

Omelette

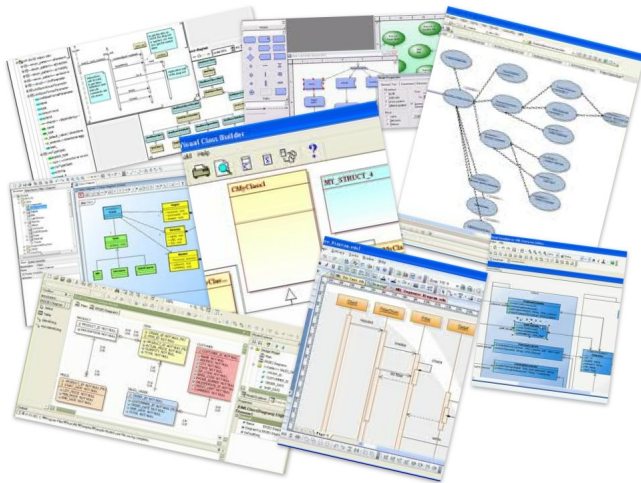
Sławomir Błatkiewicz Jakub Górniak Piotr Piechal
Bartosz Pieńkowski Barnaba Turek Michał Zochniak

14 czerwca 2011

Omelette

- 1 Wprowadzenie
- 2 Gramatyka
- 3 Prototypy
- 4 Układ diagramów
- 5 Podsumowanie

Dostępne oprogramowanie



Alternatywa w postaci opisu tekstowego

```
class Student
```

```
  — imie
```

```
  — nazwisko
```

```
class Uczelnia
```

```
  — nazwa
```

```
association
```

```
  source-object: Student
```

```
  target-object: Uczelnia
```

Wynikowy diagram



Główne zalety

- Szybkość modelowania
- Minimalizm składni
- Rozszerzalność języka
- Modularna architektura klas rysujących

Wykorzystane technologie – Python



Wykorzystane technologie – Qt



Wykorzystane technologie – cd.

- Python
- Qt
- pyQt
- pyparsing
- graphviz

Omelette

- 1 Wprowadzenie
- 2 Gramatyka**
- 3 Prototypy
- 4 Układ diagramów
- 5 Podsumowanie

Opis gramatyki

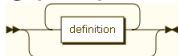
Opis formalny

```
grammar ::= (definition)*
definition ::= ('\n')* header (element | '\n')*
header ::= ( | prototype) parent_name ( | object_name) '\n'
element ::= (operation | attribute | property | constraint) '\n'
attribute ::= ( | static) visibility attribute_name ( |(': ' attribute_type))
              ( |(' = ' attribute_default))
operation ::= ( | static) visibility method_name '(' ( | parameters) ')'
              ( |(': ' return_type))
property ::= property_name ': ' property_value
constraint ::= constraint_type constraint_key constraint_value
```

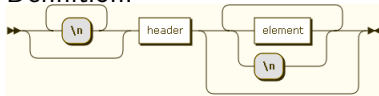
Przykładowy rozbiór

```
dziewczynka ala
+ kot : String = 'buras'
+ idz(gdzie : String) : void
```

Grammar:



Definition:



Header:

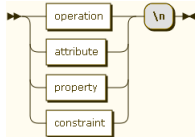


grammar \rightarrow definition \rightarrow parent_name name \rightarrow 'dziewczynka' 'ala'

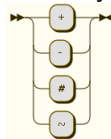
Przykładowy rozbiór 2

```
dziewczynka ala
+ kot : String = 'buras'
+ idz(gdzie : String) : void
```

Element:



Visibility:



Attribute:



element → attribute → visibility ':' type '=' attribute_value → '+' 'kot' ':' 'String' '=' 'buras'

Przykładowy rozbiór 3

```
dziewczynka ala  
+ kot : String = 'buras'  
+ idz(gdzie : String) : void
```

Operation:



Parameters:



Parameter:



element \rightarrow operation \rightarrow visibility method_name '(' parameters ')' ':' return_type

\rightarrow 'idz' '(' parameter ')' ':' void \rightarrow 'idz' '(' parameter_name ':' parameter_type ')' ':' 'void'

\rightarrow 'idz' '(' 'gdzie' ':' 'String' ')' ':' 'void'

Omelette

- 1 Wprowadzenie
- 2 Gramatyka
- 3 Prototypy**
- 4 Układ diagramów
- 5 Podsumowanie

- Biblioteka standardowa
- Powszechność cech wspólnych pomiędzy konstrukcjami
- Podobne obiekty na diagramie

Biblioteka standardowa

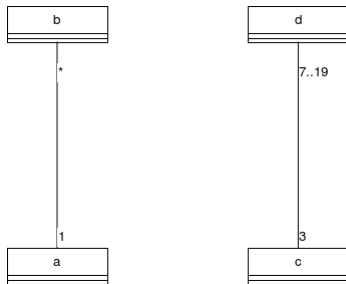
Biblioteka w kodzie

- Łatwe wczytywanie

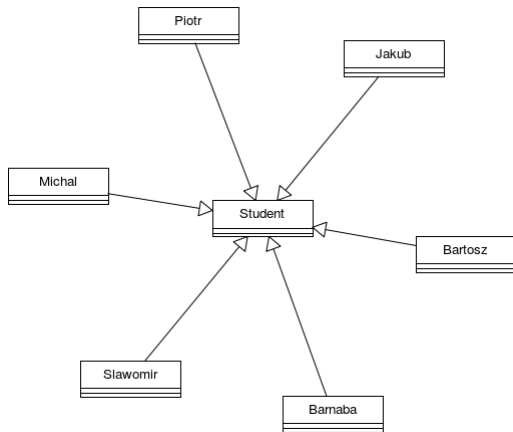
Oddzielna biblioteka

- Łatwa modyfikacja
- Łatwe rozszerzanie
- Łatwe wykorzystanie

Powszechność cech wspólnych pomiędzy konstrukcjami



Podobne obiekty na diagramie

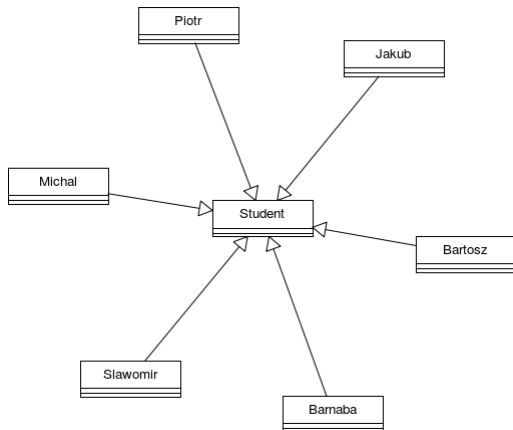


Na czym polega prototypowanie

Jeżeli obiekt **A** jest prototypem obiektu **B**, to:

- Niezainicjowane właściwości obiektu **B** przyjmują wartości własności obiektu **A**
- Obiekt **B** musi spełniać wszystkie wymagania walidacji które musiał spełniać obiekt **A** oraz wymagania walidacji zdefiniowane w obiekcie **A**

Podobne obiekty na diagramie - przypomnienie



Podobne obiekty na diagramie - implementacja

```
generalization student
  source-object : Barnaba
  target-object : Student
```

```
student
  source-object : Michal
```

```
student
  source-object : Bartosz
```

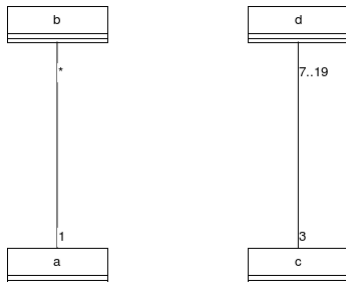
```
...
```

Słowo kluczowe prototype

Istnieje specjalny typ obiektów, prototype.

- Obiekt prototype nie jest umieszczany na diagramie.
- Obiekt prototype nie musi być poprawnym obiektem - nie musi spełniać wymagań walidacji.

Cechy wspólne konstrukcji - przypomnienie



Cechy wspólne konstrukcji - implementacja

```
prototype jeden-do-wielu
```

```
  source-count    : 1
```

```
  target-count    : *
```

```
jeden-do-wielu
```

```
  source-object   : a
```

```
  target-object   : b
```

Dane walidacji

Możliwości

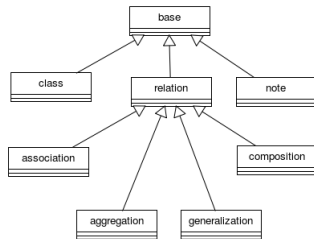
Możemy **wymusić** ustawienie jakiegoś klucza, podać listę **dozwolonych** dla tego klucza wartości, lub **zabronić** używania tego klucza.

```
prototype base relation
  allow name STRING

  allow arrow STRING
  allow direction [none, source, target, both]

  require source-object OBJECT
  allow source-count MULTIPLICITY
  ...
```

Struktura biblioteki



Błędy związane z prototypami

- Cykliczne zależności
- Nieistniejące zależności
- Błędy walidacji
 - Brak wymaganego klucza
 - Zły typ wartości klucza
 - Niedozwolona wartość klucza
 - Referencja do nieistniejącego obiektu

Omelette

- 1 Wprowadzenie
- 2 Gramatyka
- 3 Prototypy
- 4 Układ diagramów**
- 5 Podsumowanie

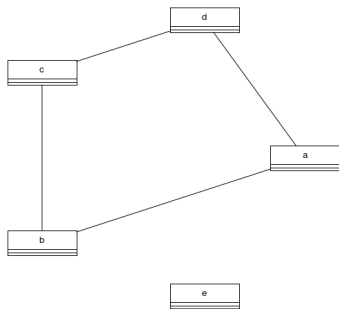
Układ diagramów

- Co to jest układ diagramów?

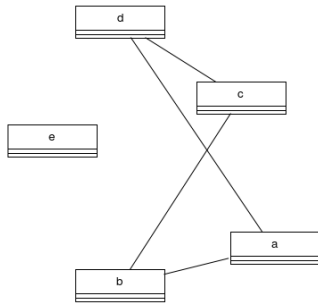
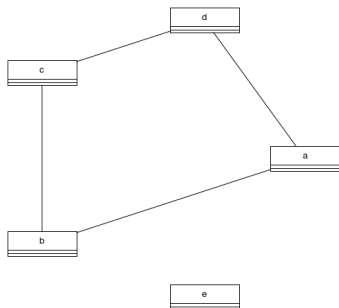
Układ diagramów

- Co to jest układ diagramów?
- Dlaczego jest taki ważny?

Układ diagramów



Układ diagramów



Układ diagramów

- Co to jest układ diagramów?
- Dlaczego jest taki ważny?
- Czym tak naprawdę jest diagram?

Wybrane algorytmy grafowe

- Planaryzacja
- Rozkład kołowy
- Metody energetyczne

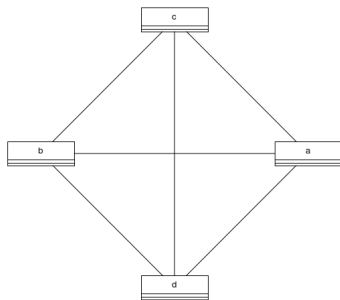
Planaryzacja

- Co znaczy planarny?

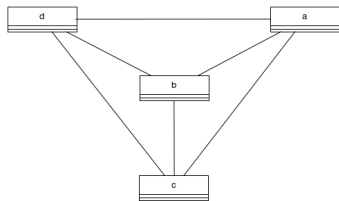
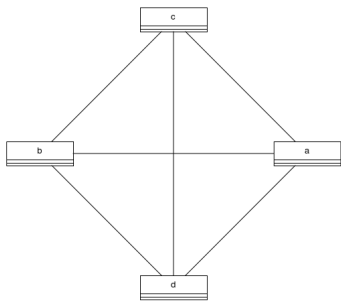
Planaryzacja

- Co znaczy planarny?
- Czym więc jest planaryzacja?

Planaryzacja



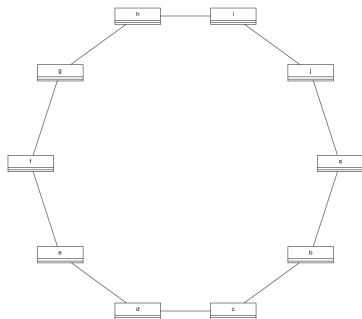
Planaryzacja



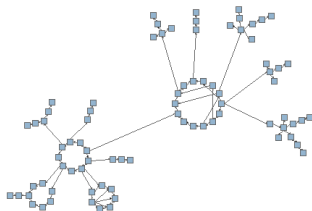
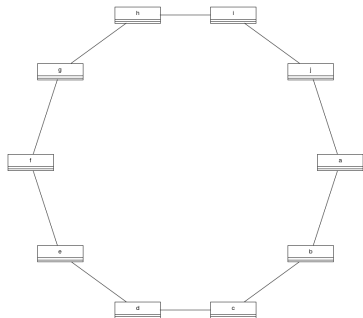
Algorytmy kołowe

- Jak działają?

Algorytmy kołowe

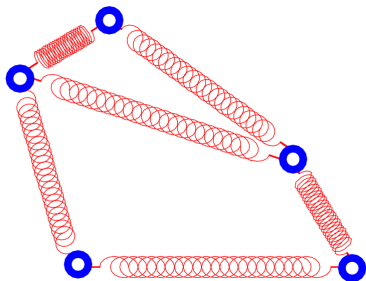


Algorytmy kołowe



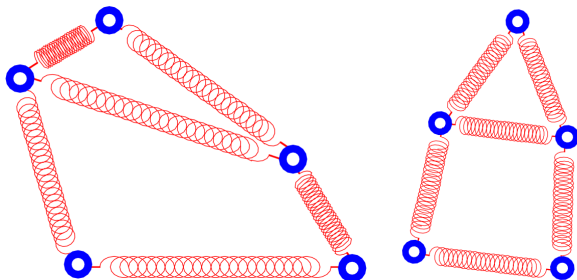
Algorytmy energetyczne

■ Model mechaniczny



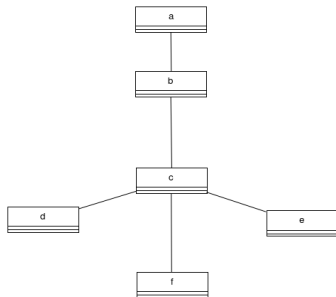
Algorytmy energetyczne

■ Model mechaniczny



Algorytmy energetyczne

■ Model mechaniczny



- Modyfikacje

Algorytmy energetyczne

- Modyfikacje
 - Kamada-Kawai

Algorytmy energetyczne

- Modyfikacje
 - Kamada-Kawai
 - Fruchterman-Reingold

Algorytmy energetyczne

- Modyfikacje
 - Kamada-Kawai
 - Fruchterman-Reingold
 - Hadany-Harel

- Modyfikacje
 - Kamada-Kawai
 - Fruchterman-Reingold
 - Hadany-Harel
 - Harel-Koren

- Modyfikacje
 - Kamada-Kawai
 - Fruchterman-Reingold
 - Hadany-Harel
 - Harel-Koren
 - Model grawitacyjny

- Co to jest?

- Co to jest?
- Dlaczego o nim mówię?

Omelette

- 1 Wprowadzenie
- 2 Gramatyka
- 3 Prototypy
- 4 Układ diagramów
- 5 Podsumowanie**

`github.com/pienkowb/omelette`