AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

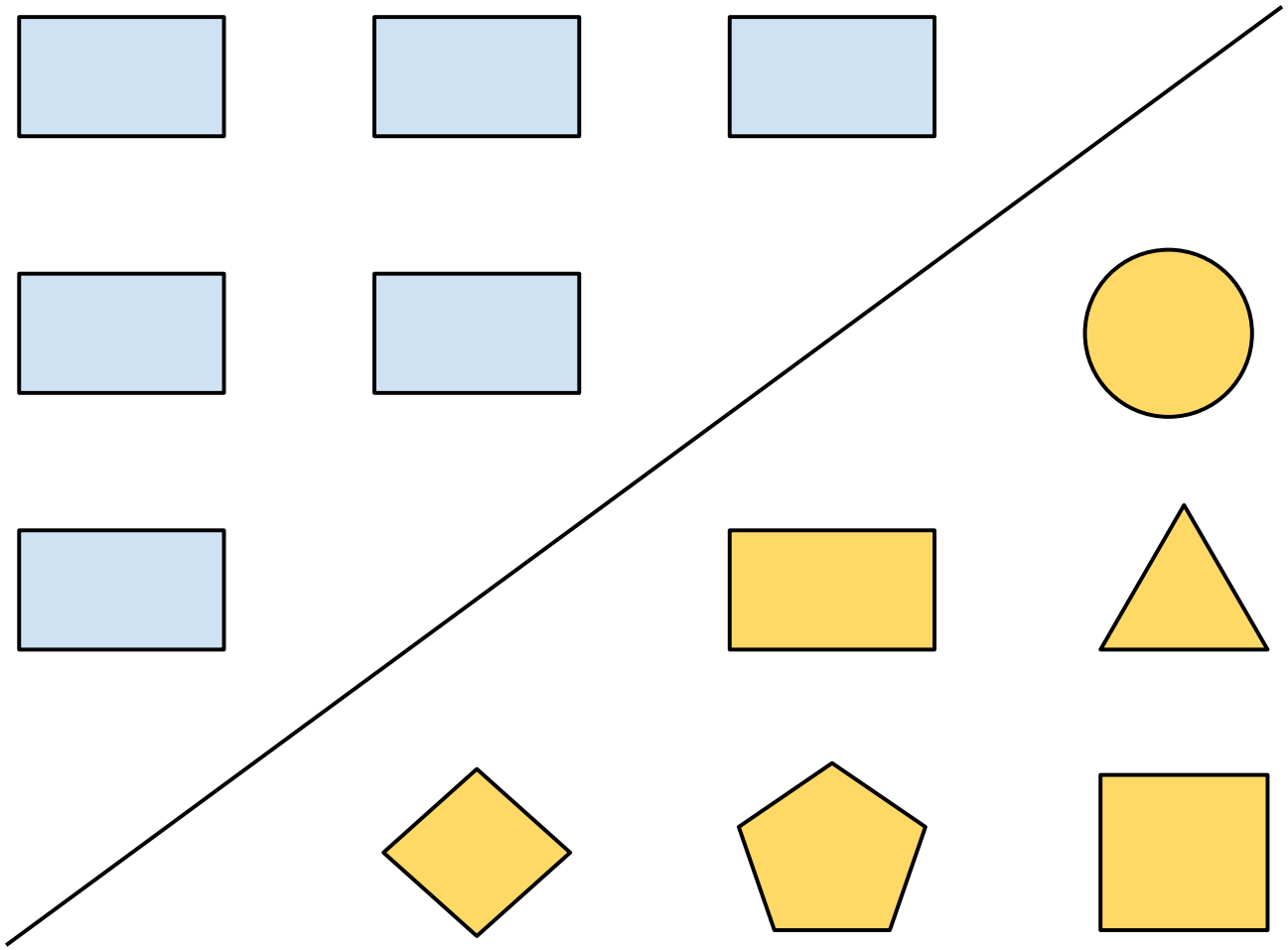
Projekt języka programowania wspierającego przetwarzanie rozproszone na platformach heterogenicznych.

Kajetan Rzepecki

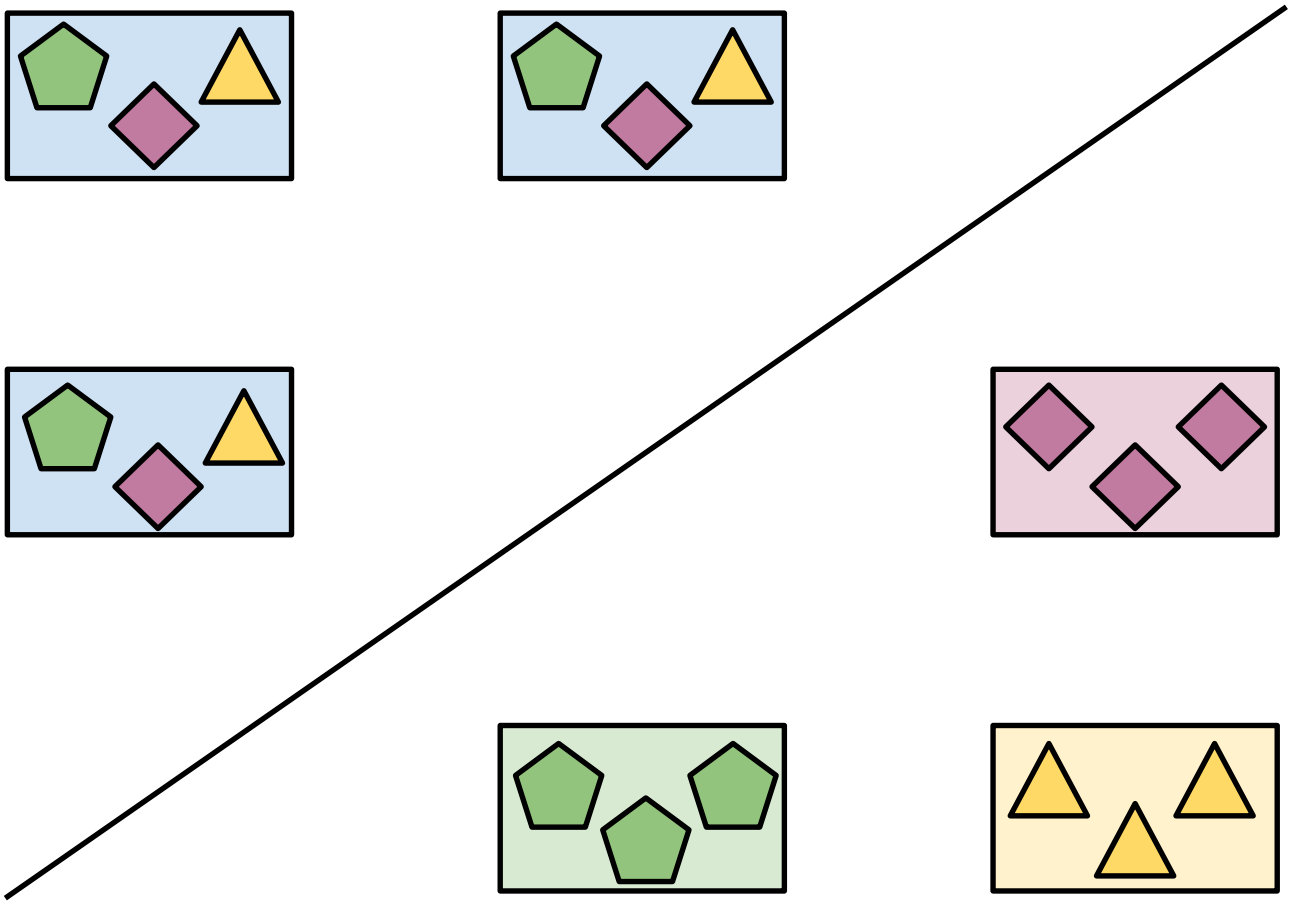
Wydział EAIiIB
Katedra Informatyki Stosowanej

27 maja 2015

Problem - Heterogeniczność



Problem - Heterogeniczność



Problem - Na czym się skupiamy

- Skalowalność
- Dynamiczność
- Niezawodność
- Konfiguracja

Problem - Na czym się skupiamy

- Skalowalność
- Dynamiczność
- Niezawodność
- Konfiguracja

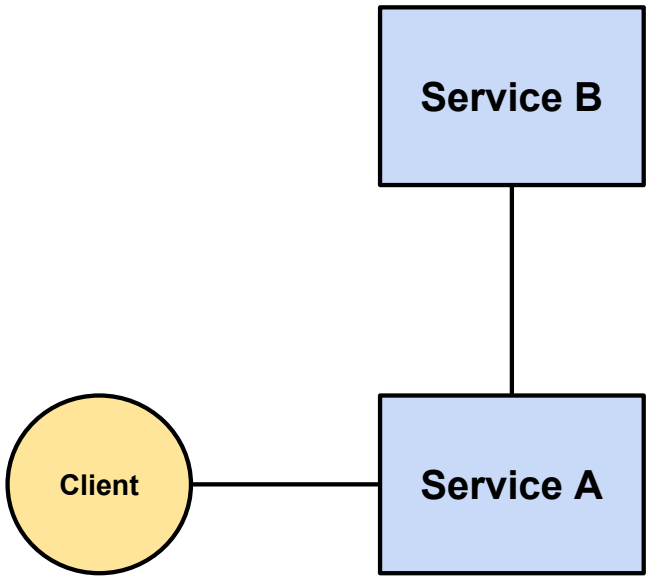
- Heterogeniczność?

Problem - State of the Art

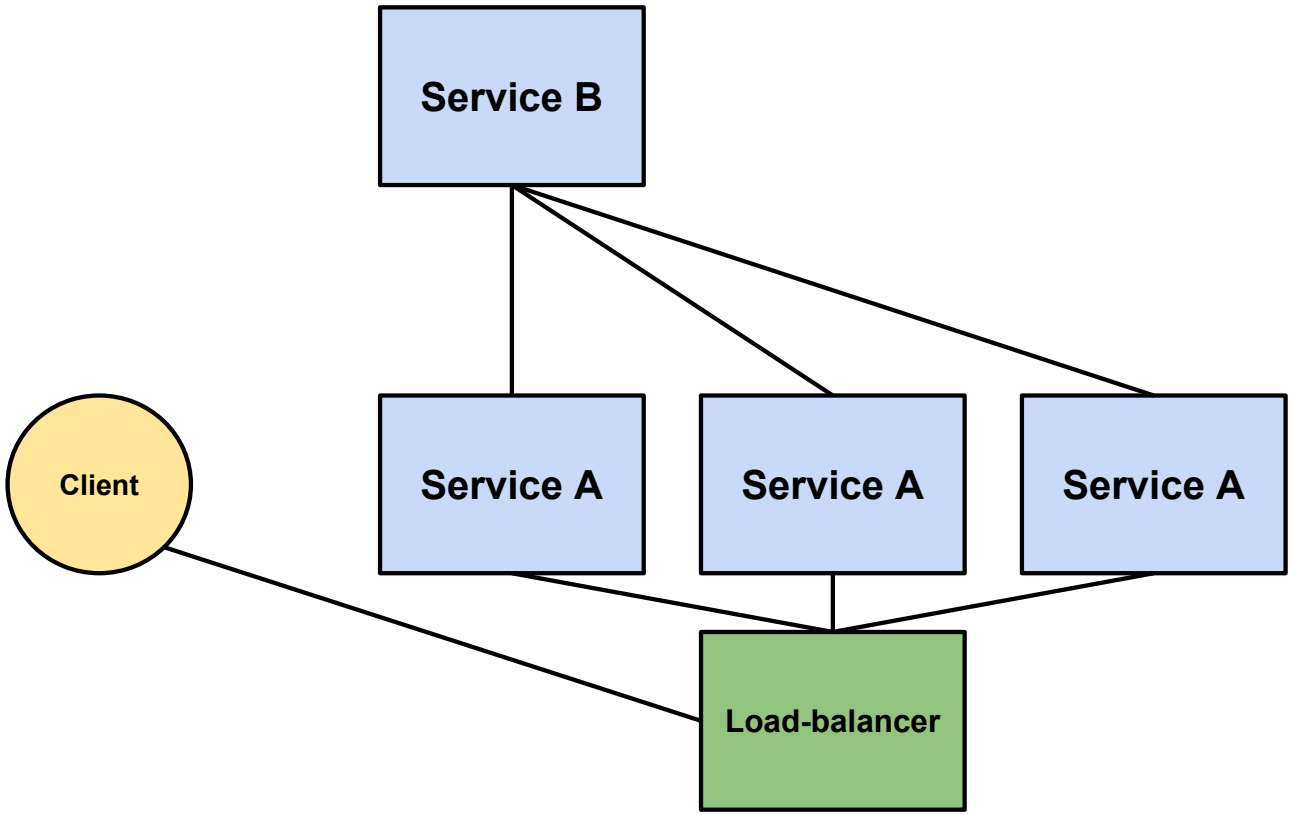
Problem - State of the Art

- Heterogeniczność?

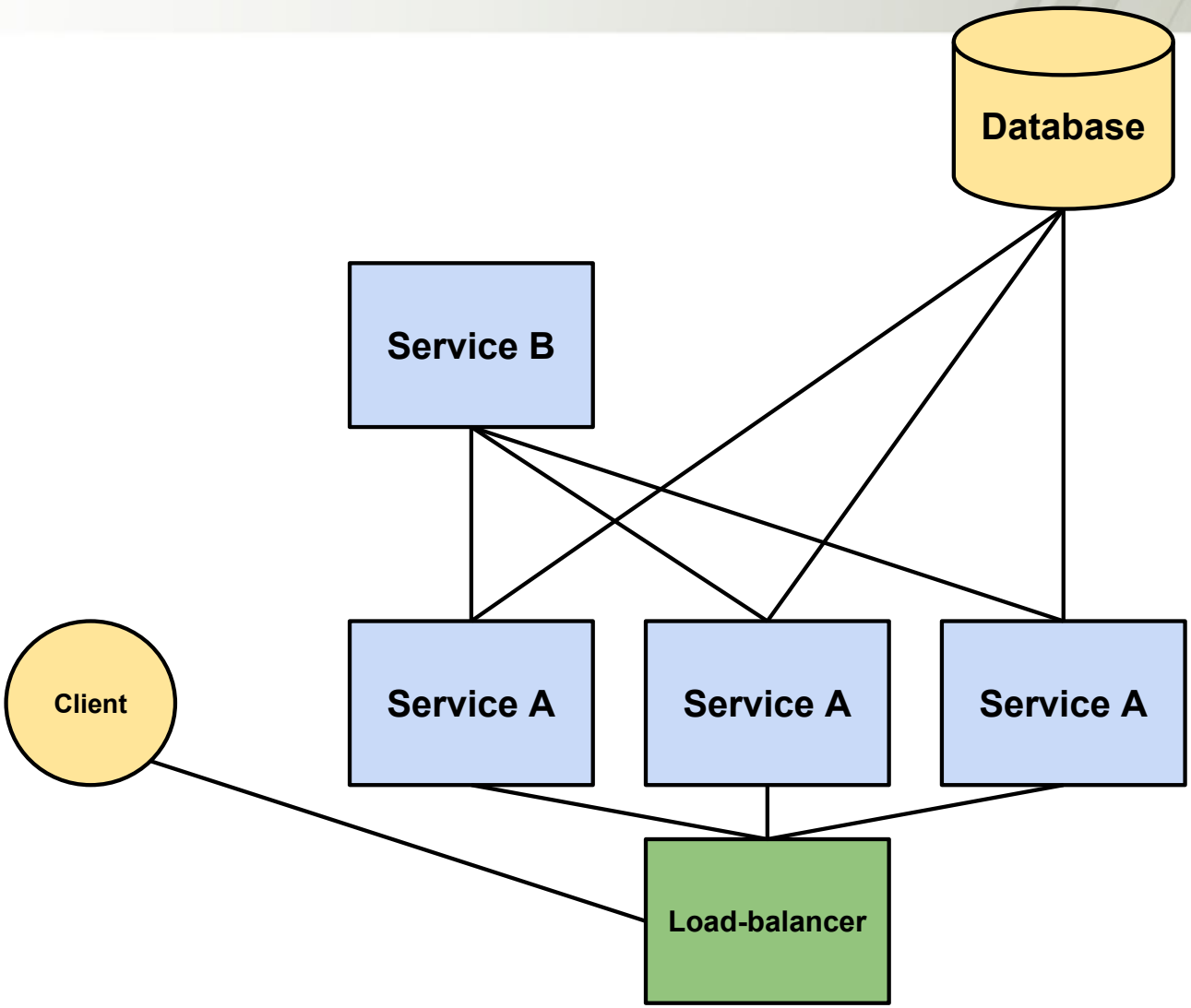
Problem - Przykład



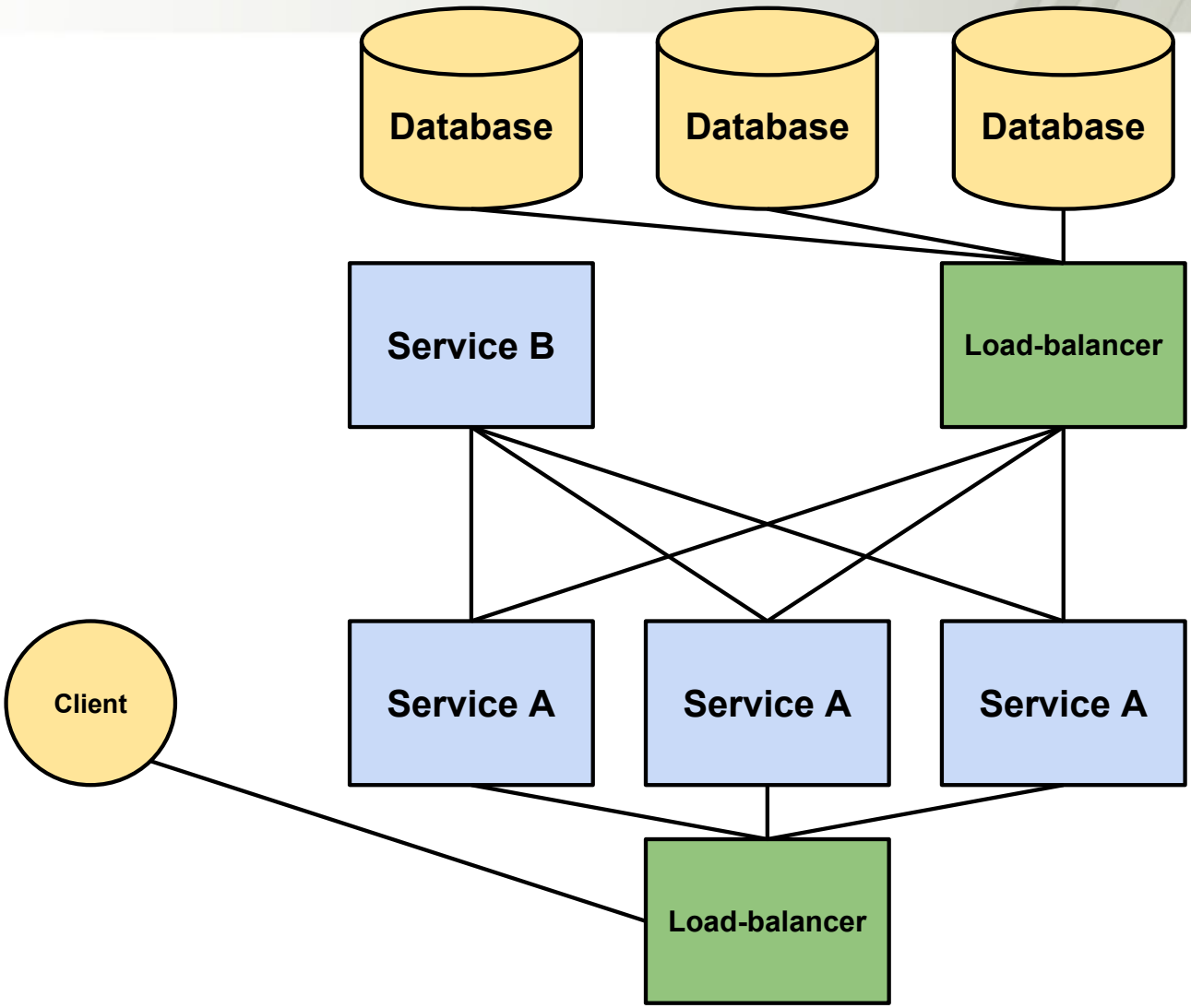
Problem - Przykład



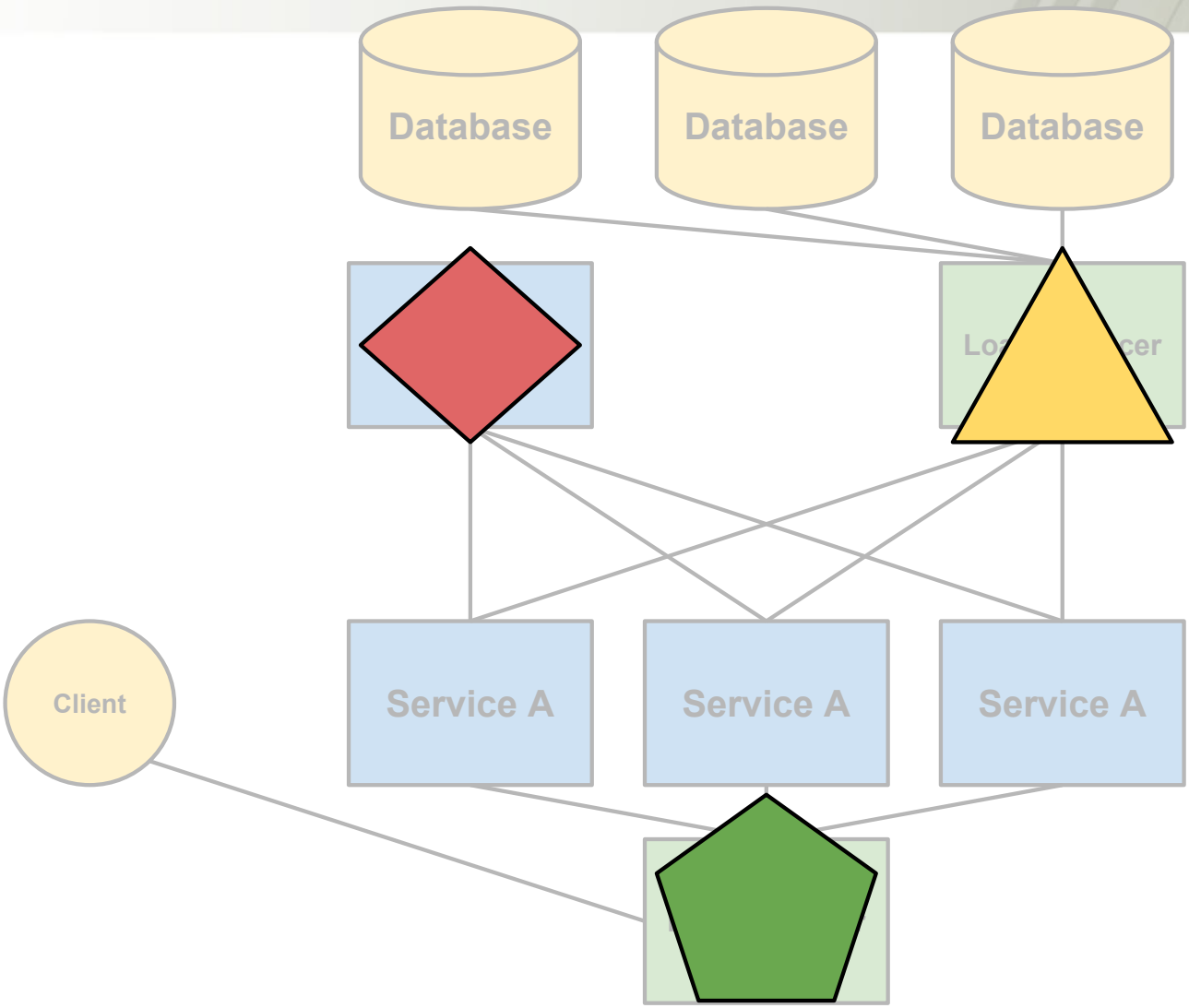
Problem - Przykład



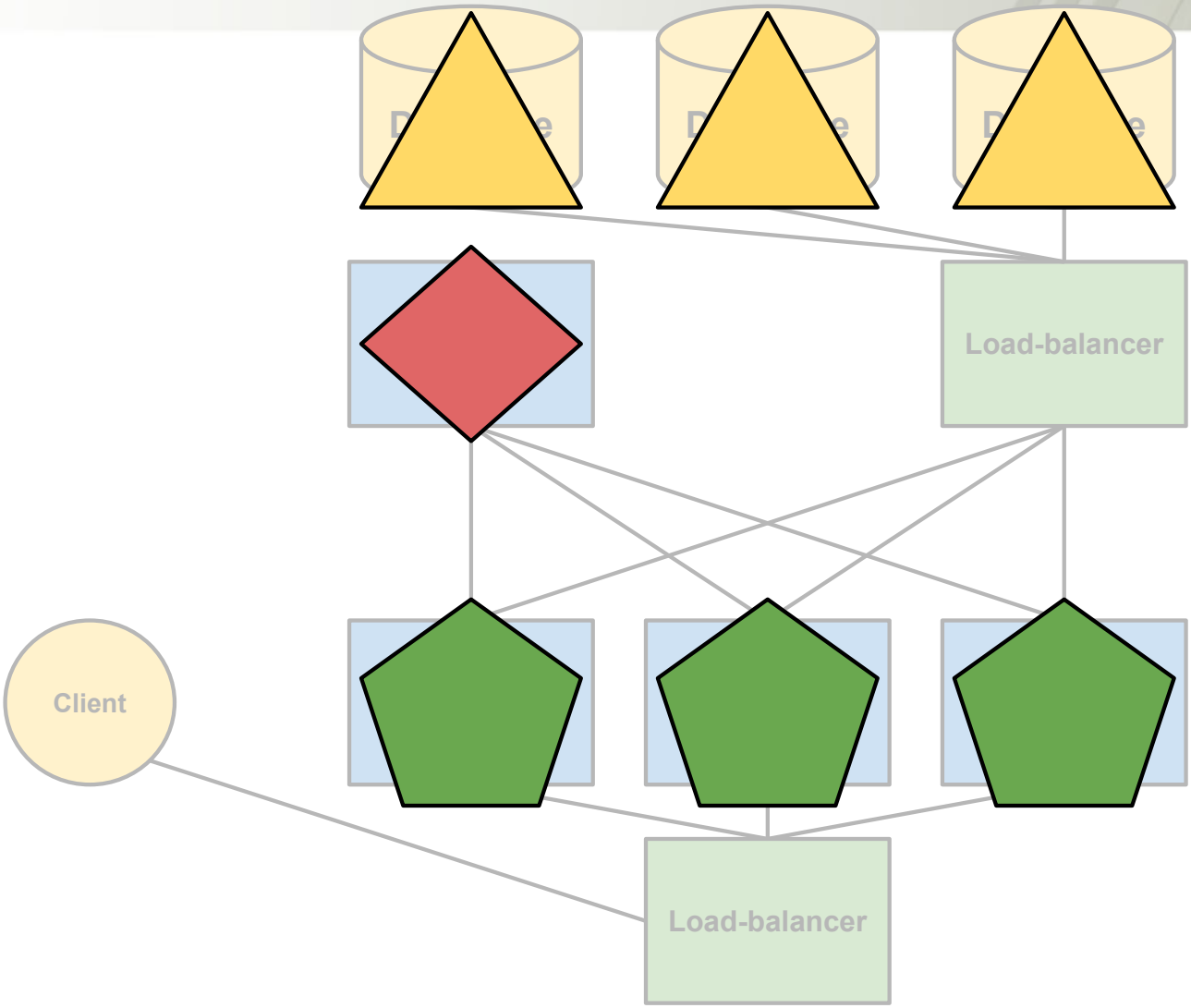
Problem - Przykład



Problem - Niezależność vs. świadomość



Problem - Niezależność vs. świadomość



Architektura - Proste jądro

- przenośne i integrowalne z wieloma platformami

Architektura - Proste jądro

- przenośne i integrowalne z wieloma platformami
- λ calculus - funkcje

```
(lambda (x) x)
```

Architektura - Proste jądro

- przenośne i integrowalne z wieloma platformami
- λ calculus - funkcje

```
(lambda (x) x)
```

- kontynuacje

```
(+ 1 (reset (* 2 (shift k (k (k 4)))))) ; 17 wtf!?
```

Architektura - Proste jądro

- przenośne i integrowalne z wieloma platformami
- λ calculus - funkcje

```
(lambda (x) x)
```

- kontynuacje

```
(+ 1 (reset (* 2 (shift k (k (k 4)))))) ; 17 wtf!?
```

- system regułowy

```
(whenever (and (module ?m) (provides ?m feature))
  (start-using ?m))
```

Architektura - System modułowy

- moduł posiada interfejs vs. moduł wymaga interfejsu:

Architektura - System modułowy

- moduł posiada interfejs vs. moduł wymaga interfejsu:

```
(module Foo
  (provide bar)
  (function (bar) ...))
(module Bar
  (require Foo))
```

Architektura - System modułowy

- moduł posiada interfejs vs. moduł wymaga interfejsu:

```
(module Foo
  (provide bar)
  (function (bar) ...))
(module Bar
  (require Foo))
```

```
(module Foo
  (function (bar) ...))
(module Bar
  (require ?m
    (and (module ?m)
         (provides ?m bar))))
```

Architektura - System modułowy

- moduł posiada interfejs vs. moduł wymaga interfejsu:

```
(module Foo
  (provide bar)
  (function (bar) ...))
(module Bar
  (require Foo))
```

```
(module Foo
  (function (bar) ...))
(module Bar
  (require ?m
    (and (module ?m)
         (provides ?m bar))))
```

- zintegrowany z kompilatorem

Architektura - Przetwarzanie wiedzy

- automatyczne odkrywanie i propagacja wiedzy

```
(module Foo                    (module Foo)
  (function (bar) ...))        (provides Foo bar)
```

Architektura - Przetwarzanie wiedzy

- automatyczne odkrywanie i propagacja wiedzy

```
(module Foo                          (module Foo)
  (function (bar) ...))              (provides Foo bar)
```

- dynamiczność, niezawodność i skalowalność

Architektura - Przetwarzanie wiedzy

- automatyczne odkrywanie i propagacja wiedzy

```
(module Foo                          (module Foo)
  (function (bar) ...))              (provides Foo bar)
```

- dynamiczność, niezawodność i skalowalność
- przetwarzanie złożonych zdarzeń

Architektura - Przetwarzanie wiedzy

- automatyczne odkrywanie i propagacja wiedzy

```
(module Foo                          (module Foo)
  (function (bar) ...))              (provides Foo bar)
```

- dynamiczność, niezawodność i skalowalność
- przetwarzanie złożonych zdarzeń
- świadomość platformy

Implementacja - Kompilator

- bootstrap
- LLVM

Implementacja - Kompilator

- bootstrap
- LLVM

- dystrybucja kodu przez uruchamialne archiva LLVM IR
- AOT i JIT

Implementacja - Kompilator

- bootstrap
- LLVM

- dystrybucja kodu przez uruchamialne archiva LLVM IR
- AOT i JIT

- x86, ARM, JavaScript, ...

Implementacja - Kompilator

- bootstrap
- LLVM

- dystrybucja kodu przez uruchamialne archiva LLVM IR
- AOT i JIT

- x86, ARM, JavaScript, ...

- Epiphany, Xtensa

Implementacja - Kompilator

- bootstrap
- LLVM

- dystrybucja kodu przez uruchamialne archiva LLVM IR
- AOT i JIT

- x86, ARM, JavaScript, ...

- Epiphany, Xtensa

- MOS 6502?

Implementacja - System regułowy

- algorytm Rete + rozszerzenia

Implementacja - System regułowy

- algorytm Rete + rozszerzenia
- wnioskowanie w przód:

```
(whenever (and (module ?m)
               (provides ?m ?f))
  (list ?m ?f))
```

Implementacja - System regułowy

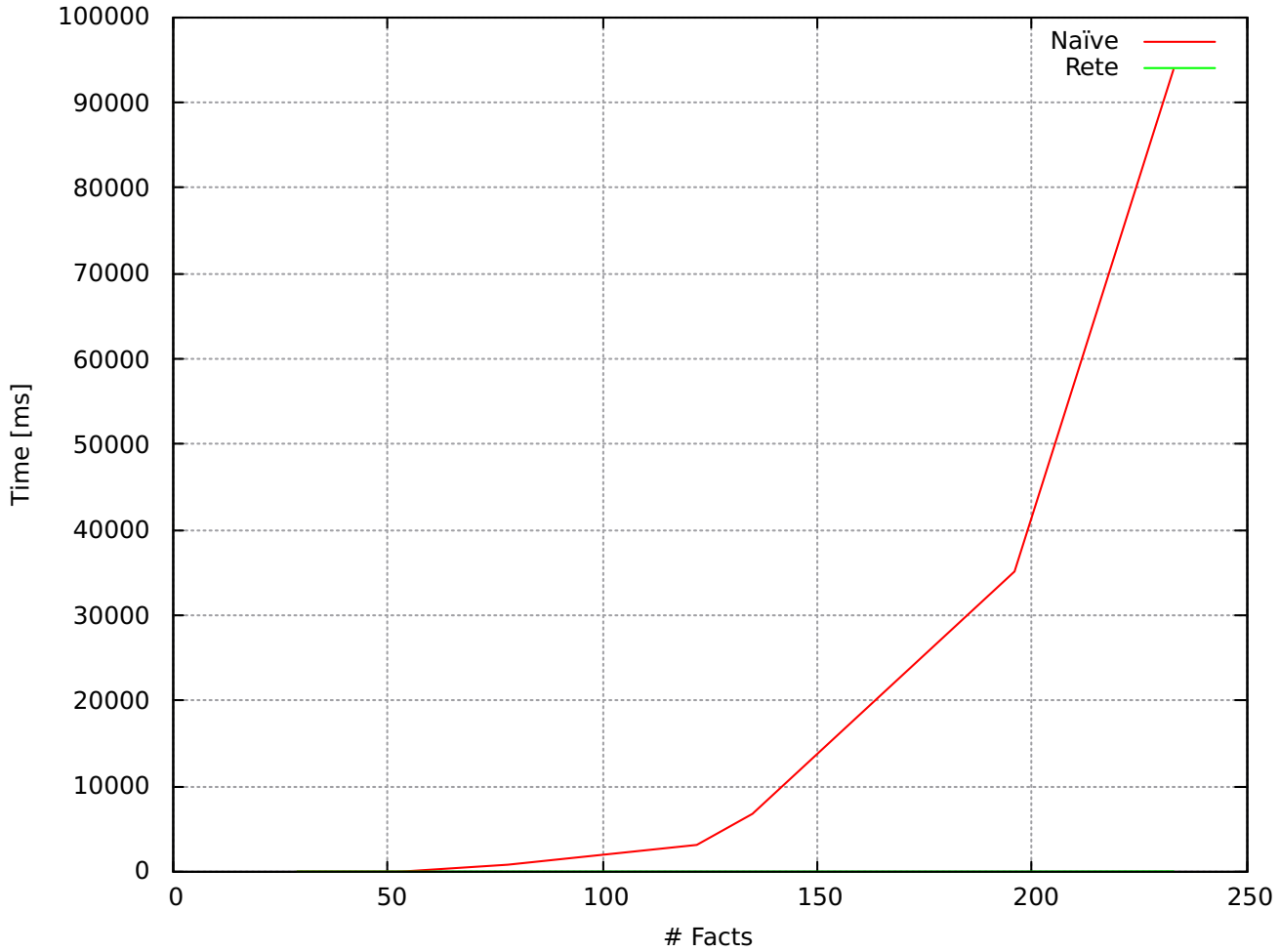
- algorytm Rete + rozszerzenia
- wnioskowanie w przód:

```
(whenever (and (module ?m)
               (provides ?m ?f))
  (list ?m ?f))
```

- wnioskowanie wstecz:

```
(select (?m ?f)
        (and (module ?m)
             (provides ?m ?f)))
```

Implementacja - Rete



AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Dziękuję za uwagę.

Kajetan Rzepecki