Model oceny jakości implementacji wzorców projektowych

mgr inż. Rafał Wojszczyk
Promotor: prof. dr hab. inż. Volodymyr Khadzhynov
Promotor pomocniczy: dr inż. Piotr Ratuszniak

Politechnika Koszalińska

Agenda

1. geneza i problem badawczy,
2. metoda oceny jakości implementacji wzorców projektowych,
3. weryfikacja,
4. wnioski i kierunki dalszych badań.

Geneza – oprogramowanie „pudełkowe”

Geneza – kontrakt pomiędzy wytwórcą a odbiorcą

Wytwórca

Funkcjonalności

Kontrakt

Funkcja 1
Funkcja 2
Funkcja n

Odbiorca

Funkcjonalności

Możliwość dalszej rozbudowy
Niezależne od pracowników
Możliwie łatwe modyfikacje

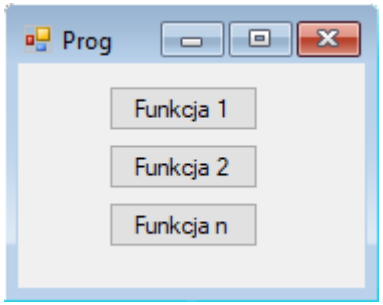
Konkurencyjność
Zmiany w prawie

Wzorce projektowe
Wzorce architektury
Komponenty i biblioteki

Geneza – realizacja kontraktu

1

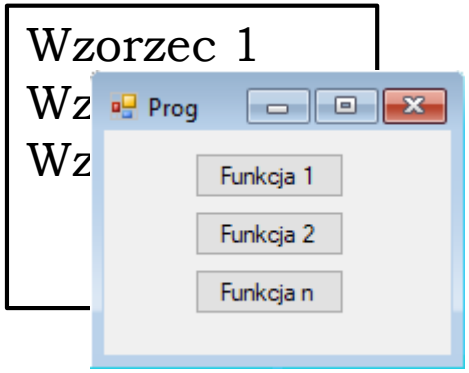
Wytwórca



Odbiorca

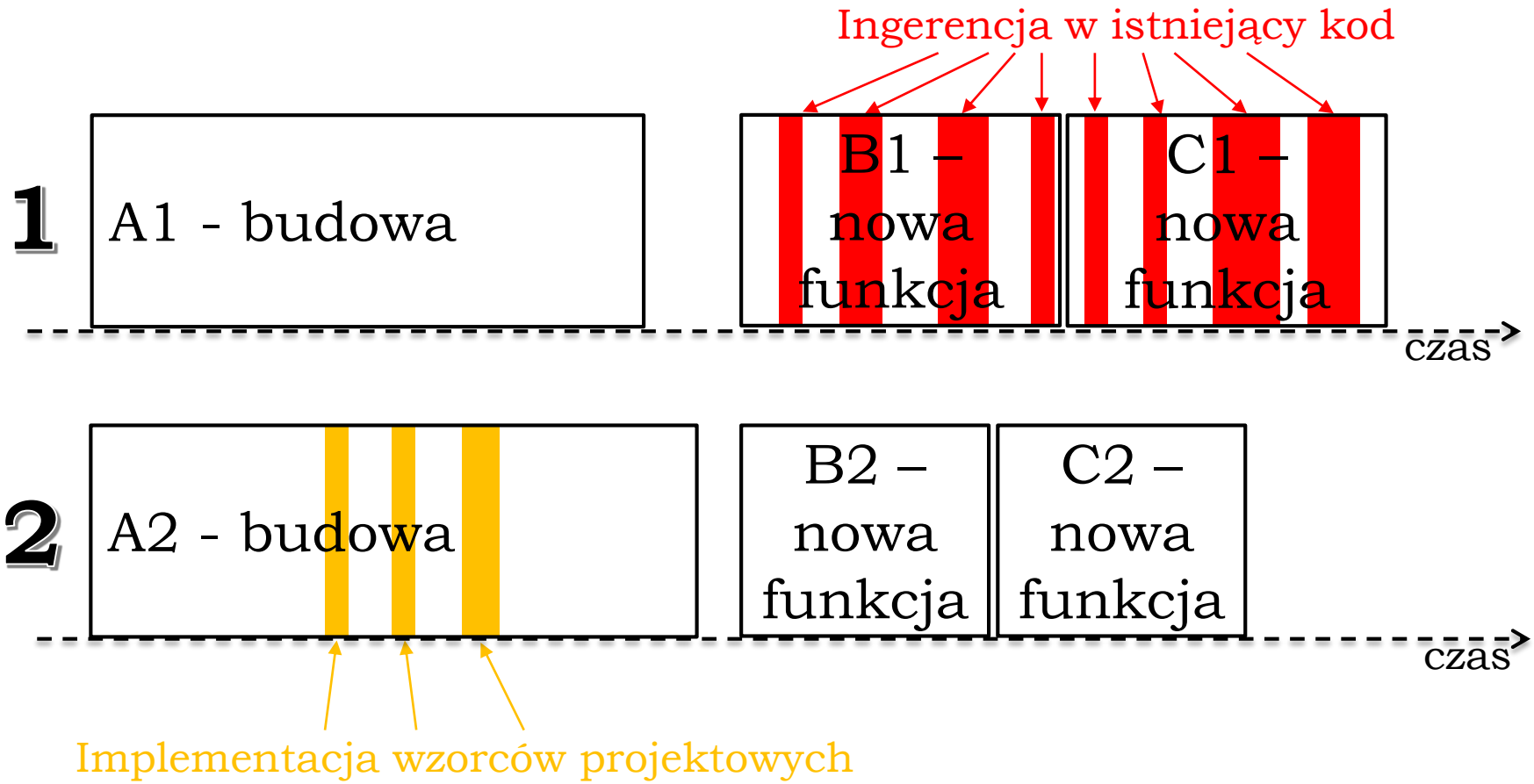
2

Wytwórca



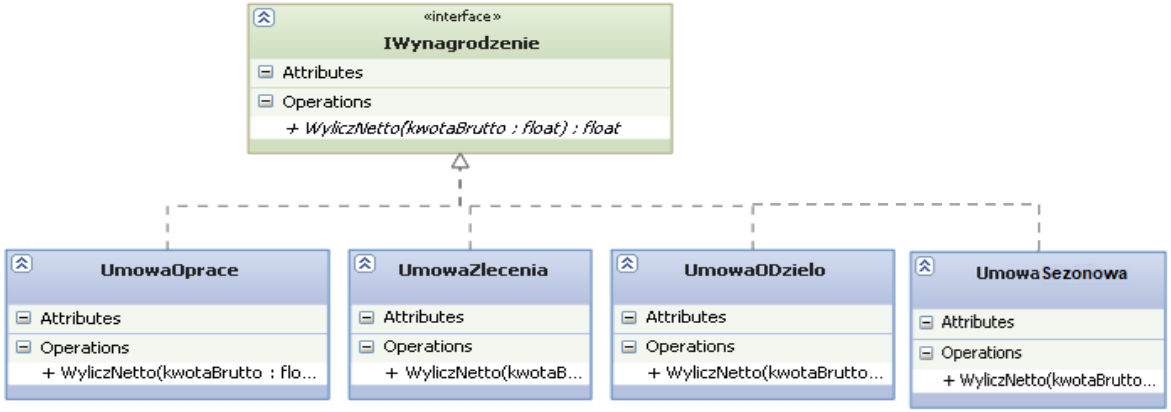
Odbiorca

Geneza – koszt rozbudowy

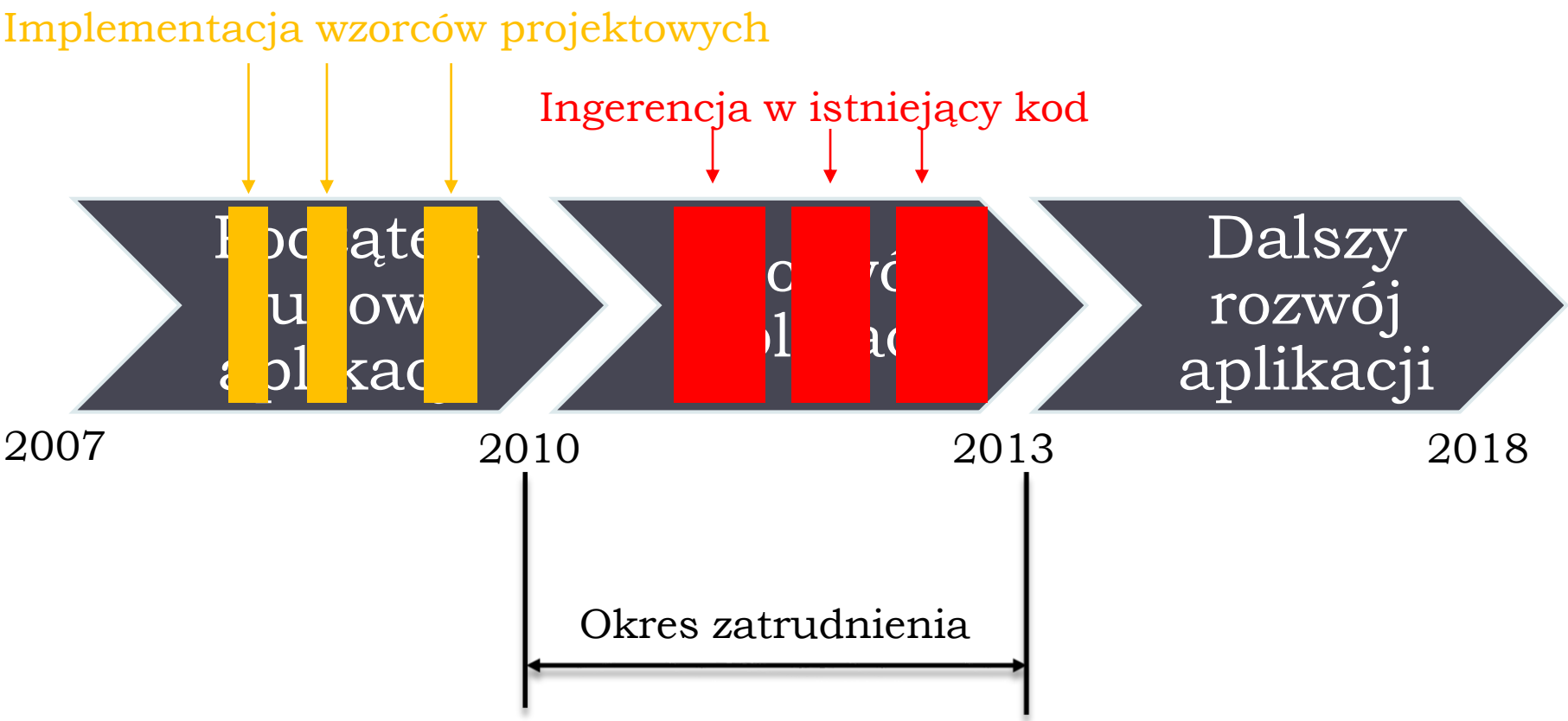


Geneza – wzorzec Strategia

- Pochodzi z katalogu wzorców autorstwa: E. Gamma, R. Helm, R. Johnson, J. Vlissides,
  - Inne popularne wzorce: Singleton, Polecenie, Fabryka
- Cel: zgrupowanie rodziny algorytmów (np. różne rodzaje sortowania) i ich zamienne wykorzystywanie



Geneza – doświadczenie zawodowe



Geneza – zła jakość implementacji wzorców

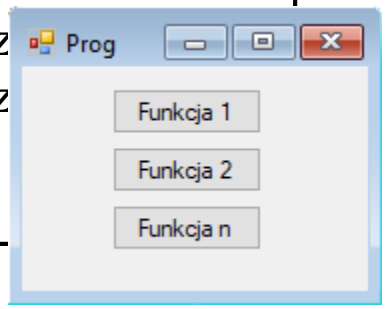
3

Wytwórca

Wzorzec 1

Wz

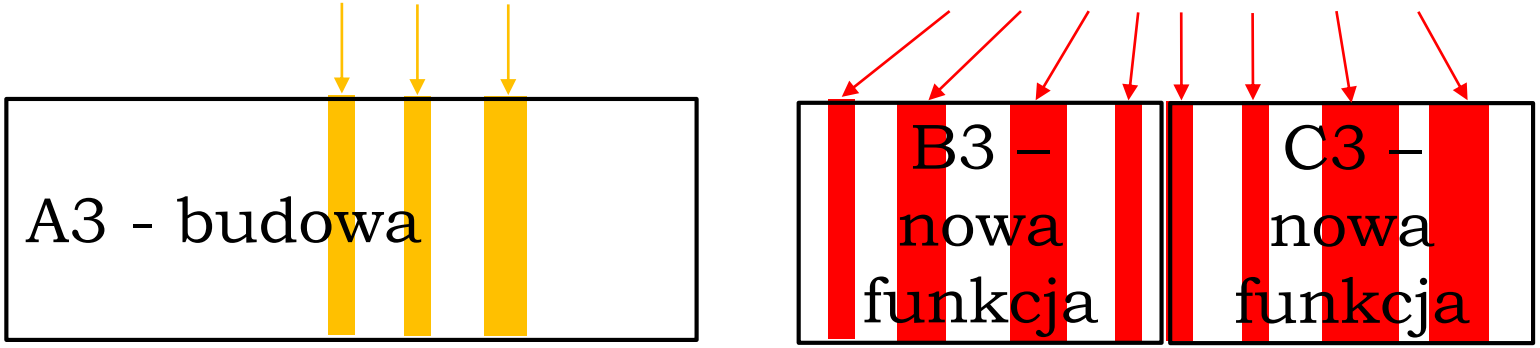
Wz



Odbiorca

Implementacja wzorców projektowych

Ingerencja w istniejący kod

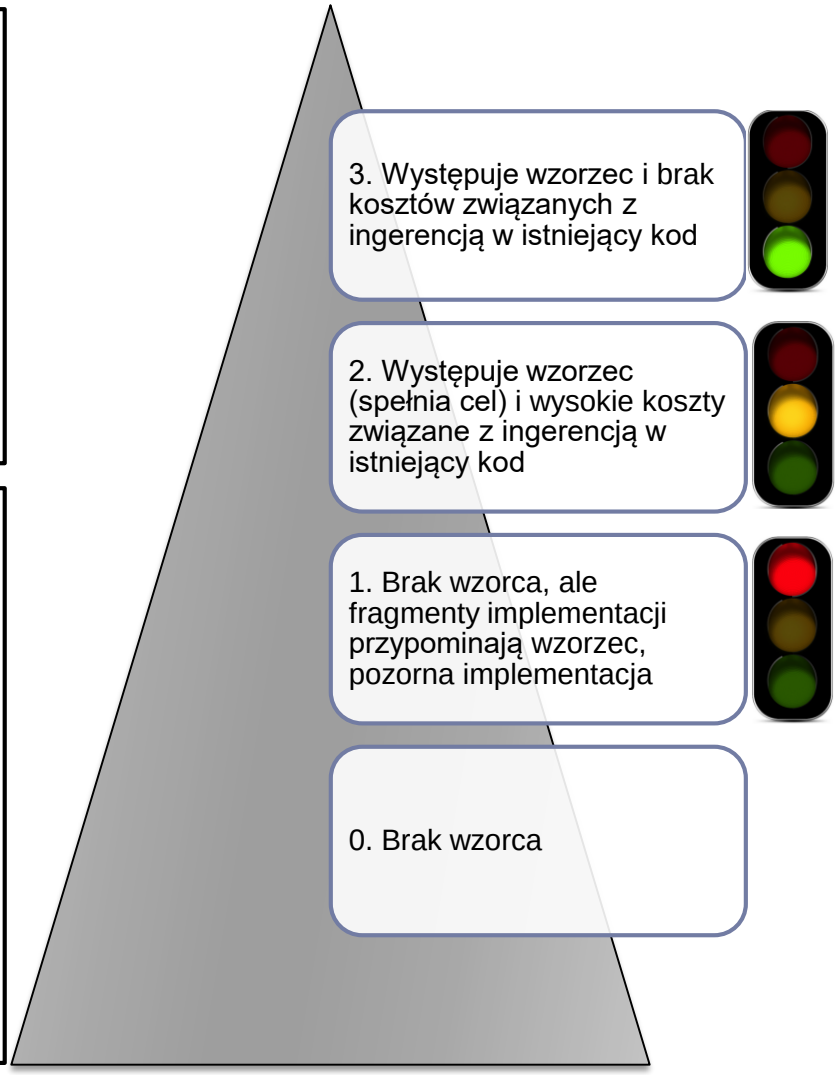


Zła jakość implementacji wzorców

Geneza – definicja jakości

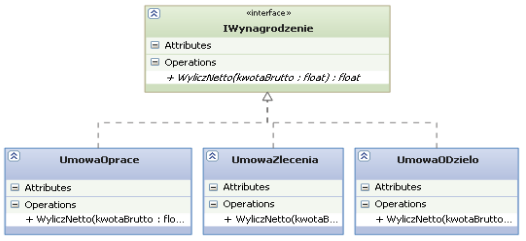
Jakość to stopień, w jakim moduł, system lub proces spełnia określone wymagania, potrzeby lub oczekiwania [IEEE 610]

*Jakość to **stopień**, w jakim implementacja wzorca projektowego (fragment kodu źródłowego) spełnia określone oczekiwania - **redukcja kosztów rozbudowy***



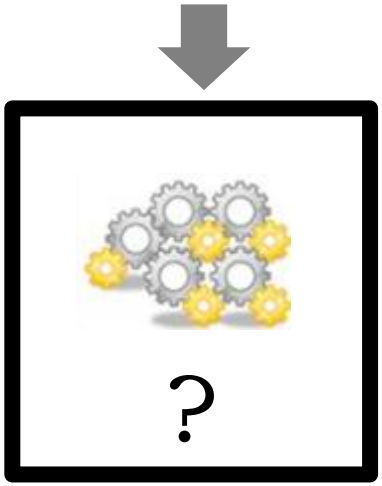
Geneza – poszukiwane rozwiązanie

Cechy wzorców

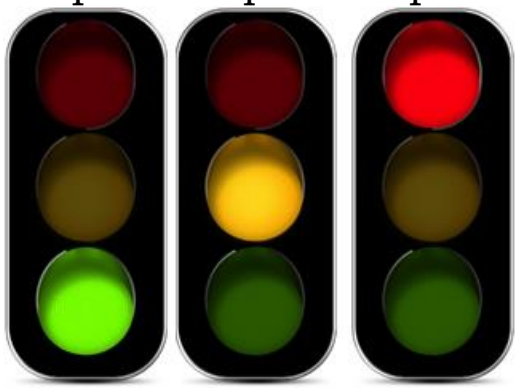


Kod programu

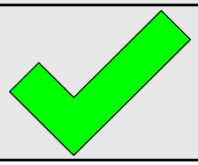
```
namespace Wynagrodzenia
{
    public interface IWynagrodzenie
    {
        float WyliczNetto(float kwotaBrutto);
    }
}

public class UmowaODzielo : IWynagrodzenie
{
    public float WyliczNetto(float kwotaBrutto)
    {
        return kwotaBrutto;
    }
public class UmowaOPrace :IWynagrodzenie
{
    public float WyliczNetto(float kwotaBrutto)
    {
        return kwotaBrutto;
    }
}
public class UmowaZlecenia : IWynagrodzenie
{
    public float WyliczNetto(float kwotaBrutto)
    {
        return kwotaBrutto;
    }
}
```



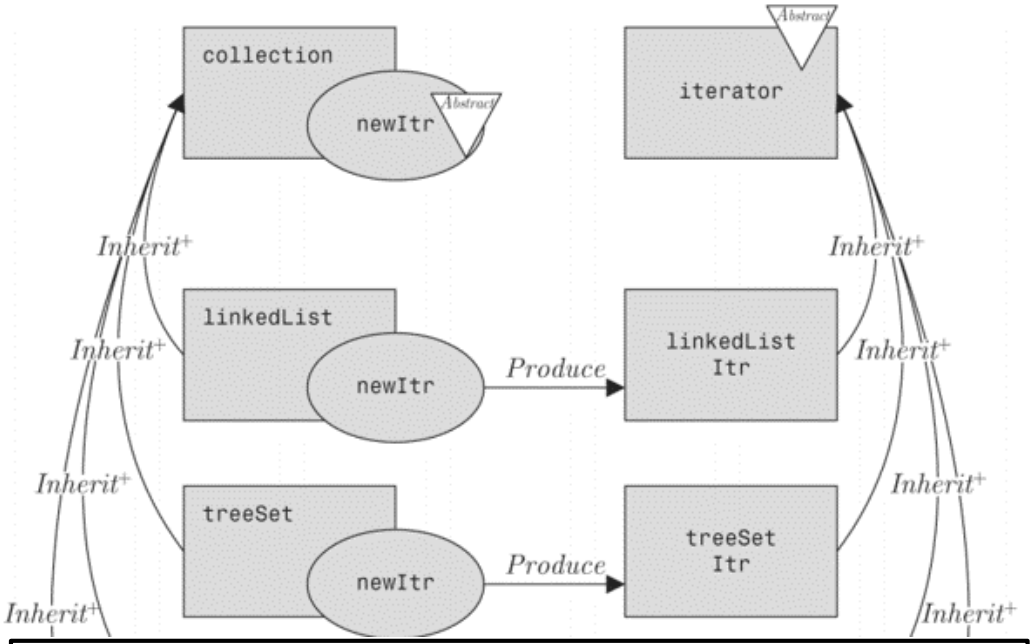
3 stopień 2 stopień 1 stopień



Geneza – alternatywne rozwiązania

Rodzaj	**Przykłady metod**	**Dokładność pozwalająca na ocenę 3 stopnia jakości**	**Koszt użycia dopuszczalny w zwinnych zespołach**
Modele jakości	FRUPS, CUPRIMDA, ISO9126, Bohem		
Metryki oprogramowania	CK, MOOD, R. Martin		
Wyszukiwanie wystąpień	Tsantalis, Kirasić, SinghRao, Deluca, Grzanek, Rasool, Węgrzynowicz		
Weryfikacja kodu	Blewit (Spine)		
Porównywanie do modeli	Nicholson (Lepus2), Mechlitz		

Geneza – przykład metody Lepus2

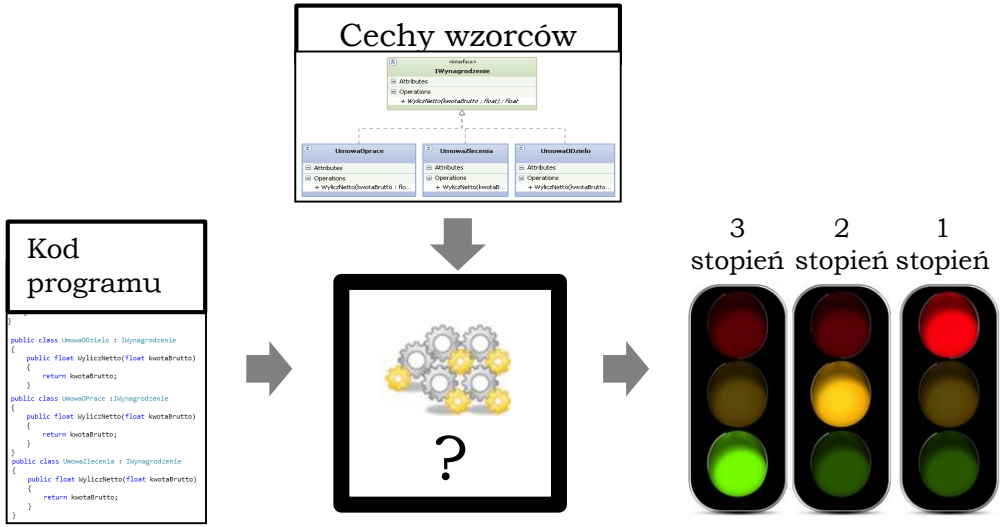


Graficzna reprezentacja wzorca Polecenie zgodna z Lepus2, źródło A. H. Eden: *Codecharts*

- Wymaga obszernej dokumentacji projektowej
- Nie uwzględnia m. in. modyfikatorów dostępu, konwencji nazewnictwa

Geneza - problem

- Dane: kod źródłowy, miejsca występowania wzorców, szablony przykładowych wzorców
- Ograniczenia: paradygmat programowania obiektowego, oprogramowanie pudełkowe, wzorce z katalogu E. Gammy, brak dokumentacji projektowej, koszt użycia metody dopuszczalny w zwinnych zespołach
- Poszukiwana jest metoda oceny jakości implementacji wzorców projektowych?



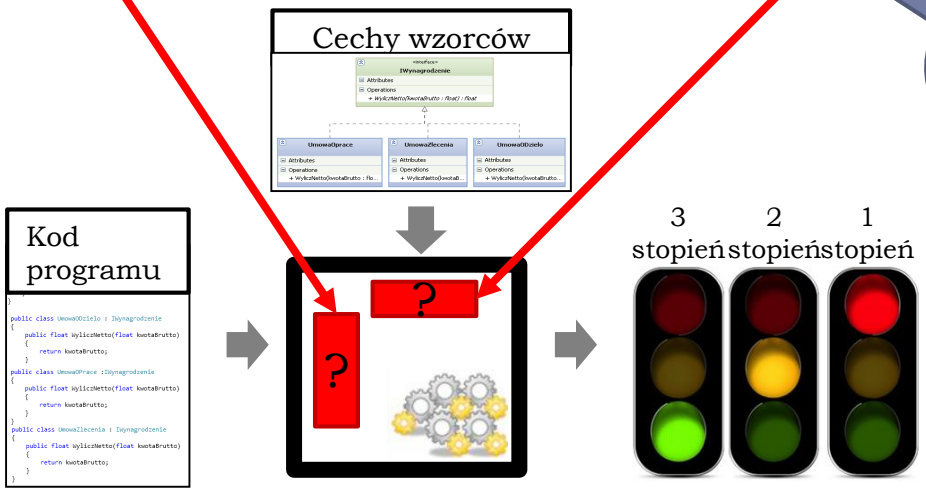
Geneza – istotne cechy kodu i wzorców

Cechy kodu programu	Cechy wzorców projektowych

• Paradygmat programowania obiektowego:
  • rodzaje danych,
  • modyfikatory ogólne,
  • modyfikatory dostępu,
  • elementy składowe,
  • parametry i treść metod,
  • …
	• Paradygmat programowania obiektowego
	• Własności ważne dla programistów:
	  • konwencje nazewnictwa,
	  • warianty wzorców
	• Własności liczbowe:
	  • głębokość drzewa dziedziczenia,
	  • współczynnik ukrycia metod
	  • …

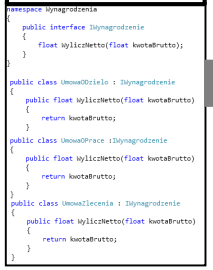
Paradygmat programowania obiektowego

Mieszane reprezentacje

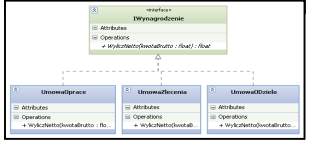


Rozwiązanie – Model oceny jakości implementacji wzorców projektowych

Kod programu



Cechy wzorców



Model programu komputerowego (MP)

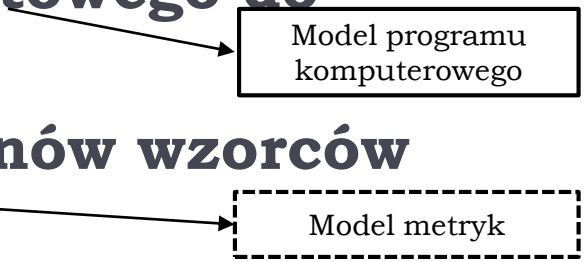
Model wzorca projektowego (MW)

Model metryk (MM)

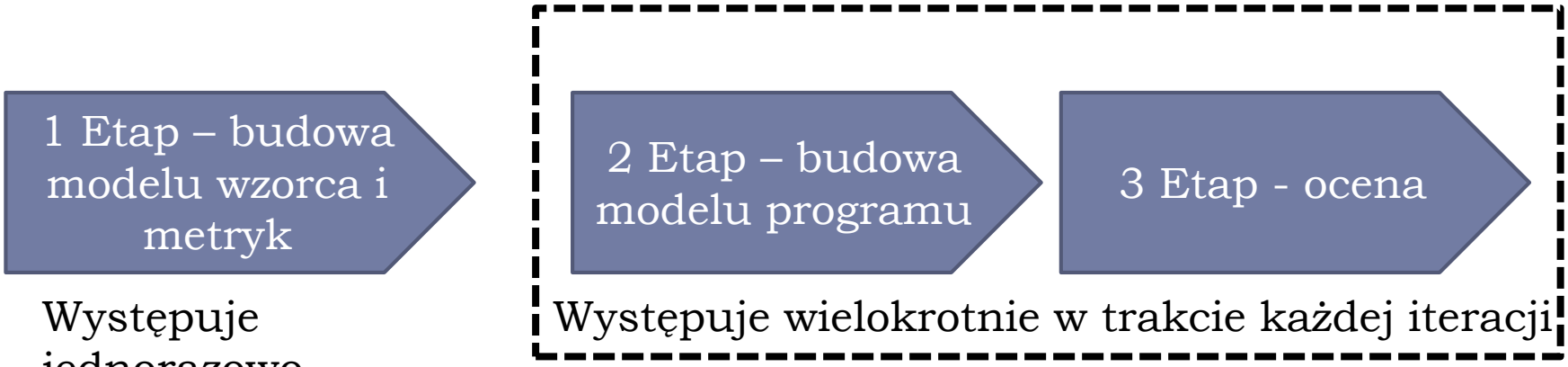
Podział zgodny z ISO 9126

Teza

- **Model wykorzystujący**
  - **paradygmat programowania obiektowego do formalizacji kodu programu oraz**
  - **mieszane reprezentacje do szablonów wzorców projektowych,**
- **umożliwia budowę metody oceny jakości implementacji wzorców projektowych, która pozwala uzyskiwać wyniki oceny**
  - **z lepszą dokładnością i nie większym kosztem niż jest to możliwe przy pomocy alternatywnych metod.**



Metoda


1 Etap – budowa modelu wzorca i metryk
2 Etap – budowa modelu programu
3 Etap - ocena
Występuje jednorazowo
Występuje wielokrotnie w trakcie każdej iteracji

Metoda – Etap 1 budowa modelu wzorca i metryk

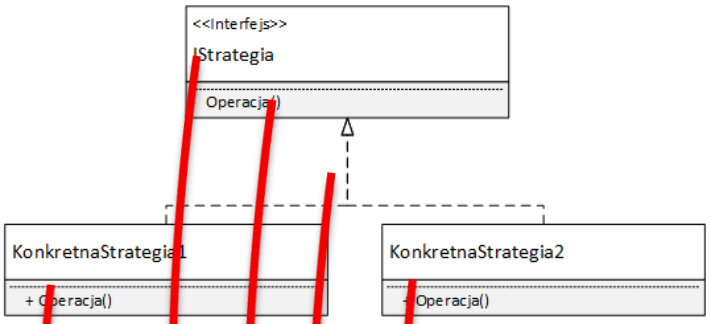
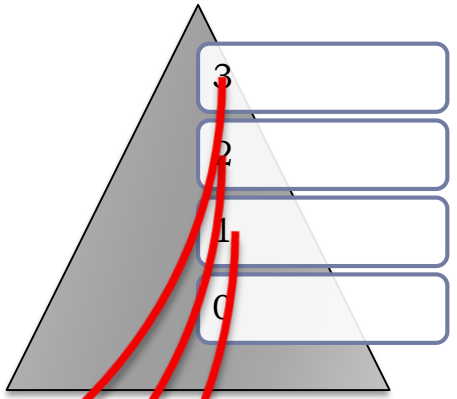


Diagram klas wzorca Strategia

?



Model Wzorca (cechy/charakterystyki):

ch_1 = (deklaracja interfejsu, M_1)
ch_2 = (deklaracja operacji, M_2)
ch_3 = (realizacja interfejsu, M_3)
ch_4 = (implementacja Operacji, M_4)
ch_5 = (kontekst implementacji, M_5)
ch_6 = (inicjalizacja i wybór, M_6)

Model Metryk dla charakterystyka 1 (ch_1):

$$m_{1,1} = \begin{cases} 3\ jeżeli\ (MET, brak, public, \ldots Strategy, interface) \\ \ldots \\ 2\ jeżeli\ (MET, abstract, public, \ldots Strategy, class) \\ \ldots \\ 1\ jeżeli\ (MET, brak, public, \ldots Strategy, class) \end{cases}$$

$$m_{1,2} = \begin{cases} 3\ jeżeli\ |MET| = 1 \\ 2\ jeżeli\ 2 \leq |MET| \leq 10 \\ 1\ jeżeli\ |MET| > 11 \\ 0\ jeżeli\ |MET| = 0 \end{cases}$$

Gdzie: MET – zbiór metod w danym typie

Metoda – Etap 2 budowa modelu programu

```
public abstract class SortStrategy
{
    public abstract void Sort(IList collection);
}
```

t_1=(Ø, Ø, MET_1, abstract, public, SortStrategy, class)

$met_{1,1}$=($PM_{1,1}$, void, 0, $CM_{1,1}$, abstract, public, Sort)

$pm_{1,1,1}$=(Ilist, collection)

```
public class QuickSort : SortStrategy
{
    public override void Sort(IList collection)
    {}
}
```

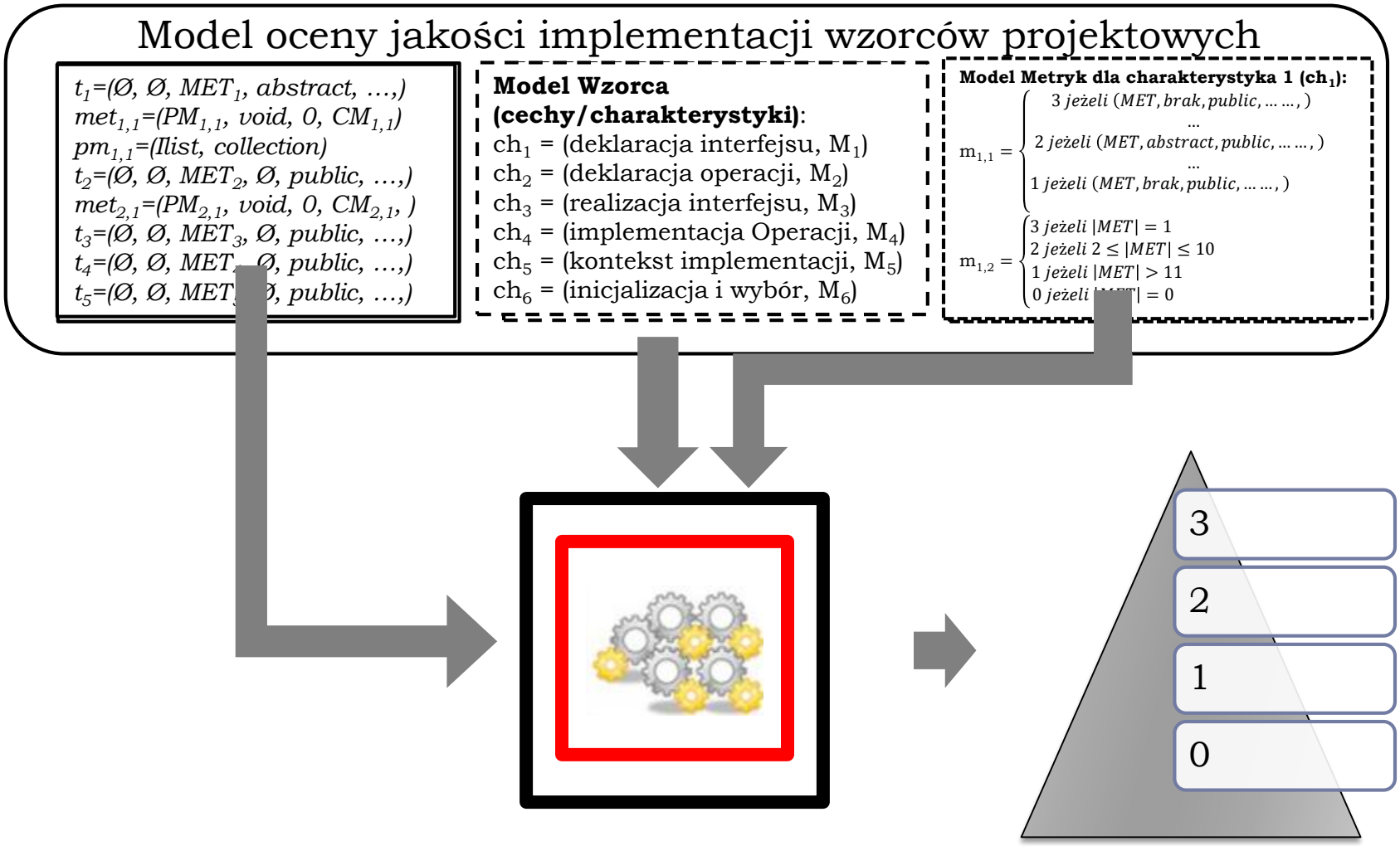
t_2=(Ø, Ø, MET_2, Ø, public, QuickSort, class)

$met_{2,1}$=($PM_{2,1}$, void, 0, $CM_{2,1}$, override, public, Sort)

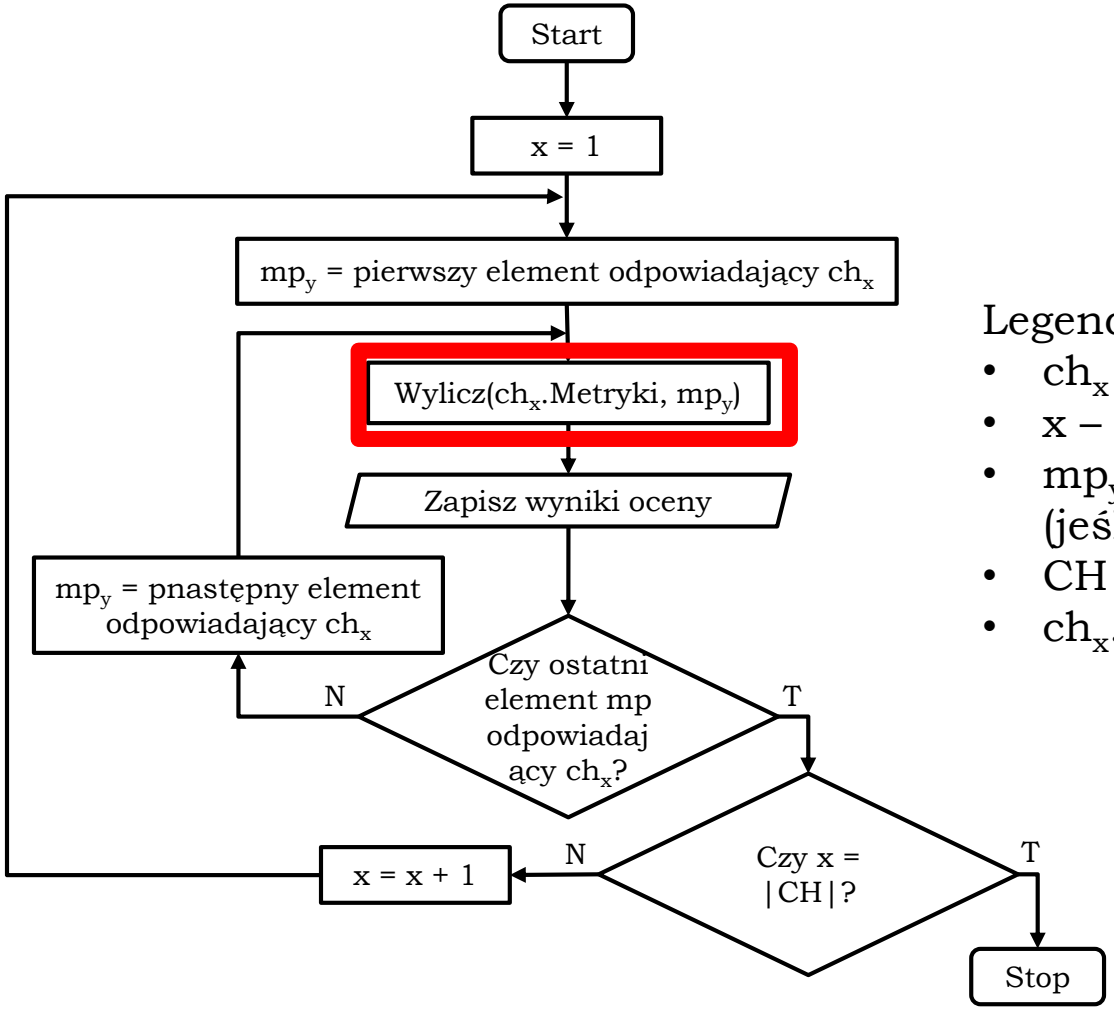
```
public class BubbleSort : SortStrategy
{
    public override void Sort(IList collection)
    {}
}
```

t_3=(Ø, Ø, MET_2, Ø, public, BubbleSort, class)

Metoda – Etap 3 ocena



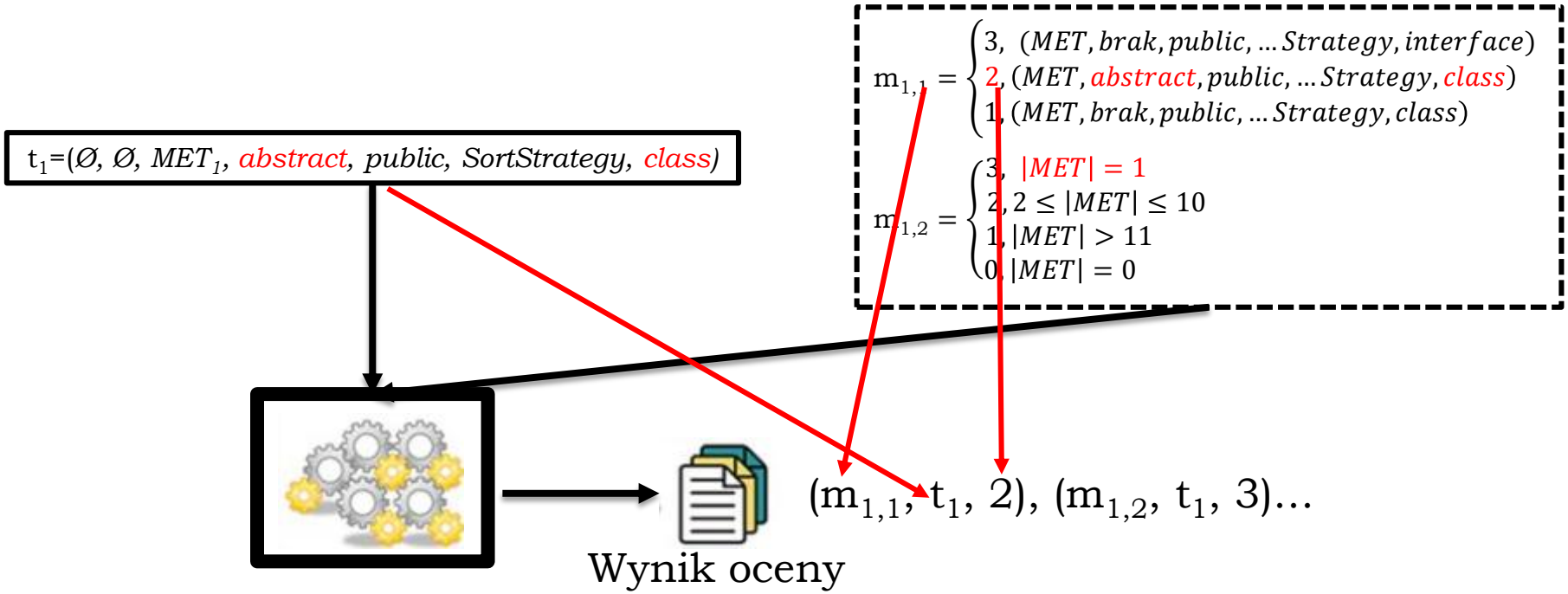
Metoda – algorytm oceny



Legenda:

- ch_x – dowolna charakterystyka
- x – numer charakterystyki
- mp_y – dowolny element z MP (jeśli x=1 to mp_y = rdzeń wzorca)
- CH – zbiór charakterystyk
- ch_x.Metryki – metryki w danej ch

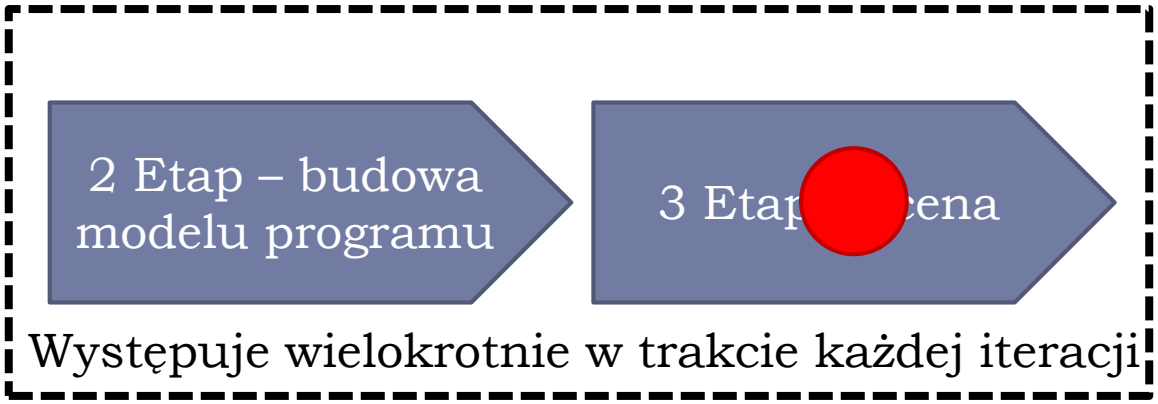
Metoda – wyliczanie wyników

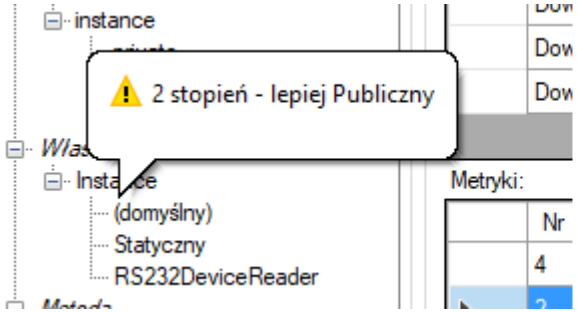


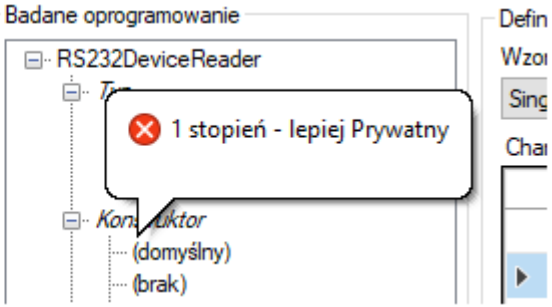
IF *fragment modelu programu* = *fragment modelu metryk* THEN wynik = *stopień jakości*

IF *abstract* = *brak* AND *class* = *interface* THEN 3
IF *abstract* = *abstract* AND *class* = *class* THEN 2

Metoda – poprawa jakości implementacji wzorców







Metoda - podsumowanie

Model oceny jakości implementacji wzorców projektowych

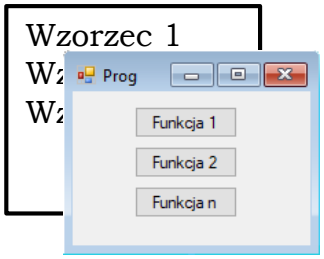
(MP, (MW, MM))

Zbiór metryk MET = {met_p | p = 1, ..., met}
***Metryka met_p** to dwójka postaci: $met_p = g(f_{m,n}(x,y), IWF_{k,l})$, gdzie: g to funkcja interpretująca wynik z funkcji f zgodnie z interpretacją wyników funkcji (IWF)*
*Zbiór **funkcji oceny F** $F = \{f_{m,n}(x,y) \mid m = 1, ..., rf, n=1, ..., if\}$, gdzie: rf to liczba rodzajów, if to liczba funkcji, $f_{m,n}$ n-ta funkcja m-tego rodzaju, x to dowolny byt z MP, y to dowolny zbiór lub wystąpienie z modelu referencyjnego lub funkcji matematycznych*

Weryfikacja – środowisko akademickie

1

Student

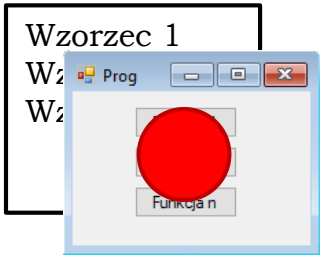


+ Strategia

Student 1 – 3,5 rbh
Student 2 – 5,5 rbh
Student 3 – 7 rbh

2

Student



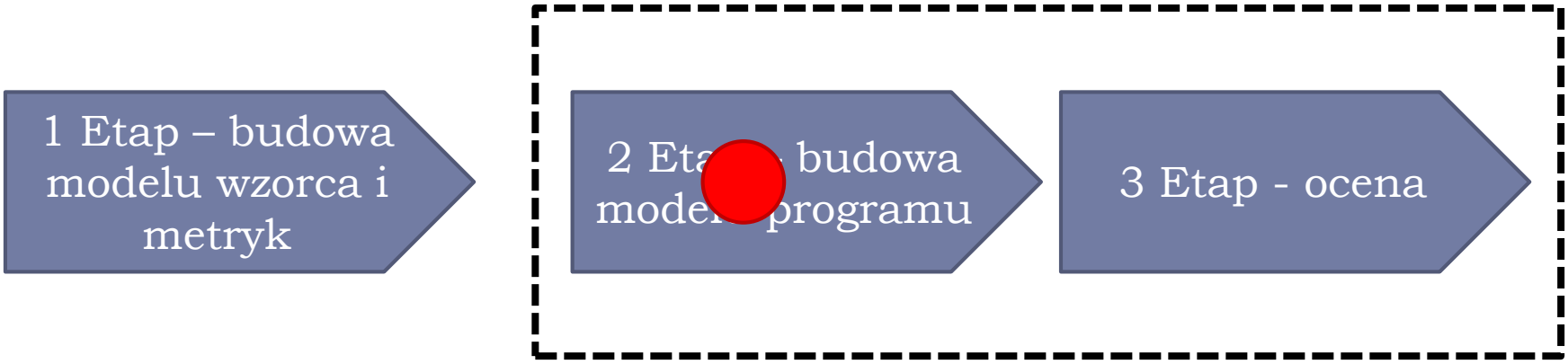
+ Strategia

Student 1 – 2 rbh
Student 2 – 4 rbh
Student 3 – 6 rbh

Zysk

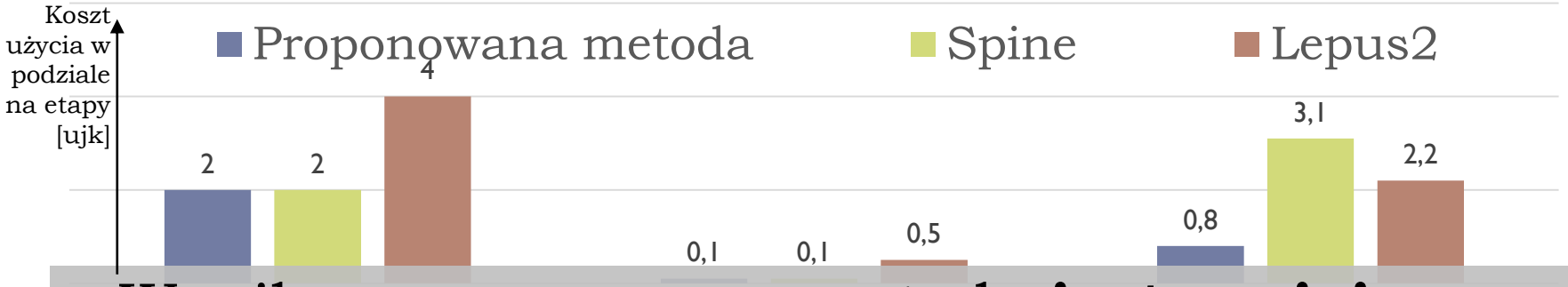
Średnio 28%

Weryfikacja – środowisko komercyjne

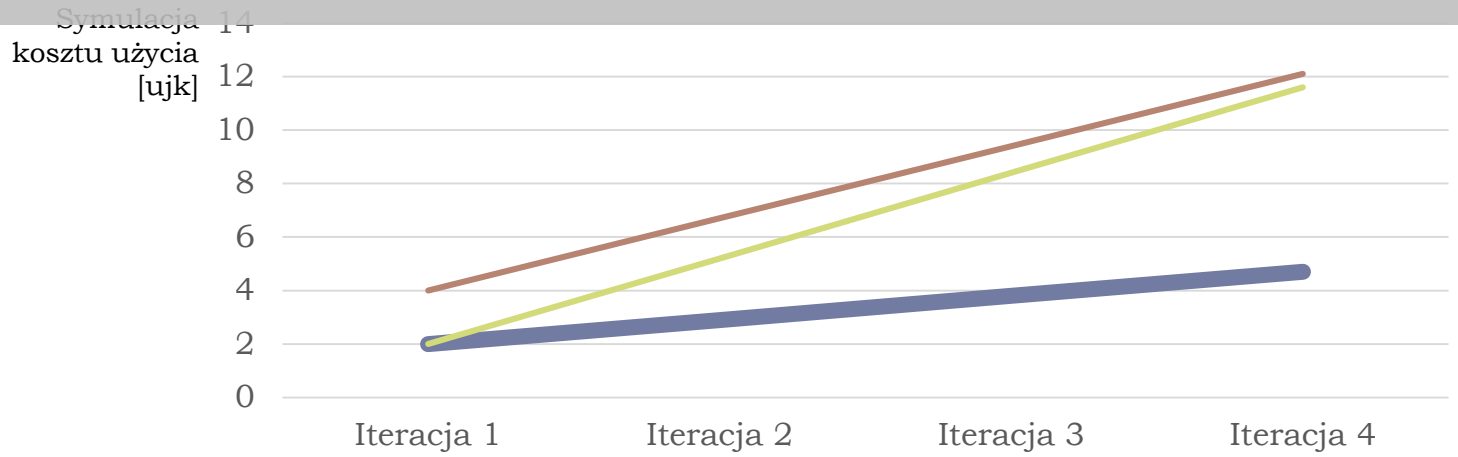


- przy współpracy z Quick-Solutions,
- wykorzystanie metody w trakcie pracy zespołu programistów,
- wzorce: Polecenie, Fabryka, Singleton.
- **Oszczędność 14 rbh** (20 rbh bez użycia metody, 6 rbh po użyciu metody),
- Oszczędność około 20% czasu pracy jednego pracownika w jednej iteracji.

Weryfikacja – koszt użycia

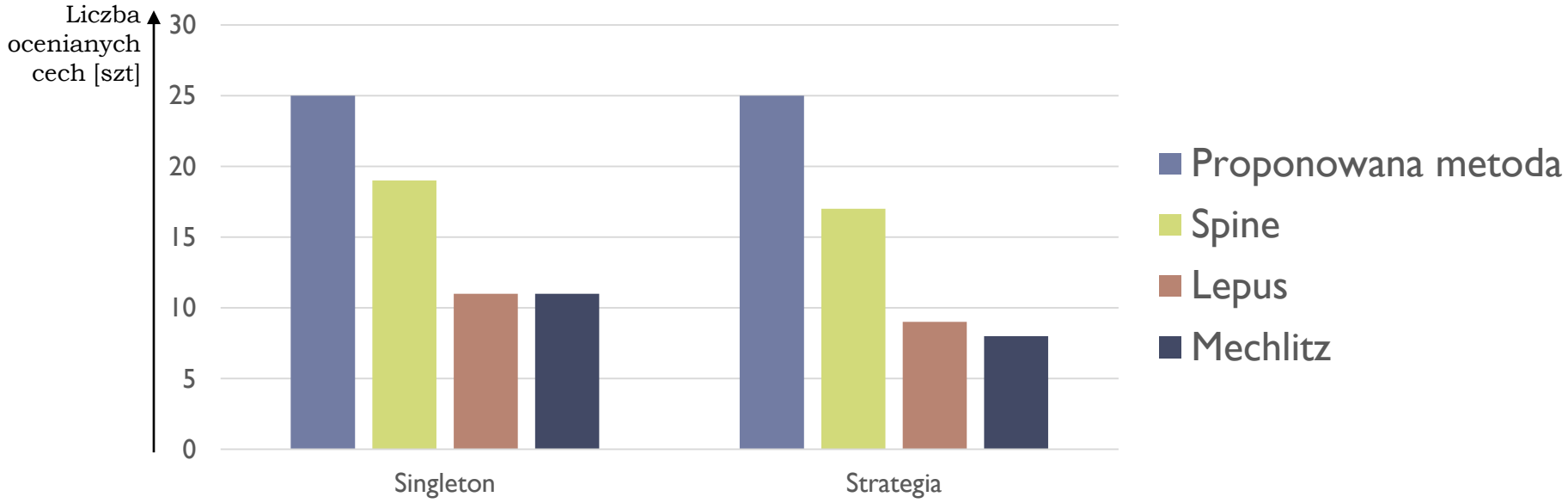


- Wynik – proponowana metoda **jest mniej kosztowna** w użyciu:
  - **45 – 57 %** - sumaryczne zestawienie kosztów,
  - **60 – 61 %** - symulacja użycia przez jeden rok.



Weryfikacja - dokładność

- Analiza porównawcza wybranych metod na przykładzie wzorców Singleton i Strategia.
- Wynik – metoda **jest dokładniejsza**:
  - **25-57 %** w przypadku wzorca Singleton,
  - **30-68 %** w przypadku wzorca Strategia.



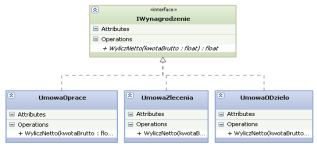
Wnioski

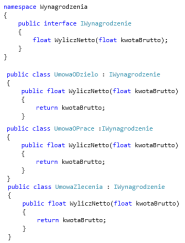
- Obniżony koszt rozbudowy oprogramowania o 20%
- Dokładność większa o 25%
- Koszt użycia niższy o 45%

Wyniki badań

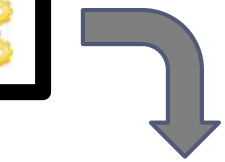
1. opracowano model dla wybranych wzorców projektowych oraz procedurę budowy nowych modeli,
2. pracowano nową metodę oceny jakości implementacji wzorców projektowych,
3. pozytywnie zweryfikowano metodę:
   - zweryfikowano możliwość wykorzystania przez zwinny zespół wytwórczy,
   - przynosi oszczędność w pracach programistycznych.

Niedostatki

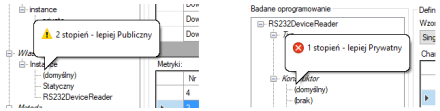




```
namespace Wynagrodzenia
{
    public interface IWynagrodzenie
    {
        float WyliczNetto(float kwotaBrutto);
    }

public class UmowaODzielo : IWynagrodzenie
{
    public float WyliczNetto(float kwotaBrutto)
    {
        return kwotaBrutto;
    }
public class UmowaOPrace :IWynagrodzenie
{
    public float WyliczNetto(float kwotaBrutto)
    {
        return kwotaBrutto;
    }
}
public class UmowaZlecenia : IWynagrodzenie
{
    public float WyliczNetto(float kwotaBrutto)
    {
        return kwotaBrutto;
    }
}
```



$(m_{I,1}, t_I, 2), (m_{I,2}, t_I, 3)\ldots$



- Zastosowanie wyłącznie do programów obiektowych
- Koszt 1 etapu – budowa modelu wzorca i metryk

Kierunki dalszych badań

- Ewaluacja:
  - nowe wzorce,
  - inna klasa oprogramowania,
  - automatyzacja metody i komercjalizacja,
  - inne kryteria oceny,
- rozszerzenie modelu o dodatkowe reprezentacje:
  - umożliwiające dynamiczną analizę oprogramowania (program w trakcie działania),
  - oparte na ontologiach,
  - rozmycie metryk,
- ekstrakcja wiedzy ukrytej z dobrych praktyk.

Wykaz publikacji 1/4

- **Rafał Wojszczyk**, Piotr Stola, *Method of quality assessment of the implementation of design patterns used in production*, w: Advances in Intelligent Systems and Computing, **Springer** International Publishing, 2018/2019 – przyjęte do druku.
- **Rafał Wojszczyk**, *The Experiment with Quality Assessment Method Based on Strategy Design Pattern Example*, w: Advances in Intelligent Systems and Computing, Vol. 656, strony od 103 do 112, **Springer** International Publishing, 2018.
- **Rafał Wojszczyk**, *Quality Assessment of Implementation of Strategy Design Pattern,* w: Advances in Intelligent Systems and Computing, Vol. 620, strony od 37 do 44, **Springer** International Publishing, 2018.
- **Rafał Wojszczyk**, Włodzimierz Khadzhynov, *The Process of Verifying the Implementation of Design Patterns - Used Data Models*, w: Advances in Intelligent Systems and Computing, Vol. 521, strony od 103 do 116, **Springer** International Publishing, 2017.

Wykaz publikacji 2/4

- **Rafał Wojszczyk**, Piotr Ratuszniak, *Trudności w implementacji wzorców projektowych w małych zespołach programistycznych*, w: Przedsiębiorstwo we współczesnej gospodarce – teoria i praktyka, no. 2/2017, strony od 189 do 201, Wydawnictwo Politechniki Gdańskiej, ISSN 2084-6495 Gdańsk 2017.
- Daniel Czyczyn-Egird, **Rafał Wojszczyk**, *Determining the popularity of design patterns used by programmers based on the analysis of questions and answers on stackoverflow.com Social Network*, w: Communications in Computer and Information Science, vol. 608, strony od 421 do 433, **Springer** International Publishing, 2016.
- **Rafał Wojszczyk**, Robert Wójcik, *The Model of Quality Assessment of Implementation of Design Patterns*, w: Advances in Intelligent Systems and Computing, vol. 474, strony od 515 do 524, **Springer** International Publishing, 2016.
- **Rafał Wojszczyk**, Włodzimierz Khadzhynov, *Wykorzystanie modeli danych do weryfikacji implementacji wzorców projektowych*, w: Zeszyty Naukowe Wydziału Elektroniki i Informatyki nr 10, strony od 193 do 209, Wydawnictwo Uczelniane Politechniki Koszalińskiej, ISSN 1897-7421, Koszalin 2016.

Wykaz publikacji 3/4

- Daniel Czyczyn-Egird, **Rafał Wojszczyk**, *Zastosowanie technik eksploracji danych na przykładzie badania popularności wzorców projektowych w serwisie społecznościowym Stackoverflow.com*, w: Zeszyty Naukowe Wydziału Elektroniki i Informatyki nr 10, strony od 81 do 94, Wydawnictwo Uczelniane Politechniki Koszalińskiej, ISSN 1897-7421, Koszalin 2016.
- **Rafał Wojszczyk**, Włodzimierz Khadzhynov, *Data Models in the Verification of the Singleton Pattern*, w: Journal of Theoretical and Applied Computer Science, vol. 9, no. 3/2015, Szczecin 2016.
- **Rafał Wojszczyk**, *Model oceny jakości implementacji wzorców projektowych w oprogramowaniu*, w: Innowacje w zarządzaniu i inżynierii produkcji, Tom II, strony od 300 do 309, Oficyna Wydawnicza Polskiego Towarzystwa Zarządzania Produkcją, Opole 2016.
- **Rafał Wojszczyk**, *The model and function of quality assessment of implementation of design patterns*, w: Applied Computer Science, Vol. 11, No. 3, strony od 45 do 56, Politechnika Lubelska, ISSN 1895-3735, Lublin 2015.

Wykaz publikacji 4/4

- **Rafał Wojszczyk**, *Weryfikacja poprawności implementacji struktury wzorców projektowych w oparciu o model referencyjny*, w: Od procesów do oprogramowania: badania i praktyka, strony od 73 do 83, Zeszyty Rady Naukowej Polskiego Towarzystwa Informatycznego, ISBN 978-83-60810-73-6, Warszawa 2015.
- **Rafał Wojszczyk**, *Koncepcja hybrydowej metody do oceny jakości zaimplementowanych wzorców projektowych*, w: Zeszyty Naukowe Wydziału Elektroniki i Informatyki nr 7, strony od 17 do 26, Wydawnictwo Uczelniane Politechniki Koszalińskiej, ISSN 1897-7421, Koszalin 2015.
- **Rafał Wojszczyk**, *Pozyskiwanie struktury obiektowej z kodu zarządzanego przy wykorzystaniu metod inżynierii odwrotnej*, w: Inżynieria oprogramowania: badania i praktyka, strony od 199 do 213, Zeszyty Rady Naukowej Polskiego Towarzystwa Informatycznego, ISBN 978-83-63919-15-3, Warszawa 2014.
- **Rafał Wojszczyk**, *Porównanie sposobów reprezentacji wzorców projektowych*, w: Modele inżynierii teleinformatyki 9, strony od 133 do 145, Wydawnictwo Uczelniane Politechniki Koszalińskiej, ISSN 2353-6535, Koszalin 2014.
- **Rafał Wojszczyk**, *Zestawienie metryk oprogramowania obiektowego opartych na statycznej analizie kodu źródłowego*, w: Zarządzanie projektami i modelowanie procesów, strony od 95 do 107, Zeszyty Rady Naukowej Polskiego Towarzystwa Informatycznego, ISBN 978-83-7518-599-7, Warszawa 2013.

Model oceny jakości implementacji wzorców projektowych

mgr inż. Rafał Wojszczyk

dziękuję za uwagę