Złożone architektury bez podziału na warstwy.

Paweł Cesar Sanjuan Szklarz

Confitura 2012

30 czerwca 2012

Plan

Plan:

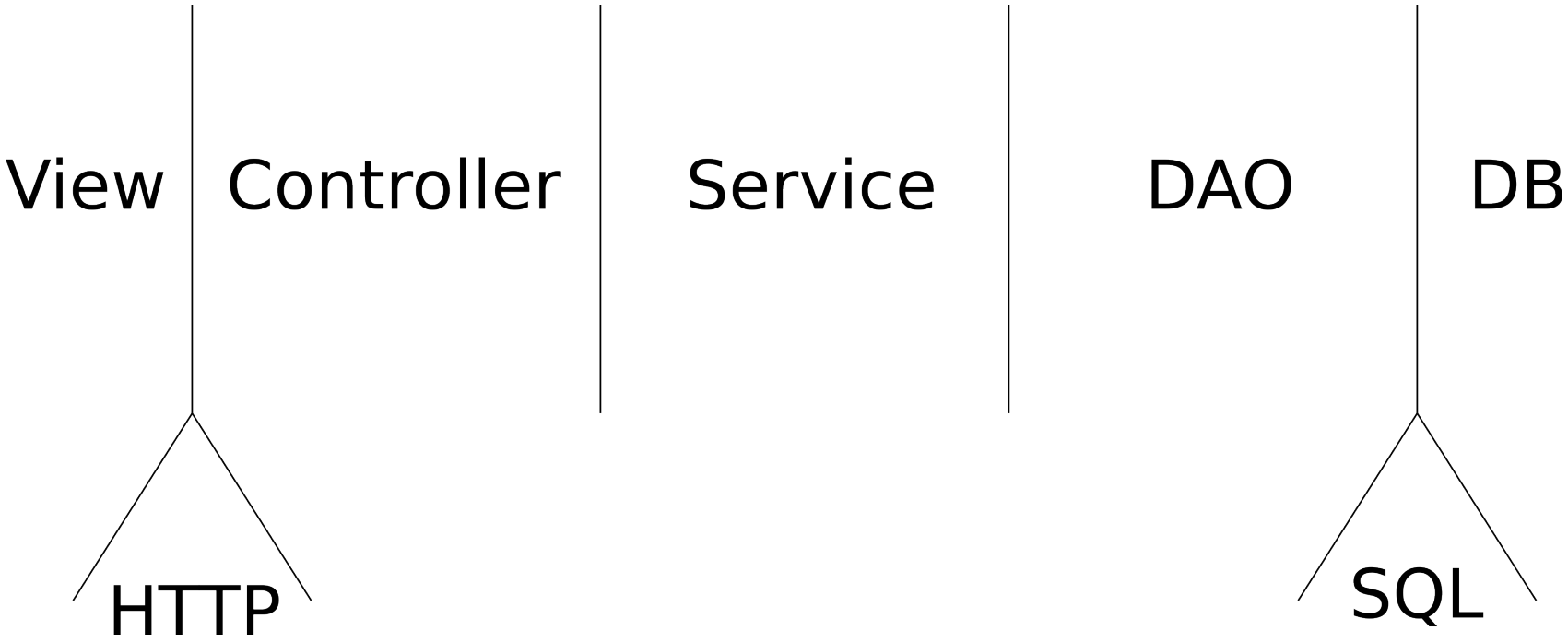
od

Warstwy $\Rightarrow$ Porządek $\Rightarrow$ Złożone systemy

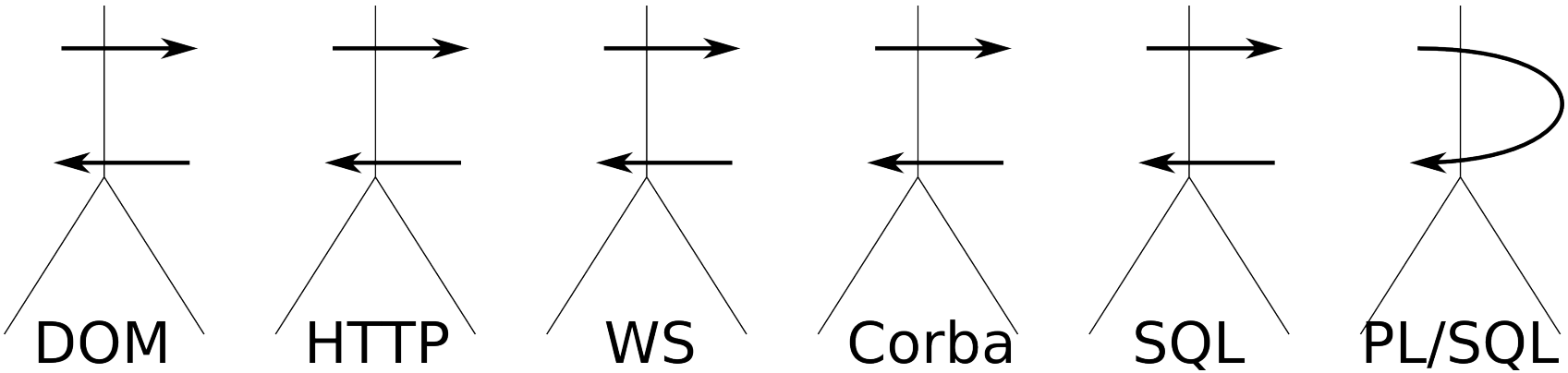
do

Inject $\Rightarrow$ Architektura $\Rightarrow$ Złożone systemy

Warstwy???



Za dużo warstw



Demo

Logika wywołań pomiędzy warstwami.

Wymuszenie podziału logiki w kodzie

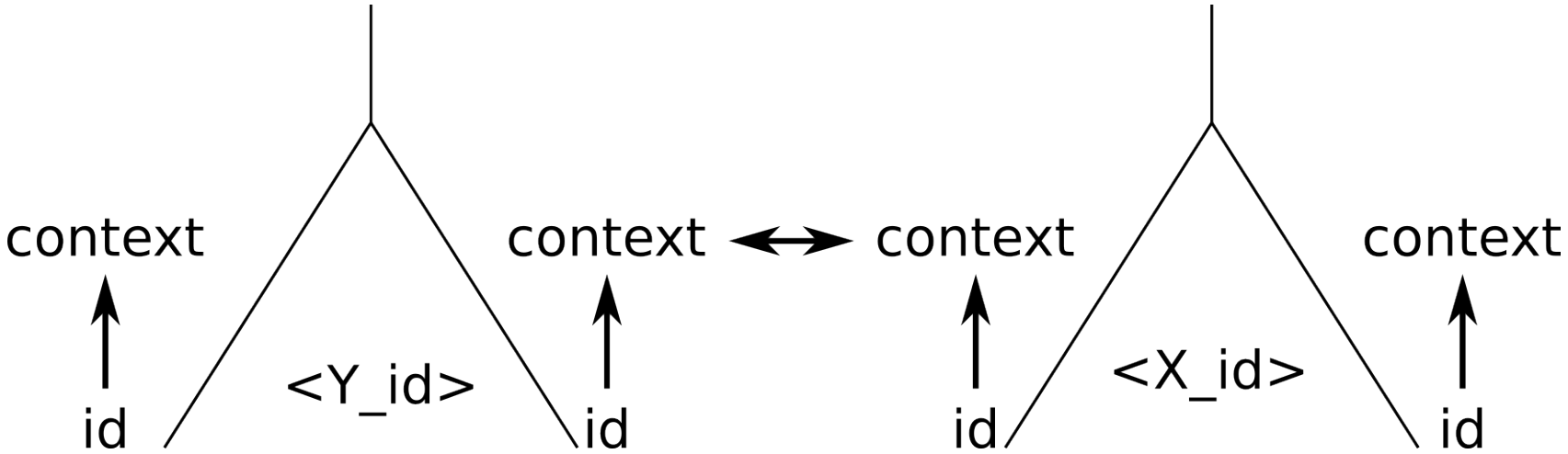
```
public class Controller {
  private Service service;
  void operation(String args) {
    service.operation(args);
  }
}

public class Service {
  private DAO dao;
  void operation(String args) {
    dao.operation(args);
  }
}

public class DAO {
  private DB db;
  void operation(String args) {
    // logic
  }
}
```

Ubogie przekazywanie kontekstu

"`service.operation(args)`" : args $\Rightarrow$ kontekst

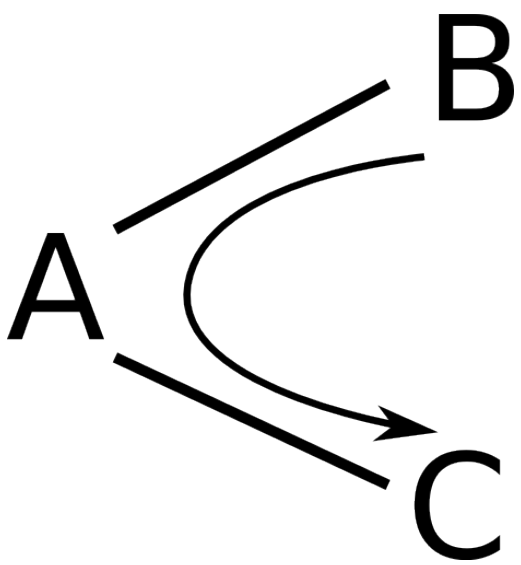
Ubogie przekazywanie kontekstu

"`service.operation(args)`" : args $\Rightarrow$ kontekst

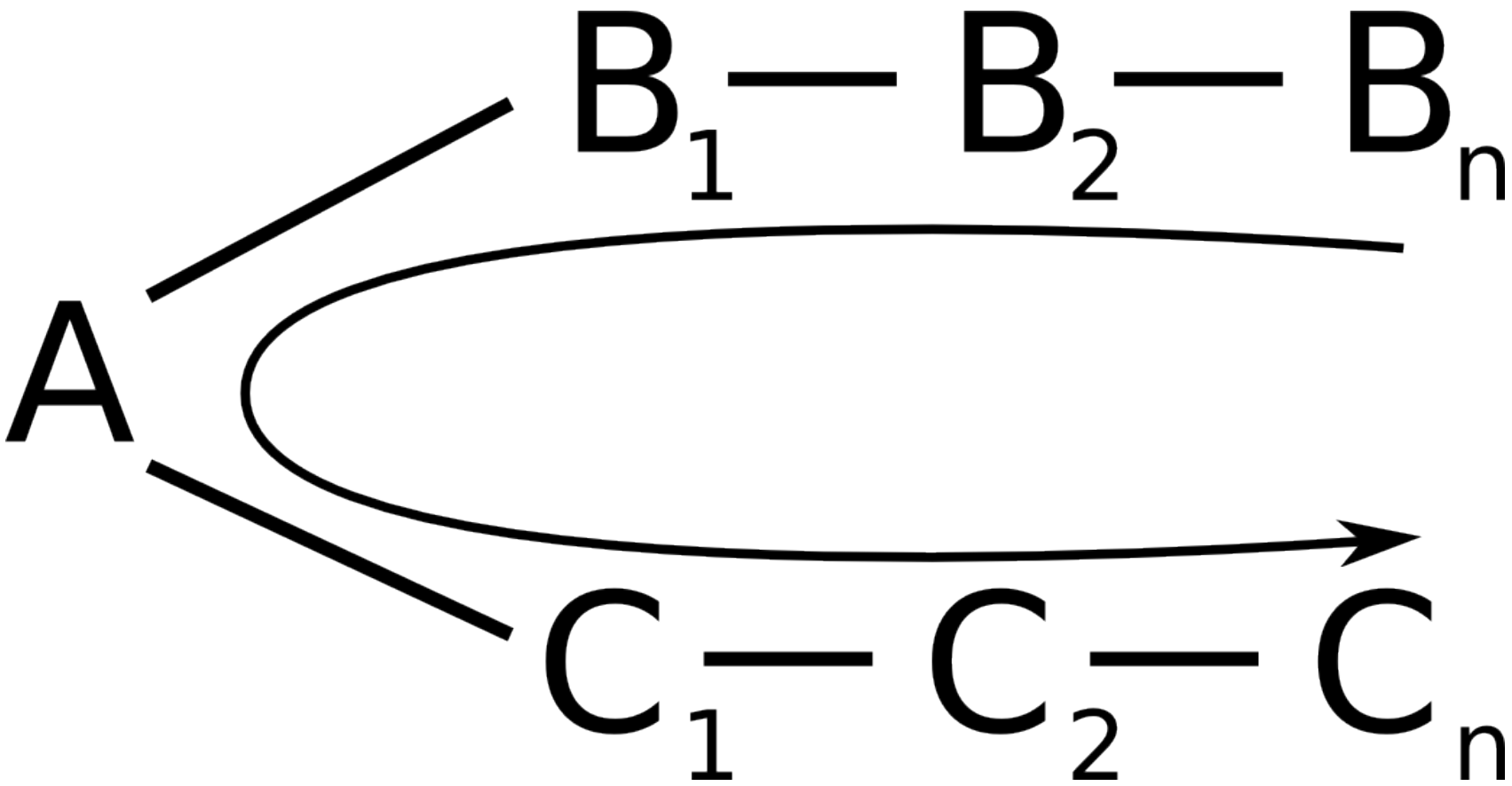


Zaśmiecenie logiki

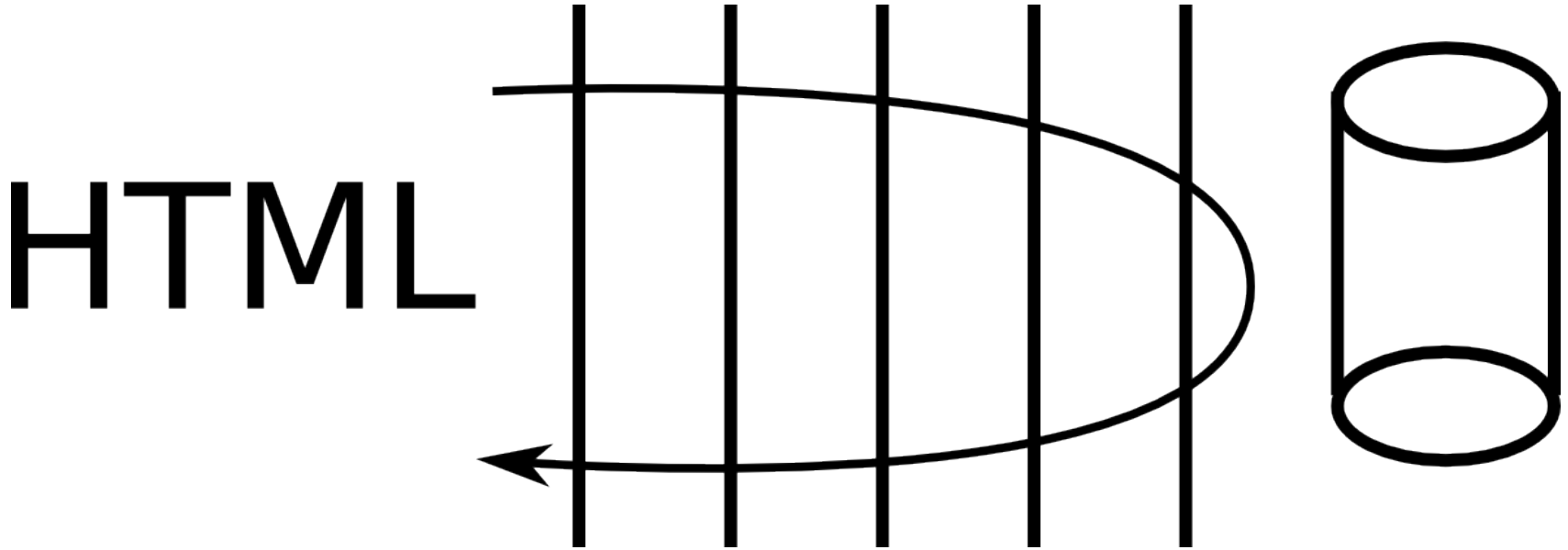
```
A {
  Data logicParameter = B.extractData(...);
  C.businessOperation(logicParameter);
}
```



PASKUDNE!!! zaśmiecenie logiki

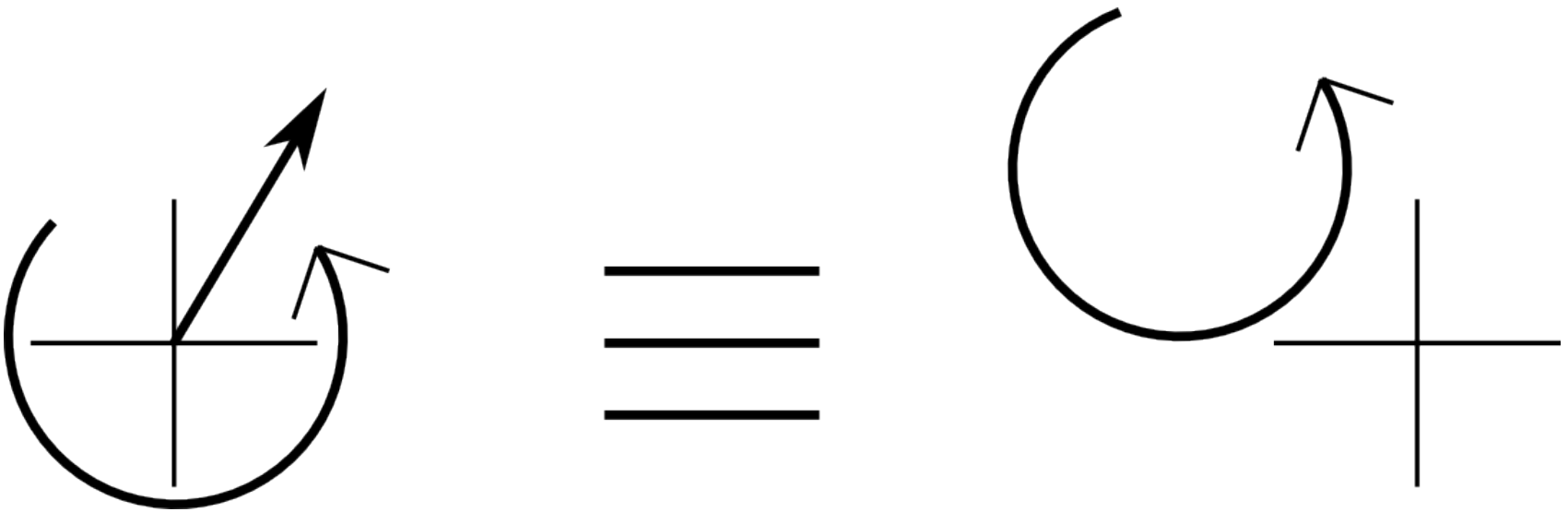


Pozornie wyróżniony kierunek



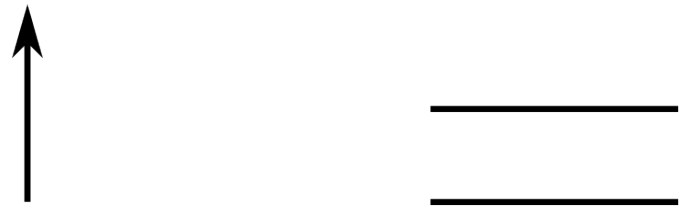
Ciekawostka

2d: Obrót + przesunięcie = Obrót



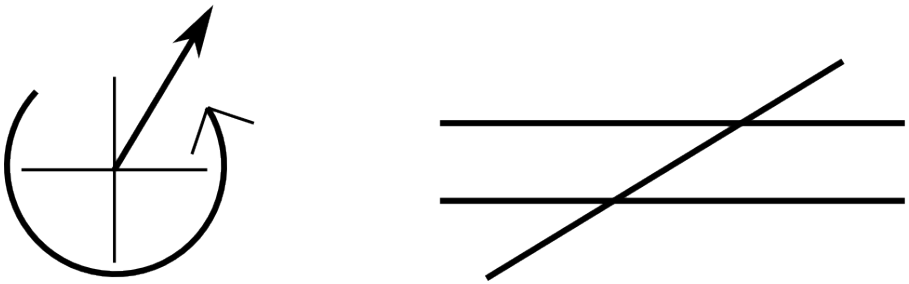
Dowód

$$(O_1 O_2)(O_2 O_3) = O_1 O_3$$



2 odbicia

2 odbicia



2 odbicia!!!

Demo

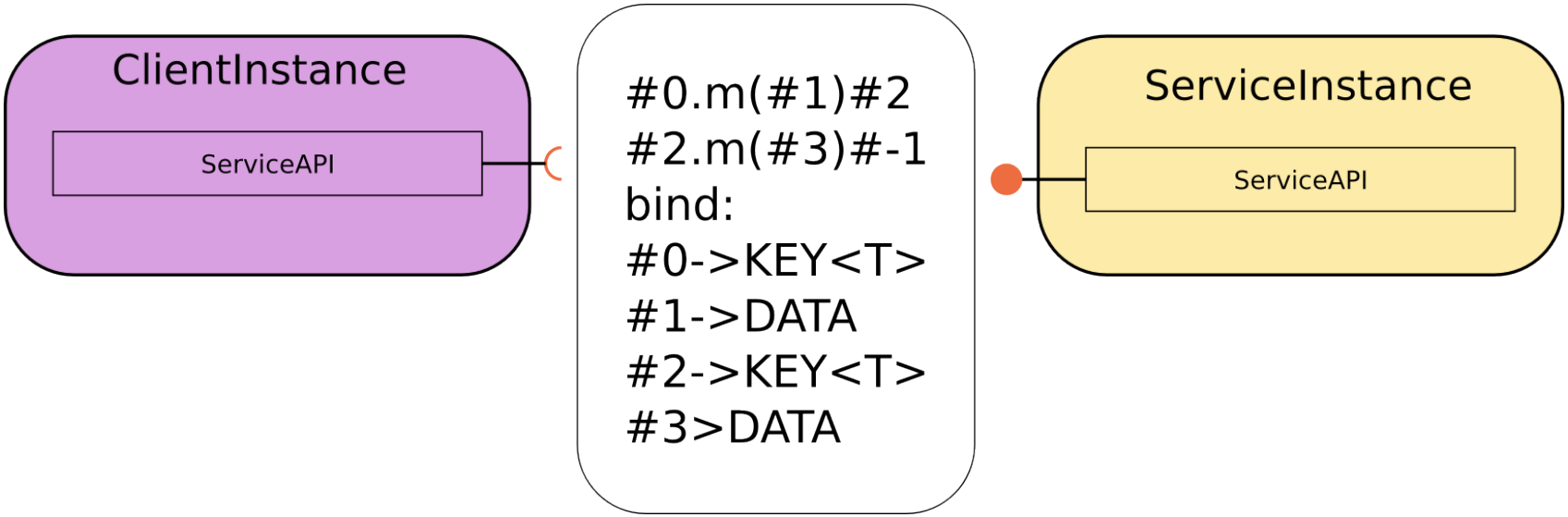
DEMO

Zdalne injektowanie zależności

```
class A {
  @Inject private Service service;
}
class B implements Service {}
```

Zdalne injektowanie zależności

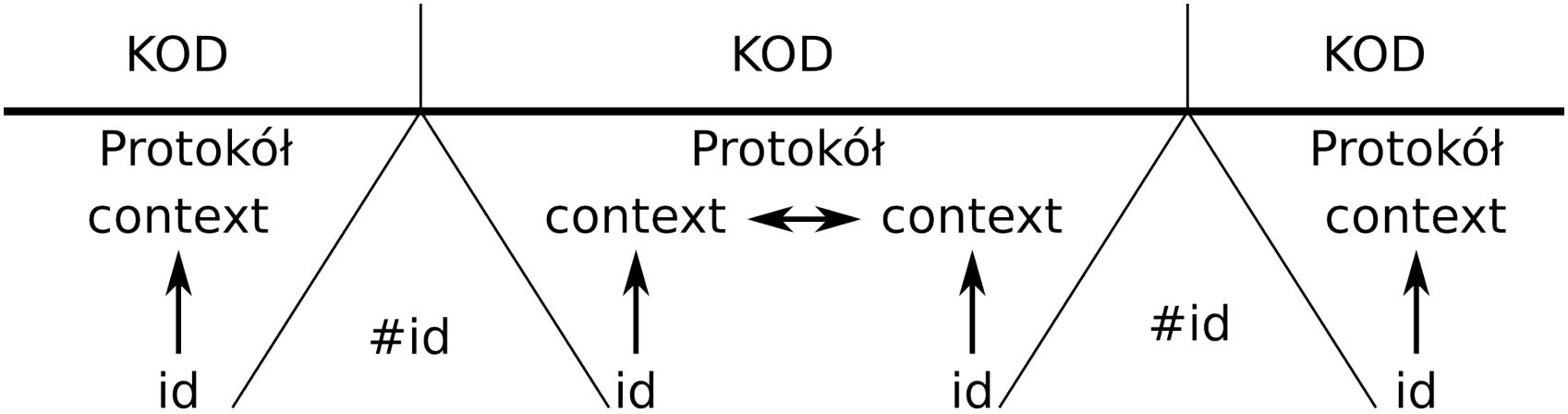
```
class A {
  @Inject private Service service;
}
class B implements Service {}
```



Problem - Wymuszenie podziału logiki w kodzie

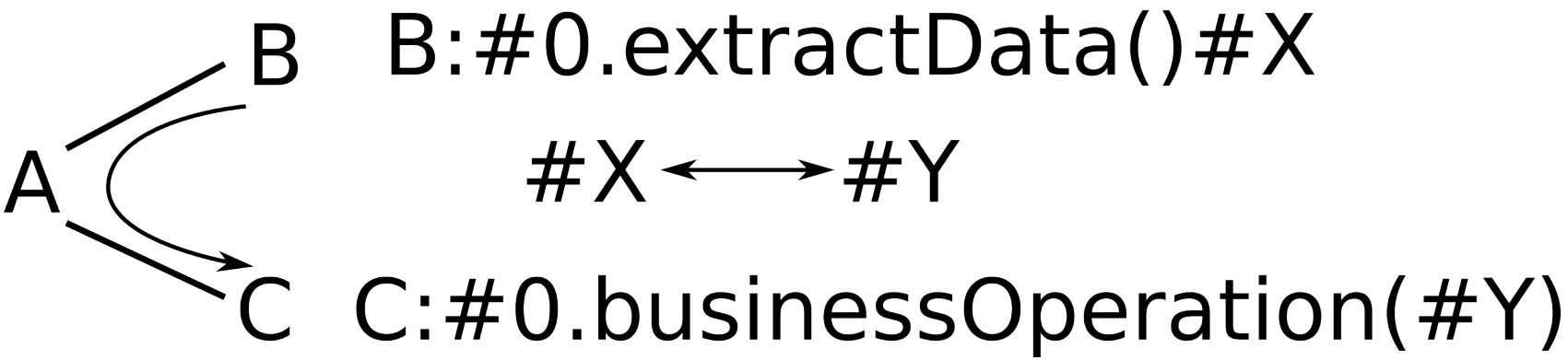
```
class client {
  @Inject private PublicAPI service;
}

interface PublicAPI {
}

class ServiceImplementation implements InternalAPI {
}
interface InternalAPI extends PublicAPI {
}
```

Problem - Ubogie przekazywanie kontekstu



Problem - Zaśmiecenie logiki

```
A {
  Data logicParameter = B.extractData(...);
  C.businessOperation(logicParameter);
}
```



Demo

DEMO

Czy naprawdę warto??

Masło maślane?

Czy naprawdę warto??

Masło maślane?

Why Functional Programming Matters - John Hughes 1984

Modular design + Glue $\Rightarrow$ Complex systems

Czy naprawdę warto??

Masło maślane?

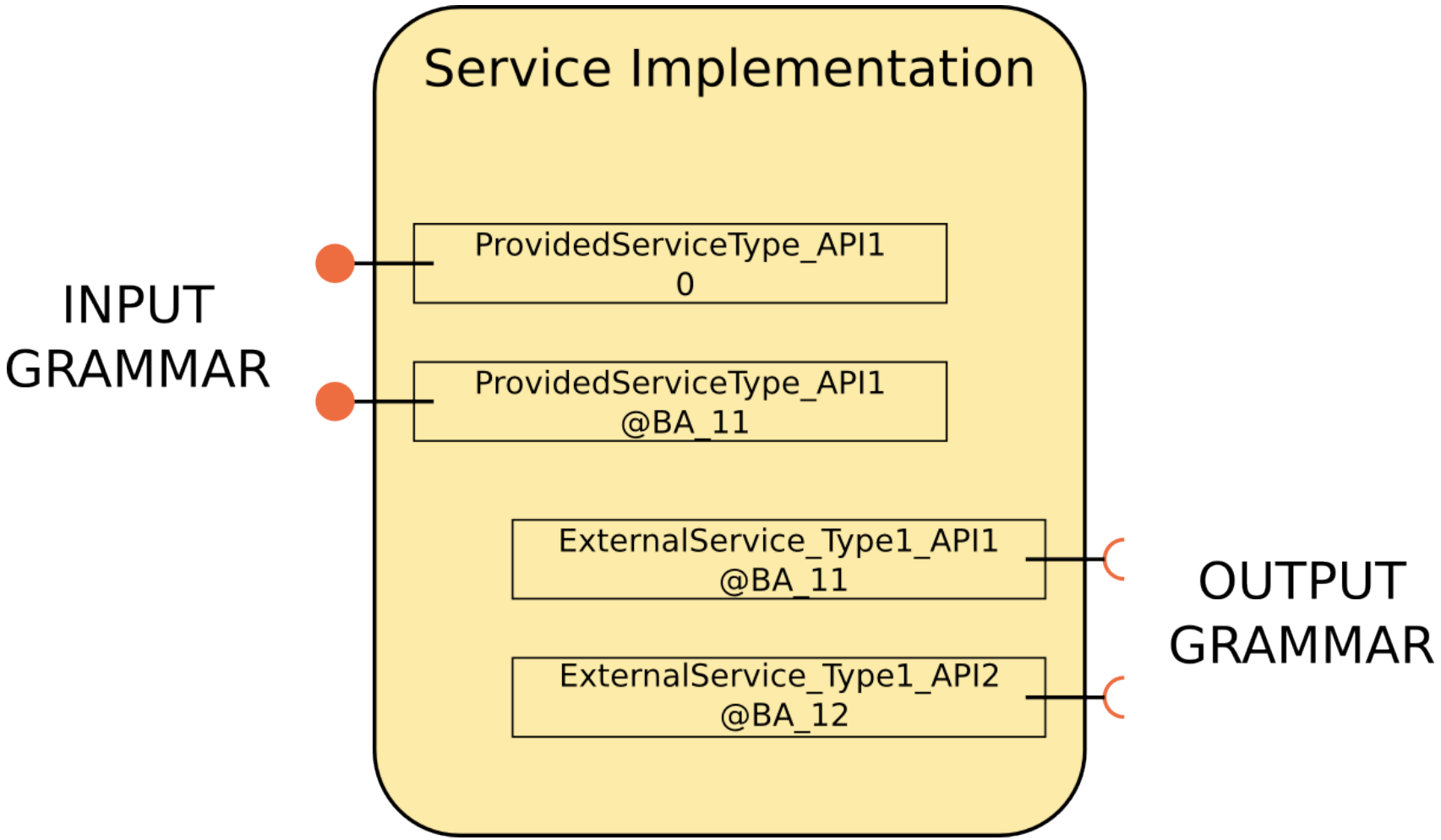
Why Functional Programming Matters - John Hughes 1984

Modular design + Glue $\Rightarrow$ Complex systems

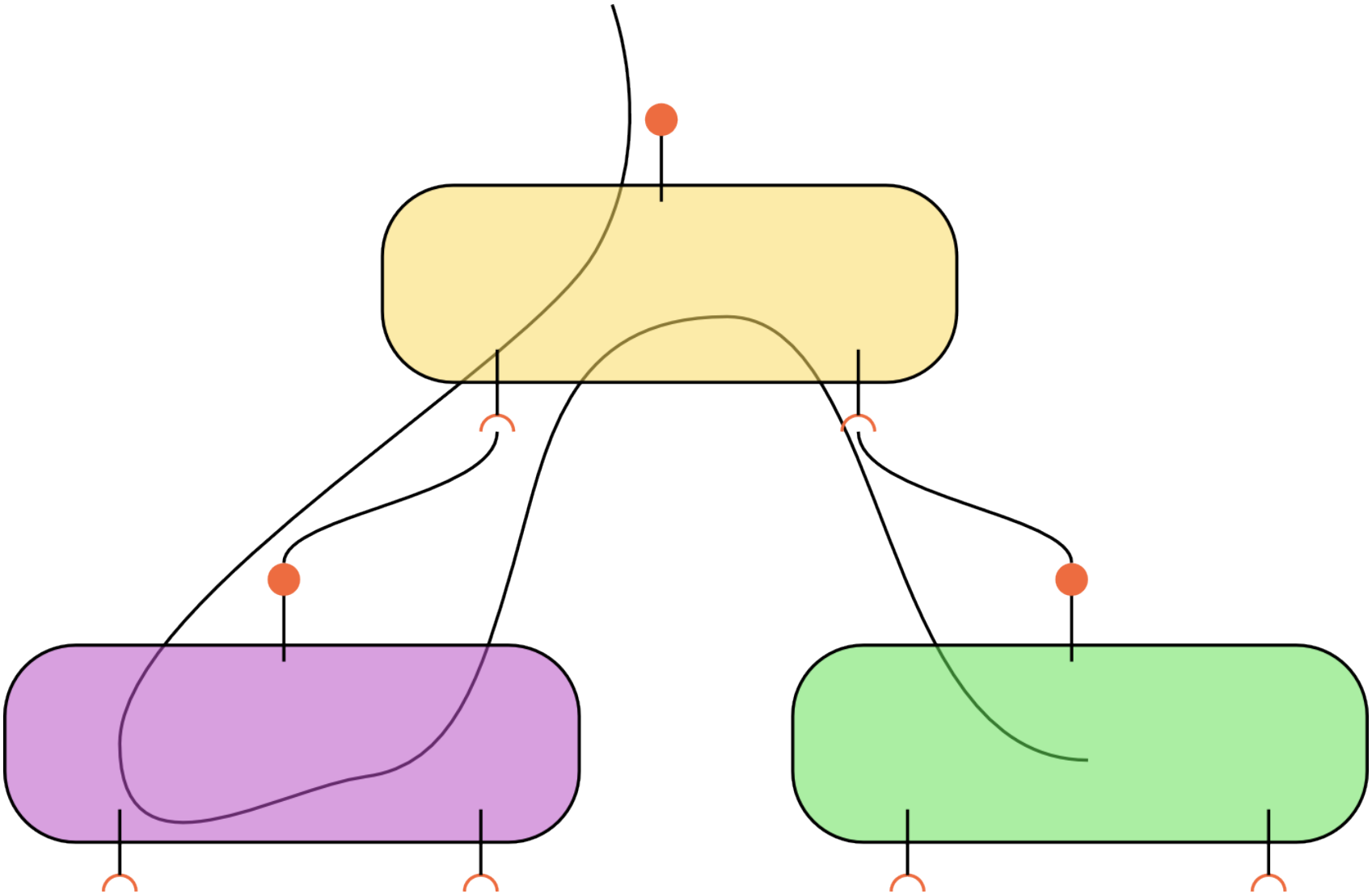
Programowanie funkcyjne daje nowe narzędzia:

1. Funkcje wyższego rzędu
2. Lazy evaluation

Moduły jako funkcje



Lazy evaluation



Plany

"Program napisany przez jedną osobę to mały program. Inaczej by się nie udało".

Plany

Plan: Napisać mały program, który umożliwy budowanie złożonych systemów.

Plany

Plan: Napisać mały program, który umożliwy budowanie złożonych systemów.

1. Implementacja rozproszona

Plany

Plan: Napisać mały program, który umożliwy budowanie złożonych systemów.

1. Implementacja rozproszona
2. Test przeciążenia - warsztaty bez ograniczeń na ilości osób

Plany

Plan: Napisać mały program, który umożliwy budowanie złożonych systemów.

1. Implementacja rozproszona
2. Test przeciążenia - warsztaty bez ograniczeń na ilości osób
3. Injection Bus

Pytania

Pytania?????

Pytania

Pytania:

1. Garbage collector ????

Pytania

Pytania:

1. Garbage collector ????
2. Zmiana kontekstu transakcji ????

Pytania

Pytania:

1. Garbage collector ????
2. Zmiana kontekstu transakcji ????
3. Wyjątki ????

Pytania

Pytania:

1. Garbage collector ????
2. Zmiana kontekstu transakcji ????
3. Wyjątki ????
4. Call reorder - formal specification ????