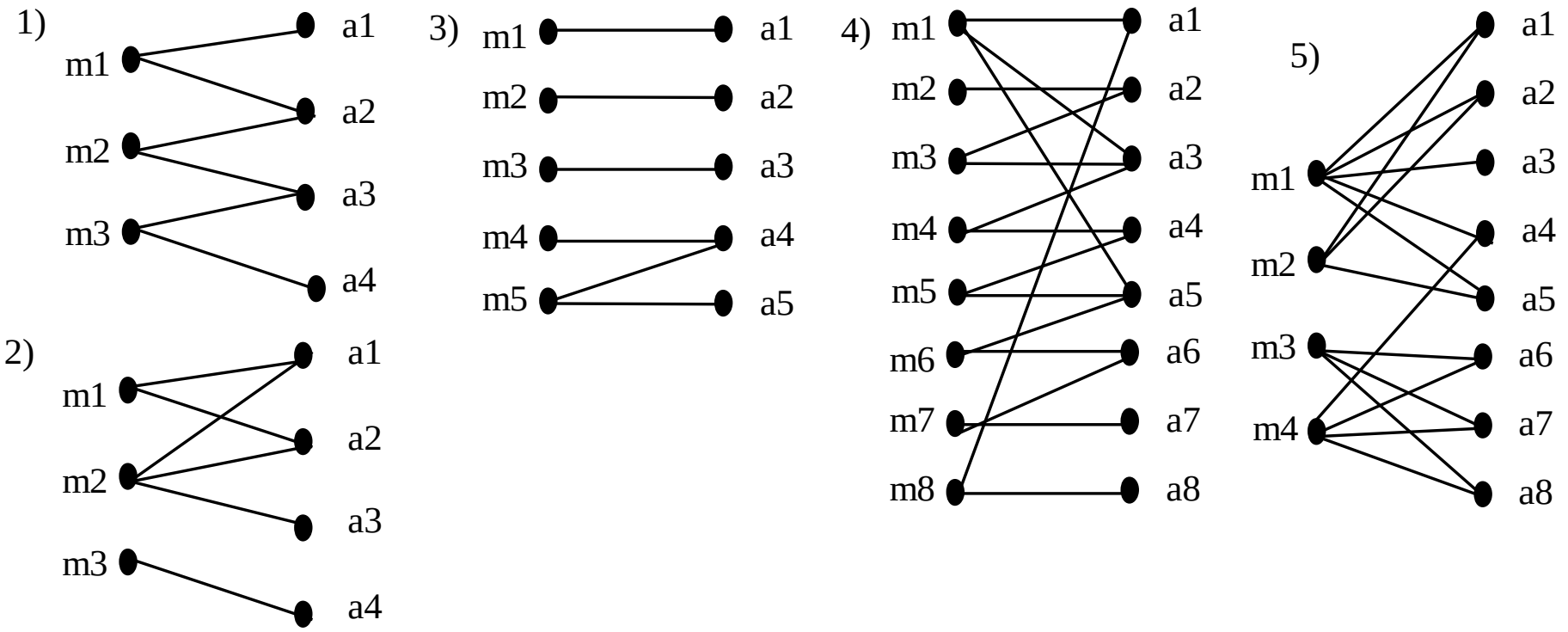
Metryki spójności klasy

LCOM1 – metryka wyznacza sumę P zbioru wszystkich par metod operujących na zbiorach rozłącznych atrybutów oraz sumę Q zbioru wszystkich par metod operujących na zbiorach spójnych atrybutów. Różnica mocy tych zbiorów jest wartością metryki, gdy moc |P| jest większa od mocy |Q|, w przeciwnym wypadku jest równa 0.

Jeśli klasa jest minimalnie spójna (żadna metoda nie jest powiązana z inną metodą i liczba metod jest równa n. Wtedy |P| = (n-1)*n/2 i |Q|= 0, czyli LCOM1=(n-1)*n/2)

Uwagi:
1) Duża wartość metryki oznacza trudność testowania,
2) jednak mała wartość lub równa 0 nie zawsze oznacza klasę poprawnie zbudowaną.
3) Zbyt wiele różnych klas ma tę samą wartość metryki.
4) Brak modelowania property i uwzględnienia wywoływania metody przez metodę

Grafy dwudzielne jako modele klas do wyznaczania metryki LCOM



(1) Przykłady obliczeń metryki LCOM1

1) – trzy metody

- Metoda 1 ma zbiór atrybutów I1 = {a1, a2}
- Metoda 2 ma zbiór atrybutów I2 = {a2, a3}
- Metoda 3 ma zbiór atrybutów: I3 = {a3, a4}
- Zbiór rozłącznych par: P = {(I1, I3)} -> |P| = 1
- Zbiór spójnych par: Q = {(I1, I2), (I2, I3)} -> |Q| = 2
  - **LCOM = 0 dla |P| <= |Q|**

2) - trzy metody

- Metoda 1 ma zbiór atrybutów I1 = {a1, a2}
- Metoda 2 ma zbiór atrybutów I2 = {a1, a2, a3}
- Metoda 3 ma zbiór atrybutów: I3 = {a4}
- Zbiór rozłącznych par: P = {(I1, I3), (I2, I3)} -> |P| = 2
- Zbiór spójnych par: Q = {(I1, I2)} -> |Q| = 1
  - **LCOM = |P| - |Q| = 2 – 1 = 1 dla |P| > |Q|**

3) – pięć metod

- Metoda 1 ma zbiór atrybutów I1 = {a1}
- Metoda 2 ma zbiór atrybutów I2 = {a2}
- Metoda 3 ma zbiór atrybutów: I3 = {a3}
- Metoda 4 ma zbiór atrybutów: I4 = {a4}
- Metoda 5 ma zbiór atrybutów: I5 = {a4, a5}
- Zbiór rozłącznych par: P = {(I1, I2), (I1,I3), (I1,I4), (I1, I5), (I2, I3), (I2, I4), (I2, I5), (I3, I4), (I3,I5)} -> |P| = 9
- Zbiór spójnych par: Q = {(I4, I5)} -> |Q| = 1
  - **LCOM = |P| - |Q| = 9-1=8 dla |P| > |Q|**

(2) Przykłady obliczeń metryki LCOM1

4) – osiem metod

- Metoda 1 ma zbiór atrybutów I1 = {a1, a3, a5}
- Metoda 2 ma zbiór atrybutów I2 = {a2}
- Metoda 3 ma zbiór atrybutów: I3 = {a2, a3}
- Metoda 4 ma zbiór atrybutów: I4 = {a3, a4}
- Metoda 5 ma zbiór atrybutów: I5 = {a4, a5}
- Metoda 6 ma zbiór atrybutów: I6 = {a5, a6}
- Metoda 7 ma zbiór atrybutów: I7 = {a6, a7}
- Metoda 8 ma zbiór atrybutów: I8 = {a1, a8}
- Zbiór rozłącznych par: P = {(I1, I2), (I1, I7), (I2, I4), (I2, I5), (I2, I6), (I2, I7), (I2, I8), (I3, I5), (I3, I6), (I3, I7), (I3, I8), (I4, I6), (I4, I7), (I4, I8), (I5, I7), (I5, I8), (I6, I8), (I7, I8)}
  -> |P| = 18
- Zbiór spójnych par: Q = {(I1, I3), (I1, I4), (I1, I5), (I1, I6) (I1, I8) (I2, I3), (I3, I4), (I4, I5), (I5, I6), (I6, I7),}
  -> |Q| = 10

- **LCOM = |P| - |Q| = 18-10=8 dla |P| > |Q|**

(3) Przykłady obliczeń metryki LCOM1

5) – cztery metody

- Metoda 1 ma zbiór atrybutów I1 = {a1, a2, a3, a4, a5}
- Metoda 2 ma zbiór atrybutów I2 = {a1, a2, a5}
- Metoda 3 ma zbiór atrybutów: I3 = {a6, a7, a8}
- Metoda 4 ma zbiór atrybutów: I4 = {a4, a6, a7, a8}
- Zbiór rozłącznych par: P = {(I1, I3), (I2, I3), (I2, I4)} -> |P| = 3
- Zbiór spójnych par: Q = {(I1, I2), (I1, I4), (I3, I4)} -> |Q| = 3
    - **LCOM = 0 dla |P| <= |Q|**

Rozszerzenie definicji metryk spójności LCOM (1)

Metryka LCOM2 (Constantine & Graham, Henderson-Sellers)

$$LCOM2 = 1 - \frac{\left(\frac{1}{a}\sum_{j=1}^{a} \mu(A_j)\right)}{m} = 1 - \frac{r}{m * a}$$

- gdzie *m* jest liczbą wierzchołków zbioru *M* metod, *a* jest liczbą wierzchołków *A* atrybutów, natomiast wyrażenie $\mu(Aj)$ liczbą krawędzi grafu wiążącą atrybut *Aj* z określoną liczbą metod (elementy zbioru R).
- Maksymalna i zarazem najlepsza wartość spójności LCOM2 oznacza wartość 0 metryk, co uzyskuje się przy grafie pełnym *(r* = |*M*|*|*A*| krawędzi).
- Wartość metryki LCOM2 zawarta między „0..mniejszy od 1” oznacza obiektowy model klasy, jednak warta bliska 1 oznacza najgorszy przypadek klasy.
- W metryce LCOM2 muszą przynajmniej istnieć jedna metoda i jeden atrybut.

Kolejność grafów wg malejącej spójności: 5, 1, 4, 2, 3

Lp	*m*	*a*	*r*	*k*	*LCOM1*	*LCOM2*	*LCOM3*
1	3	4	6	1	0	0.5	0.75
2	3	4	6	2	1	0.5	0.75
3	5	5	6	4	8	0.76	0.95
4	8	8	16	1	8	0.75	0.8571
5	**4**	**8**	**15**	**1**	**0**	**0.5313**	**0.7083**

Rozszerzenie definicji metryk spójności LCOM (2)

Metryka LCOM3 (Constantine & Graham, Henderson-Sellers)

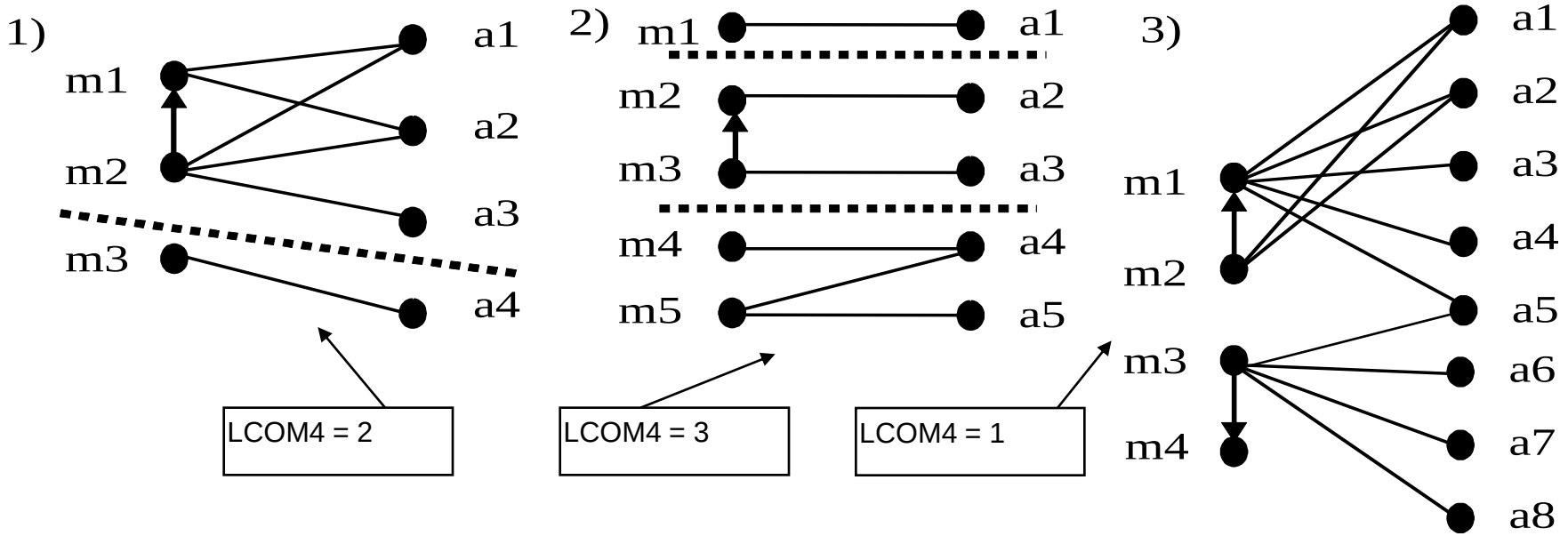
Zakres wartości „0..0.2”.

$$LCOM3 = \frac{\left(\frac{1}{a}\sum_{j=1}^{a}\mu(A_j)\right) - m}{1 - m} = \frac{\frac{r}{a} - m}{1 - m}$$

- gdzie *m* jest liczbą wierzchołków zbioru *M* metod, *a* jest liczbą wierzchołków *A* atrybutów, natomiast wyrażenie $\mu(Aj)$ liczbą krawędzi grafu wiążącą atrybut *Aj* z określoną liczbą metod (elementy zbioru R).
- Maksymalna i zarazem najlepsza wartość spójności LCOM3 oznacza wartość 0 metryk, co uzyskuje się przy grafie pełnym ($r=|M|*|A|$ krawędzi).
- Wartość metryki LCOM3 zawarta między „0..1” oznacza obiektowy model klasy (wartość 1 oznacza minimalnie spójną klasę – równa liczby metod i atrybutów). Dopuszczalny zakres „0..0.2”.
- W metryce LCOM3 w klasie nie może istnieć tylko jedna metoda i musi być przynajmniej jeden atrybut.

Rozszerzenie definicji metryk spójności LCOM (3)

LCOM4 - (Hitz & Montazeri)



- **LCOM4 mierzy liczbę „połączonych komponentów” w klasie.** „Połączony komponent” jest zbiorem połączonych metod (zbiór takich metod a i b, gdzie metoda a wywołuje metodę b lub metoda b wywołuje metodę a, lub obie metody a i b wywołują ten sam atrybut klasy) i atrybutów, przy czym dopuszcza się jeden taki komponent klasy.
- Jeśli wartość metryki jest równa 2 lub więcej, należy klasę podzielić na dwie klasy lub więcej klas, tak aby posiadała tylko jeden „połączony komponent”.

Przykład metryk trzech systemów

System analyzed	Java	Java	C++
Classes	46	1000	1617
Lines	50,000	300,000	500,000
Quality	"Low"	"High"	"Medium"
CBO	2.48	1.25	2.09
LCOM1	447.65	78.34	113.94
RFC	80.39	43.84	28.60
NOC	0.07	0.35	0.39
DIT	0.37	0.97	1.02
WMC	45.7	11.10	23.97