Zofia Kruczkiewicz, Politechnika Wrocławska

Streszczenie

W pracy rozważa się problem wpływu spójności modeli oprogramowania obiektowego na odporność na błędy. Podkreśla się nadrzędną rolę modelowania w procesie powstawania oprogramowania i wpływ charakterystyk wewnętrznych modelu na cechy zewnętrzne oprogramowania. Podstawą oceny modeli, przyjętą w pracy, są wartości opracowanych metryk spójności jako charakterystyk wewnętrznych oprogramowania oraz wpływ spójności na pojawianie się błędów w oprogramowaniu.

Spójność modeli oprogramowania obiektowego

1. Zagadnienia tworzenia efektywnego oprogramowania obiektowego

Podejście obiektowe charakteryzuje się syntetycznym tworzeniem oprogramowania wynikającym ze stopniowego rozwiązywania problemu. Synteza oprogramowania przebiega w ramach trzech etapów: analizy, projektowania i programowania. W technikach obiektowych etap analizy związany jest między innymi z wyodrębnianiem klas i obiektów czyli tworzeniem modeli obiektowych opartych na metodach wywodzących się z metody ER (Entity Relationship) podane przez Chena [5]. W pracy do tworzenia modelu obiektowego wybrano metodę

ORM (Object Role Modelling), która wywodzi się z metody ER i pozwala na budowę szczegółowego modelu klas i obiektów, umożliwiającego syntezę wszystkich zależności funkcyjnych i niefunkcyjnych występujących w problemie [1,5,6,9].

Tworzenie modelu struktury klas jako składnika modelu oprogramowania obiektowego [10] jest powiązane z oceną wpływu jego złożoności strukturalnej na trzy niezależne charakterystyki zewnętrzne oprogramowania: jakość (stopień osiągniętej abstrakcji, wieloużywalność, zrozumiałość, konserwowalność, testowalność), koszt i funkcjonalność [2,3,4,6,7,9]. Związki między charakterystykami wewnętrznymi i zewnętrznymi oprogramowania są wyznaczane za pomocą regresji wielorakiej w wyniku badań empirycznych. Stąd ocena zewnętrznych cech oprogramowania jest możliwa już we wczesnym stadium jego powstawania za pomocą pewnych metryk strukturalnych, wyznaczanych na podstawie obiektywnych pomiarów parametrów modeli. W niniejszej pracy opracowano metryki spójności dla modeli *ORM*. W pracy zdefiniowano *spójność* jako wyrażenie wiążące liczbę ról (lub predykatów związków między klasami) występujących w modelu, wynikającą z dziedziczenia, polimorfizmu oraz ograniczeń. Przyjęto bowiem, że powiązania między typami obiektowymi reprezentują minimalny zbiór metod realizujących te powiązania i stanowią szkielet komunikacji między wystąpieniami tych klas (populacji, obiektów).

W pracy przyjęto założenie, iż tym lepszy model, im ma mniejszą złożoność strukturalną, reprezentowaną między innymi przez większą spójność modelu. Wpływ nadrzędnej roli optymalizacji początkowej postaci oprogramowania obiektowego (tworzonej podczas analizy), przez dążenie do osiągnięcia możliwie największej spójności oprogramowania, wynika z prostych odwzorowań elementów modelu w dziedzinę ich reprezentacji na etapie implementacji czyli projektowania i programowania. Podjęto próbę formalnej oceny wpływu większej spójności modeli na poprawę odporności oprogramowania na błędy.

Problematyka tworzenia efektywnego oprogramowania stanowi kluczowe zagadnienie w inżynierii oprogramowania. Jakością oprogramowania obiektowego zajmują się specjalne grupy np.: ESEG (Experimental Software Engineering Group), ESERG (Empirical Software Engineering Research Group at Bournemouth University).

2. Wprowadzenie do metody modelowania ORM

Metoda przedstawia świat rzeczywisty jako kolekcję obiektów, które odgrywają swoje role i są uwidocznione w sformalizowany sposób na diagramie schematu konceptualnego [1,6,9]. Podstawowym elementem opisu jest typ relacji oraz typ obiektowy. Relacja w metodzie *ORM* jest nie tylko podzbiorem produktu kartezjańskiego, lecz odwzorowaniem, umożliwiającym określanie własności połączeń między elementami relacji. Na rysunku 1a pokazano typ relacji *f* w taki sam sposób jak typy obiektowe *X1* i *X2*. Typ relacji stanowi połączenie między rolami *p* i *q*, jakie obiekty *X1* i *X2* odgrywają w relacji *f*. Typ relacji *f* jest nazwany typem faktu. Definicja modelu *ORM* obejmuje

$$Model_ORM = (SI, Populacja, Ograniczenia),$$

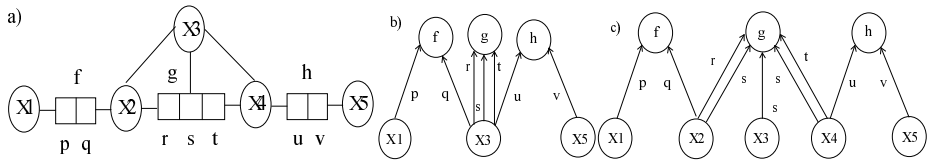
gdzie struktura informacyjna *SI* zawiera

$$SI = (TOb,TWart,TFakt,TRol,Fakt,Baza,Spec,SpecRole,WRole,Pol, PolRole,Korzen).$$

Ze względu na ramy artykułu podano jedynie krótkie wyjaśnienia dotyczące elementów modeli *ORM*: *TWart*- typ atrybutowy reprezentowany przez konkretne wartości (np. liczby); *TA*- abstrakcyjne typy obiektowe; *TRol*- skończony zbiór ról; *TFakt*- zbiór wynikający z podziału zbioru *TRol*; elementy zbioru *TFakt* określane jako typy faktu, spełniające zależność $TWart \cap TFakt \cap TA = \emptyset$; funkcja: $Fakt\text{-}TRol \rightarrow TFakt$ określająca typ faktu w zależności od roli, która należy do niego, zdefiniowana jako $Fakt(p)=f \Leftrightarrow p \in f$; typy obiektowe (zbiór niepusty)- $TOb=TA \cup TWart \cup TFakt$; funkcja określająca typ obiektowy pełniący daną rolę- $Baza: TRol \rightarrow TOb$; relacje częściowego porządku: dziedziczenia typu specjalizacja („w dół")-*Spec* oraz dziedziczenia uogólniającego („do góry")- *Pol;* funkcja wyznaczająca praprzodka drzewa dziedziczenia- $Korzen: TOb \rightarrow TOb$. Zbiór *Populacja* oznacza wystąpienia typów obiektowych określonych za pomocą ograniczeń ze zbioru *Ograniczenia*. Pozostałe elementy struktury informacyjnej modelu przedstawiono w rozdziale 3.

3. Własności grafu funkcjonalnego schematu ORM

Graf funkcjonalny [1,9] jest odwzorowaniem izomorficznym struktury informacyjnej. Jest on przedstawiany jako graf dwudzielny skierowany, w którym zbiór wierzchołków *V1* tworzą typy faktów, zbiór wierzchołków *V2* tworzą typy obiektowe, natomiast zbiór krawędzi tworzą role łączące wierzchołki ze zbioru *V1* z wierzchołkami ze zbioru *V2*. Zbiór ról jest utworzony z uwzględnieniem ról dziedziczonych przez relację specjalizacji i polimorfizmu.



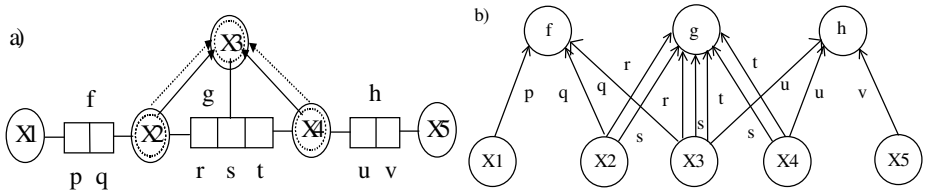
Rys. 1. a) Relacja *Spec*; .b), c) grafy funkcjonalne diagramu z rysunku 2a

Taki sposób reprezentowania typów faktów umożliwia przedstawienie złożonych schematów wyrażeń relacyjnych realizowanych za pomocą elementów struktury informacyjnej *ORM* oraz przedstawienia własności rodzin typów obiektowych związanych relacjami *Spec* lub *Pol*. W pracy tej oparto się na idei grafów funkcjonalnych przedstawionych w [1]. Na rysunku 1c pokazano, przyjętą w tej pracy, zmodyfikowaną postać grafu funkcjonalnego przykładowego diagramu *ORM* z rysunku 1a, uwidaczniającą wszystkie dostępne role każdego typu obiektowego, jakie wynikają z reguły specjalizacji: *WRole(X2)= {q,r,s}, WRole(X3)= {s}, WRole(X4)= {s,t,u}*. Na rysunku 1b przedstawiono graf funkcjonalny diagramu *ORM* z rysunku 1a, zaprezentowany w [1], który sprowadza cechy elementów drzewa dziedziczenia do korzenia *X3*.

Na rysunku 2a przedstawiono przykład diagramu *ORM* zawierający relację *Pol*. Wykorzystano diagram z rysunku 2a., zastępując relację *Spec* relacją *Pol* ze specjalizacją oraz zamieniając typy *X2, X3, X4* na typy polimorficzne. Graf funkcjonalny z rysunku 2b prezentuje wszystkie role osiągane przez każdego z typów obiektowych wynikające z relacji *Pol*

połączonej z relacją *Spec*. Z reguł specjalizacji i relacji polimorfizmu wynika

PolRole(X2)={q, r, s}, PolRole(X3)={q, r, s, t, u}, PolRole(X4)={s, t, u}.



Rys. 2 a) Relacja *Pol*; b) graf funkcjonalny diagramu z rysunku 2a

4. Metryki spójności grafu funkcjonalnego schematu ORM

W pracy badane są grafy funkcjonalne modeli *ORM* [1,9]. W wyznaczaniu spójności przyjęto w pracy metodę badania grafu, rozróżniającą przypadek jednej składowej spójności grafu i przypadek kilku składowych. Do wyznaczenia liczby składowych grafu zastosowano algorytm jednokrotnego odwiedzania jego wierzchołków ze zbiorów *V1* i *V2* w kolejności wynikającej z nierosnącej przyległości wierzchołków z *V2*. Liczba składowych spójności jest równa liczbie wierzchołków ze zbioru *V1*, które nie mają odwiedzonych w poprzednim krokach przyległych wierzchołków ze zbioru *V2*.

Metryka *Spójność_M* określa spójność modelu ORM na podstawie zależności wiążącej liczbę ról całego modelu z liczbą typów faktów i pozostałych typów obiektowych występujących w tym modelu. Jest to zależność

$$Spójność_M = \varphi\,(TFakt,\ TOb\ -TFakt,\ Role_M),$$

gdzie zbiór *Role_M* jest wyznaczony jako suma zbiorów, czyli

$$Role_M(z)=\cup_z PolRole(z) \cup \ \cup_z WRole(z).$$

Wzór ten jest prawdziwy dla każdego przypadku, gdyż zbiory ról *PolRole(x)* są zbiorami pustymi w przypadku braku polimorfizmu, natomiast zbiór *WRole* w przypadku braku specjalizacji traktuje typ obiektowy jako *Korzen* drzewa dziedziczenia i zbiór *WRole* zawiera role tego typu. Metryka *Spójność_M* jest zdefiniowana jako

$$Spójnosc_M = \frac{LW1,\ gdy\ graf\ \ funkcjonalny\ \ jest\ spójny}{LW2,\ gdy\ graf\ \ funkcjonalny\ \ jest\ niespójny}$$

W tabeli na rysunku 3 podano wartości metryki *LW1* dla grafów spójnych oraz *LW2* dla grafów niespójnych wraz z warunkami granicznymi wynikającymi ze specyfiki grafów funkcjonalnych. Wyrażenia *LW1* i *LW2* wyprowadzono na podstawie założenia, że graf maksymalnie spójny ma wartość metryki *LW1=X/Y*, graf minimalnie spójny ma *LW1=0.* Największa wartość metryki *LW2* jest *LW2=0* dla grafu minimalnie niespójnego (2 składowe spójności –w tym jedna składowa maksymalnie spójna), natomiast graf zawierający liczbę składowych spójnych równą liczbie typów faktów ma wartość *LW2= -1*.

W wyrażeniach parametr *a* jest równy mocy zbioru *Role_M(x)* - *a=|Role_M(z)|.* Parametry: *X* jest równy liczbie typów faktów- *X=|TFakt|*, *Y* jest równy liczbie typów obiektowych- *Y=|TOb-TFakt|*, *Z* jest równy liczbie obiektowych typów rekurencyjnych, natomiast *k* jest liczbą składowych spójnych grafu funkcjonalnego.

Dla grafów zawierających jeden typ obiektowy (*Y=1*) i jeden typ faktu (*X=1*), lub *X=1* i *Y >1* lub *Y=1* i *X>1*, przy założeniu spójności grafu (*k=1*) przyjęto *LW1=0,* natomiast dla grafów niespójnych przyjęto *LW2=-1* dla *Y=1* i *X>1, k=2* lub *X=2* i *Y > 1, k=2*. Wybrano skalę względną, gdzie grafy spójne mają dodatnie wartości metryk *LW1*, natomiast grafy niespójne mają wartości ujemne, wyznaczane za pomocą metryki *LW2*. Wartości wyrażeń *LW1* i *LW2* zawsze rosną wraz ze wzrostem liczby krawędzi (parametr *a*) w grafie przy spełnieniu następujących warunków dla *LW1*: *Y*>1, *X*>1 oraz *LW2: X*>2, *Y>1*. Przy stałej liczbie typów faktów i typów bazowych wzrost wartości parametru *a* oznacza wynik działania relacji *Spec* i *Pol* lub typów faktów składających się z wielu ról, wzrost liczby składowych spójnych *k* zmniejsza wartość metryki *LW2* przy zachowaniu bez zmian parametrów *X* i *Y* tej metryki.

Dane					*LW1*	*LW2*	*a (LW1)*		*a (LW2)*	
							≥	≤	≥	≤
X	*Y*	*Z*	*a*	*k*	$\frac{X}{Y} * \frac{a-X-Y-Z+1}{X*Y-X-Y+1}$	$\frac{a-X*Y+X+Y-Z-k}{X*Y-X-2*Y+k}$	*X+Y-1*	*X∗Y*	*Y*	*X*Y-(X+Y-1)+1*
5	5	0	6	4	-	-0.928	-	-	5	17
8	8	0	16	1	0,02	-	15	64	-	-

Rys. 3. Metryki grafów spójnych *LW1* oraz niespójnych *LW2* - przykłady danych

5. Formalna ocena wpływu spójności klas na odporność na błędy oprogramowania

W metodzie *ORM* model klasy w postaci grafu dwudzielnego został wykorzystany do badania wpływu spójności klasy na odporność na błędy [3,4,7]-dalsze rozważania dotyczą również modelu całego oprogramowania. Określono *współczynnik W utraty informacji* związany z przepływem informacji w klasie. W przypadku braku błędów w klasie liczba S dróg informacyjnych jest równa liczbie krawędzi grafu czyli liczbie *a* ról występujących w typach faktów *X* wiążących typy obiektowe *Y*. W przypadku silnie spójnej klasy liczba ścieżek S_{max} jest równa

$$S_{\max} = X * Y,$$

natomiast dla klasy minimalnie spójnej liczba ścieżek S_{min} wynosi

$$S_{\min} = X + Y - 1.$$

Zakładając, że w klasie występuje *m* błędnych typów faktów, *o* błędnych typów obiektowych oraz *p* błędnych typów ról, wówczas liczba ścieżek jest funkcją elementów modelu klasy pozbawionych błędów, czyli

$$S = S(X - m, Y - o, a - p).$$

W danej klasie liczba wszystkich ścieżek jest maksymalna, kiedy nie występują błędy, natomiast maleje wraz ze wzrostem błędów występujących w typach obiektowych *Y,* typach faktów *X* i rolach *a* i osiąga wartość równą 0, gdy wszystkie elementy klasy są błędne. Wpływ błędów klasy na zmniejszanie się liczby ścieżek można wyprowadzić z następującego wyrażenia, czyli *współczynnika utraty informacji W.*

$$W = \frac{S(X - m, Y - o, a - p)}{S(X, Y, a)}.$$

Współczynnik *W* dla klasy maksymalnie spójnej jest równy

$$W_{\max} = \frac{(X - m) * (Y - o) - p}{X * Y},$$

natomiast dla klasy minimalnie spójnej wynosi

$$W_{\min} = \frac{X - m + Y - o - 1 - p}{X + Y - 1}$$

Przy założeniu, że błędy *o, m* i *p* są zmiennymi losowymi, a prawdopodobieństwa ich wystąpień są równe odpowiednio *pm(m), po(o)* oraz *pp(p)*, współczynnik *W* jest również zmienną losową *RW(t),* czyli

$$RW(t) = \sum_{o=0}^{X} \sum_{m=0}^{Y} \sum_{p=0}^{a} pm(m) * po(o) * pp(p) * W(X - m, Y - o, a - p)$$

Zakładając, że prawdopodobieństwa wystąpienia błędów w elementach klasy są małe w porównaniu z liczbą typów faktów *X* klasy można przyjąć, że mają one rozkład Poissona. W przypadku prawdopodobieństwa *pm(m)* mamy

$$pm(m) = \frac{(-\lambda_m t)^m}{m!} * \exp(-\lambda_m t)$$

Podobnie zdefiniowano prawdopodobieństwa *po(o)* oraz *pp(p).* Z definicji współczynnika W_{max} oraz W_{min} wynika, że można rozpatrywać niezależnie wpływ uszkodzeń poszczególnych elementów klasy, stąd współczynniki RW_{max} i RW_{min} są równe

$$RW_{\max}(t) = \frac{1}{X * Y}\left(\sum_{m=0}^{X} \sum_{o=0}^{Y} \sum_{p=0}^{(X-m)*(Y-o)} (pm(m) * po(o) * pp(p)((X - m) * (Y - o) - p))\right)$$

$$RW_{\min}(t) = \frac{1}{X+Y-1}\left(\sum_{m=0}^{X} \sum_{o=0}^{Y} \sum_{p=0}^{X-m+Y-o-1} pm'(m) * po'(o) * pp'(p)(X-m+Y-o-1-p)\right).$$

Przy założeniu, że poszczególne prawdopodobieństwa wystąpienia dokładnie *m* błędnych typów faktów, *o* błędnych typów obiektowych oraz *p* błędnych ról mają rozkład Poissona, wyrażenie RW_{max} nie zależy od rozmiaru modelu, natomiast RW_{min} zależy od liczby typów obiektowych oraz typów faktów, czyli

$$RW_{\max}(t) = (1 - \lambda_n t) * (1 - \lambda_o t) * (1 - \lambda_p t),$$

$$RW_{\min} = (1 - \lambda'_p t) * (1 - \frac{X * \lambda'_m t + Y * \lambda'_o t}{X + Y - 1}).$$

Zakłada się, że utrata informacji klasy na skutek błędów jest zależna od spójności-im większa spójność, tym mniejsza utrata informacji. Założenie to wynika z faktu, że silny związek semantyczny między typami obiektowymi powiązanymi rolami z typami faktów pozwala na lepszą weryfikację populacji związanej z daną klasą. Jeśli $\lambda_o = \lambda_m = 0.001$ oraz $\lambda_p = 0.005$, wówczas $RW_{max} = 0.993$, natomiast dla $X=Y=10$ oraz $\lambda'_o=\lambda'_o=0.002$ oraz $\lambda'_a=0.01$ i wtedy $RW_{min}=0.988$. Oznacza to możliwość większej utraty informacji w przypadku modelu minimalnie spójnego przy założeniu, że prawdopodobieństwa wystąpienia każdego błędu jest większe w porównaniu z modelem silnie spójnym.

Rozważając wpływ spójności klasy na odporność na błędy wyznaczono prawdopodobieństwo P_b wystąpienia dokładnie *b* błędów w klasie w wyniku błędów w: typach faktów, typach obiektowych oraz w rolach. Prawdopodobieństwo *Pb* ma rozkład Poissona, czyli

$$P_b = \frac{(\Lambda t)^b}{b!} * \exp(-\Lambda t)$$

natomiast *całkowity współczynnik błędu klasy* Λ jest wyrażeniem wiążącym prawdopodobieństwa wystąpienia błędu: typu faktu λ_m, błędu typu obiektowego λ_o, oraz błędu ról λ_p, czyli

$$\Lambda = \frac{X * \lambda_m + Y * \lambda_o}{LW * a * \lambda_p}$$

Dla klas spójnych mamy

$$\Lambda = \Lambda_s = \frac{X * \lambda_m + Y * \lambda_o}{(LW1 + \frac{X}{Y}) * a * \lambda_p}$$

Dla grafów silnie spójnych, gdzie *LW=LW1=X/Y całkowity współczynnik błędu klasy* wynosi

$$\Lambda_{\max} = \frac{X * \lambda_m + Y * \lambda_o}{(1 + \frac{X}{Y}) * a * \lambda_p},$$

dla grafów minimalnie spójnych, gdzie *LW=LW1=0,* jest równy

$$\Lambda_{\min} = \frac{X * \lambda'_m + Y * \lambda'_o}{\frac{X}{Y} * a * \lambda'_p},$$

natomiast przypadku grafów niespójnych, gdy *LW2>-1* wynosi

$$\Lambda_{ns} = \frac{X * \lambda''_m + Y * \lambda''_o}{(LW2 + 1) * a * \lambda''_p}$$

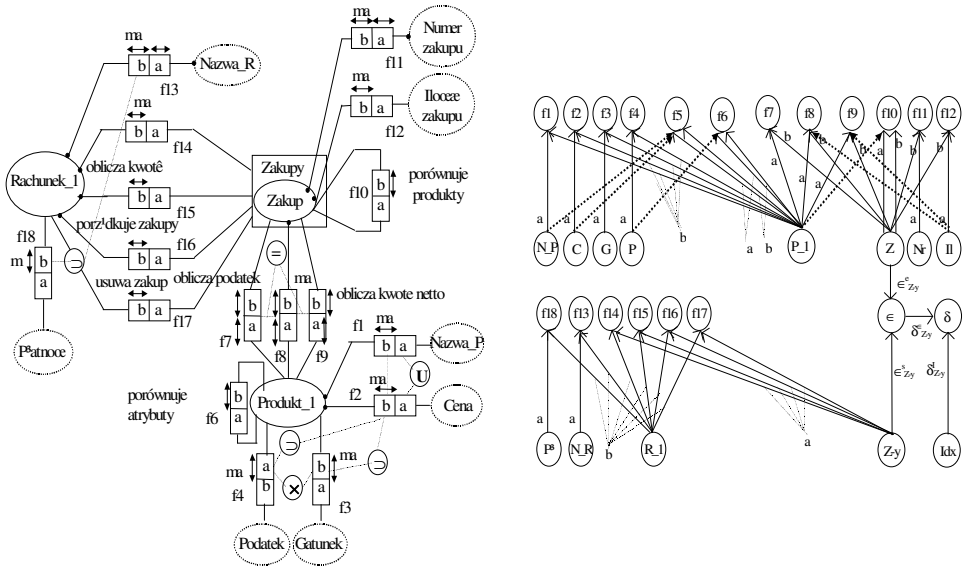
Dla danego *całkowitego współczynnika błędu klasy* Λ prawdopodobieństwo braku ani jednego błędu, czyli *odporność na błędy* R_b jest dane jako

$$R_b = \exp(-\Lambda t)$$

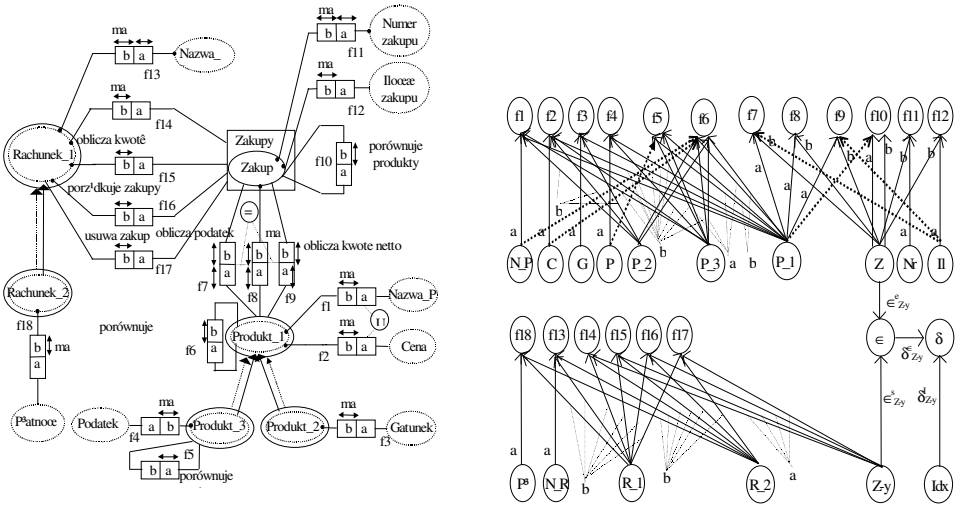
W modelach klas *odporność na błędy* rośnie wraz ze wzrostem spójności klasy (rysunek 4-model 2). Wynika to z założenia, że klasy silnie spójne są semantyczne spójne i każdy z typów faktów jest związany z tymi samymi typami obiektowymi *Y*, co wymusza wielokrotne sprawdzanie ich poprawności podczas analizy obiektowej.

Dane *SI* modeli											
Nr Modelu	*x*	*y*	*z*	*a*	*k*	λ_a	λ_m	λ_p	*LW1*	Λ	R_b
1	20	13	3	46	1	0.001	0.001	0.005	0.074	0.089	0.915
2	***20***	***16***	***6***	***64***	***1***	***0.001***	***0.001***	***0.005***	***0.101***	***0.083***	***0.920***
3	25	21	6	80	1	0.001	0.001	0.005	0.072	0.091	0.913

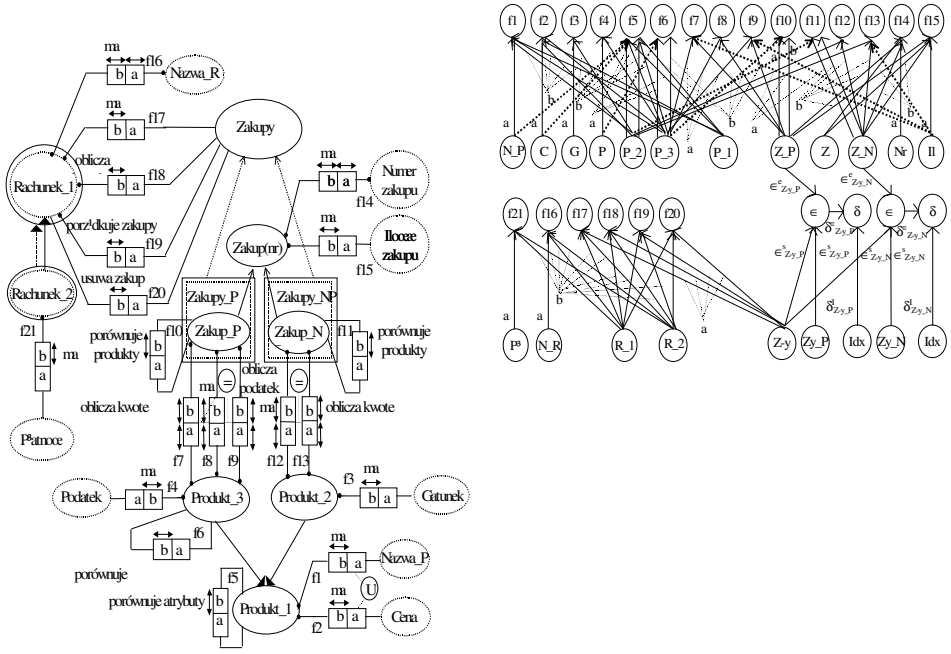
Rys 4. Wartości metryk dla trzech modeli z rys. 6, 7, 8 [8,9]



Rys. 6. a) Model *ORM - M_1*; b) graf fucjonalny *M_1*



Rys. 7. a) Model *ORM - M_2*; b) graf fucjonalny *M_2*



Rys. 8. a) Model *ORM* - *M_3*; b) graf funkcjonalny *M_3*

Zakończenie

Formalna ocena odporności modelu oprogramowania obiektowego na błędy wymaga zweryfikowania za pomocą badań empirycznych. Jednocześnie należy oszacować i zbadać wpływ pozostałych elementów złożoności strukturalnej modeli na odporność na błędy, między innymi liczby połączeń między klasami, metryk związanych z dziedziczeniem i polimorfizmem itd. [3,4,7,9].

Literatura

[1]. P.van Bommel, A.H.M. ter Hofstede, Th.P.van der Weide*: Semantics and verification of object-role models*. Information Systems, vol. 16, No.5, str. 471-495, 1991

[2]. C.Briand, S.Morasca, V.R.Basili, *Defining and Validation Measures for Object Based High-Level Design,* http://www.iese.fhg.de/ISERN/pub/isern_biblio_tech.html

[3]. L.C.Briand, J.W.Daly, V.Porter, J.Wust, *A Comprehensive Empirical Validation of Product Measures for Object-Oriented Systems,* ISERN Report No. ISERN-98-07

[4]. L.C.Briand, W.L.Melo, J.Wust, *Assessing Applicability of Fault-Proneness Models Accross Object-Oriented Software Projects,* ISERN Report No. ISERN-00-06

[5]. P. P. Chen: *Entity-Relation approach to systems analysis and design*, Proceedings of International Conference on Entity-Relation Approach to Systems Analysis and Design, Los Angeles, Decemeber 10-12 1979

[6]. T.A. Halpin, H.A. Proper: *Subtyping and polymorphism in object-role modelling*. Data & Knowledge Engineering 15 (1995) 251-281

[7]. B.Henderson-Sellers: *Object-oriented metrics, measures of complexity*. Prentice Hall PTR, New Jersey 07458, 1996

[8]. Z.Kruczkiewicz: *Metody programowania obiektowego.* Oficyna Wydawnicza Politechniki Wrocławskiej, 1996

[9]. Z.Kruczkiewicz, A.Sielicki, *Metoda oceny charakterystyk wewnętrznych modeli oprogramowania obiektowego,* Materiały XVII Ogólnopolskiej Konferencji Polioptymalizacja i Komputerowe Wspomaganie Projektowania Mielno ’99

[10]. S.R.Schach, *Classical and Object-Oriented Software Engineering with UML and JavaTM,* WCB McGraw-Hill, 1999