

Nie tylko agregaty

czyli jak projektować złożoną algebrę domeny

Szymon Janikowski

BO·TT·EGA
IT minds

 Cyclad

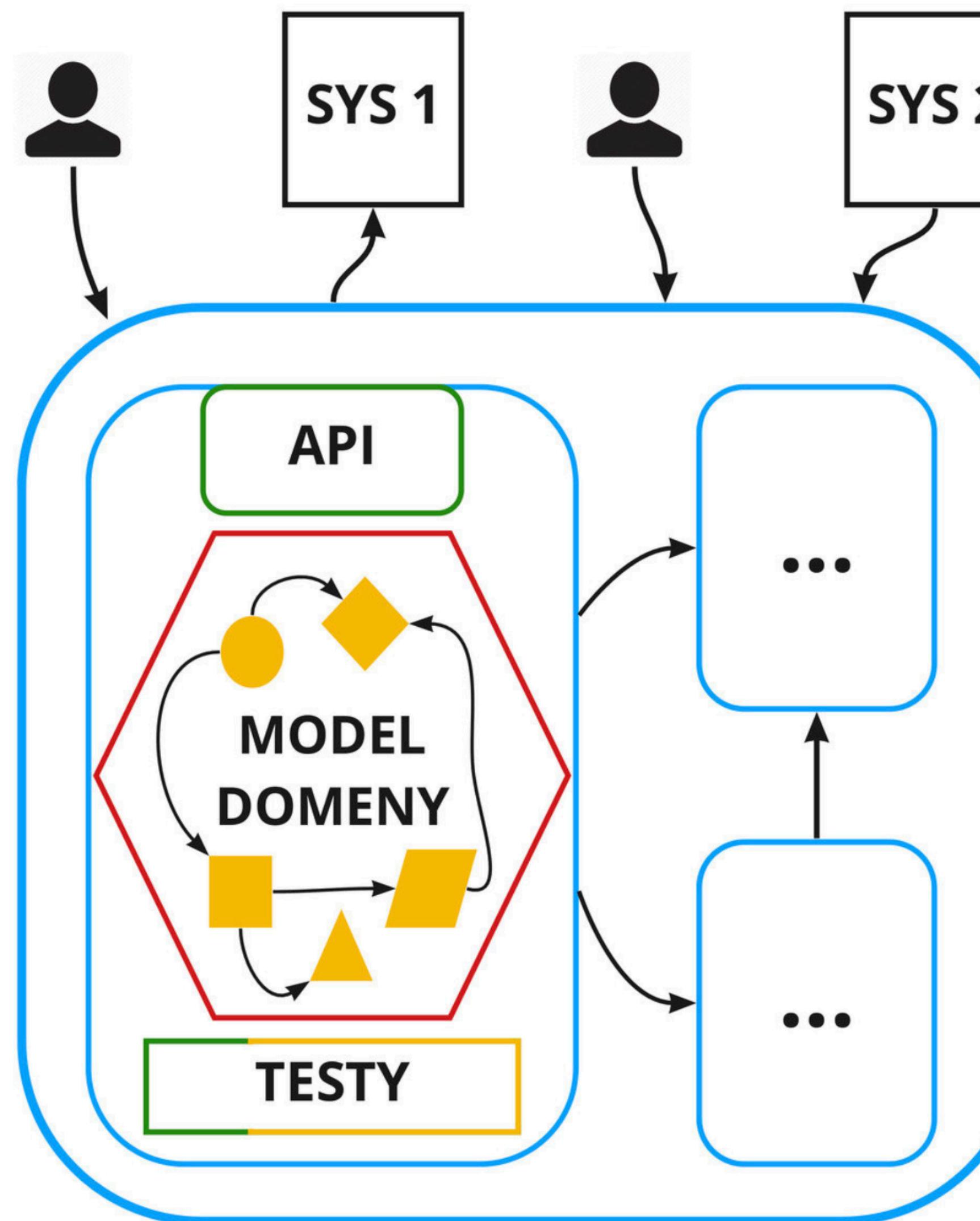


IT LIBRIUM

Uwaga, HEURYSTYKI!



Projektujemy logikę aplikacyjną



Architektura

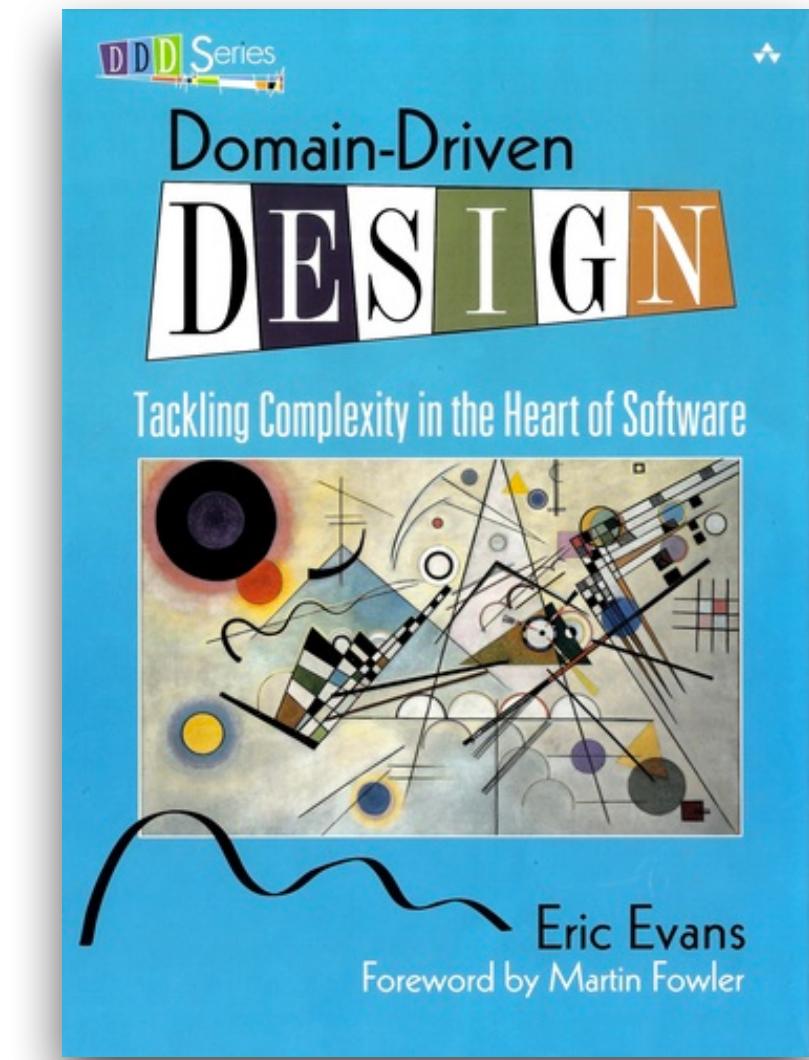
- systemowa
- aplikacyjna

Modelowanie

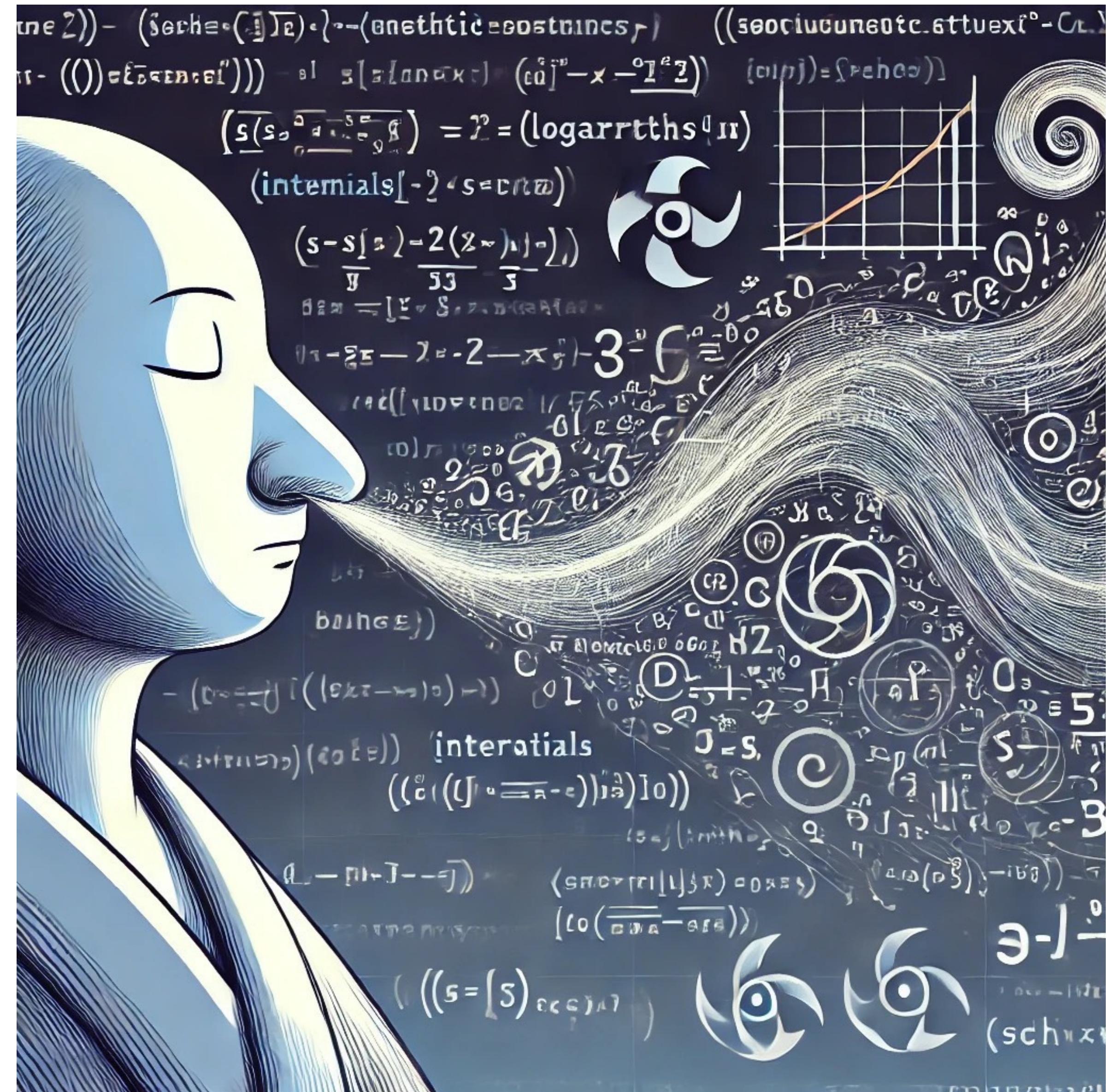
- Strategiczne DDD
- **Taktyczne DDD**
- Event Storming
 - Big Picture
 - Proces Level
 - Design Level
- BDD
- CQRS

„Supple design is the complement to deep modeling. Once you've dug out implicit concepts and made them explicit, you have the raw material.

Evans, Eric. Domain-Driven Design:
Tackling Complexity in the Heart of Software .



NOS



Use case Porównywarka noclegów

Metropolo Krakow by Golden Tulip

Zameldowanie lub wymeldowanie
sob., 21 gru pon., 23 gru 2

Sponsorowane :

Polecane opcje

Metropolo Krakow by Gol... 221 zł Sprawdź cenę >
 Oficjalna strona internetowa OKAZJA
Zapisz się bezpłatnie, by uzyskać 10% zniżki

B Booking.com 221 zł >
Bezpłatne anulowanie do 6 gru

e eSky.pl 222 zł >

Agoda 221 zł >
Bezpłatne anulowanie do 5 gru

Wszystkie opcje i

Metropolo Krakow by Gol... 221 zł Sprawdź cenę >
 Oficjalna strona internetowa
Zapisz się bezpłatnie, by uzyskać 10% zniżki

Reserving 202 zł >
Więcej cen od 217 zł

Obecnie ceny za tę podróż są **niskie**

Orzechowa 11, 30-422 Kraków
goldentulip.com

Hotele w pobliżu Restauracje Atrakcje Bary Kawiarnie Apteki Parkingi

Rzewny Solvay bp Zakopiańska Armatury Sielska Zakołpiantka Orzechowa Jagodowa Goryczkowa

Camping "Krakowianka"

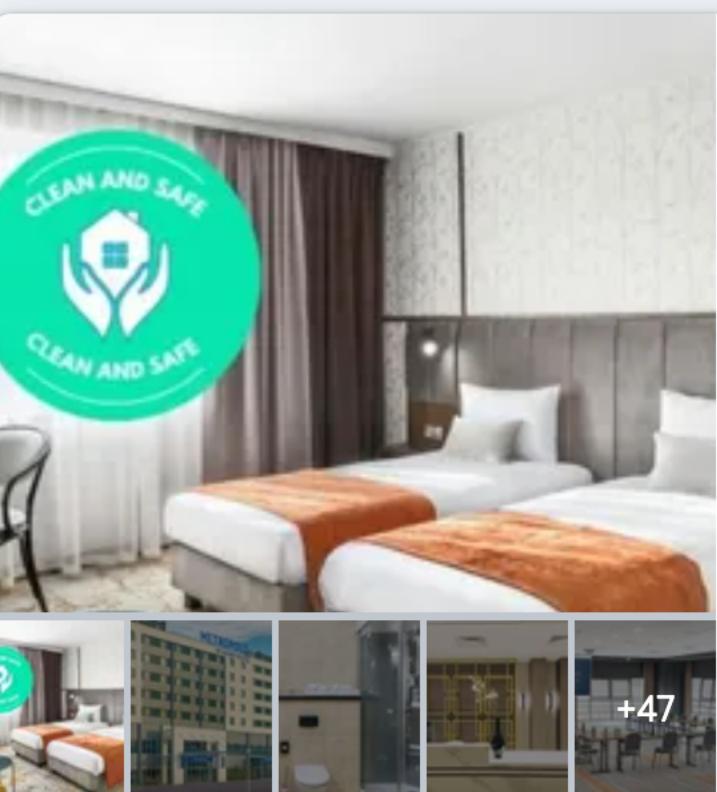
Metropolo Krakow by Golden Tulip 4.6 ★ (2057) Hotel 4-gwiazdkowy

Parking platny

Elżbieta Bierońska Neurologopeda... Hurtownia elektryczna Kraków SPECTRA PRO Aldi

Google

Warstwy ozlewisko potoku Rzewnego



Metropolo Krakow by Golden Tulip

★★★★★

2,06 km od LaLoba Care&Activity

4,5 Opinie: 207

Świetne jeśli pasjonuje Cię/dla: rodziny

Booking.com 230 zł | Trip.com 225 zł | Trip.com 225 zł

[Idź na stronę internetową](#)



Garden Square Hotel

★★★★★

2,51 km od ICE Krakow Congress Centre

5,0 Opinie: 192

Agoda 340 zł | Trip.com 389 zł | Trip.com 387 zł | [Zobacz więcej](#)

[Idź na stronę internetową](#)



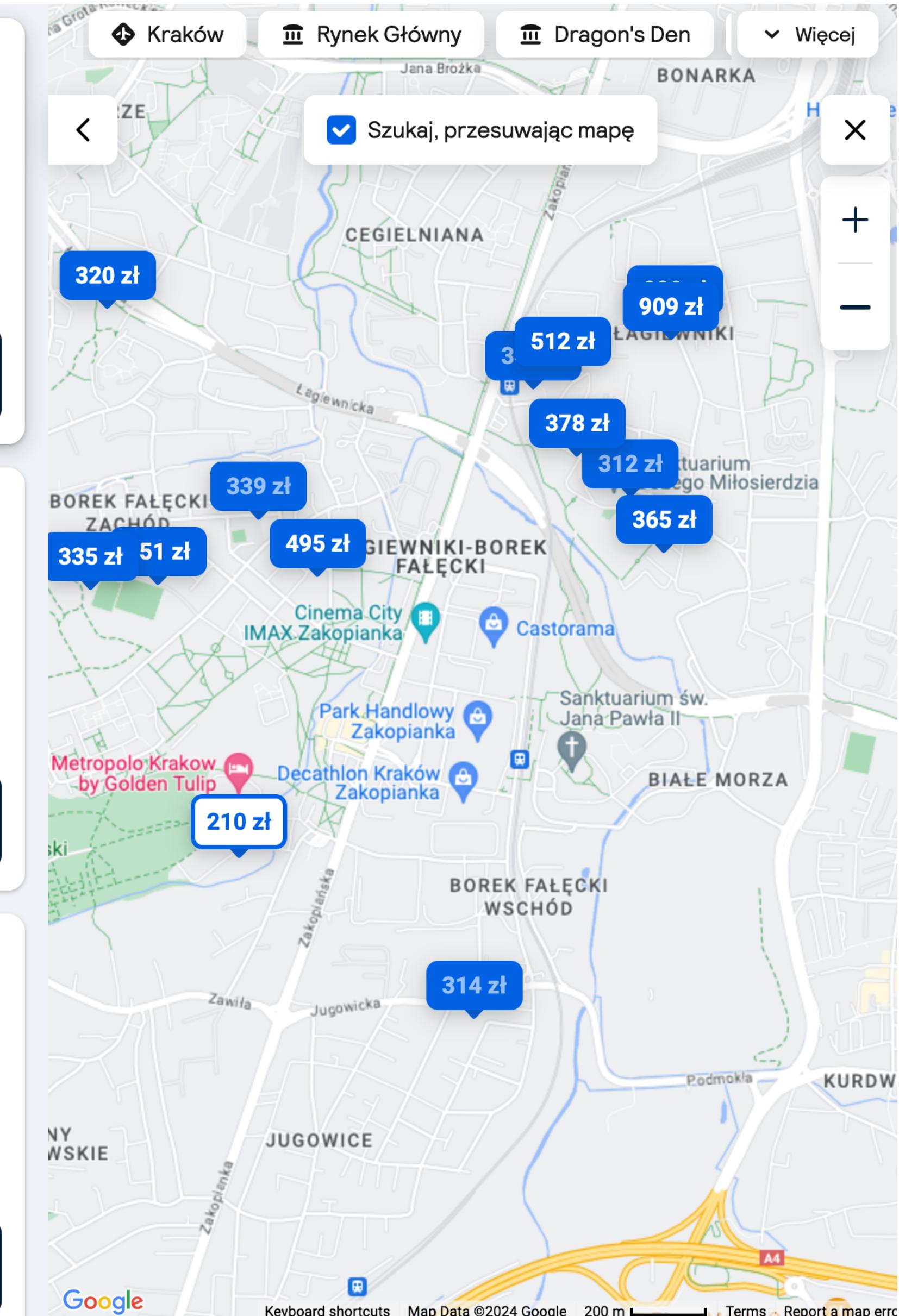
Pastoral Lodge at the Sanctuary of the Divine Mercy

★★

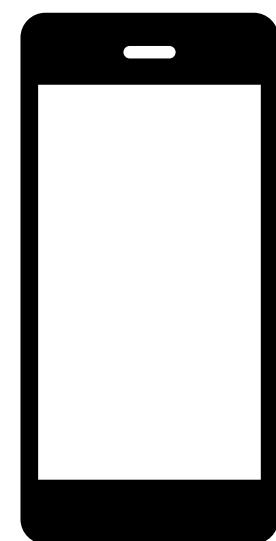
2,83 km od ICE Krakow Congress Centre

Booking.com

[Idź na stronę internetową](#)

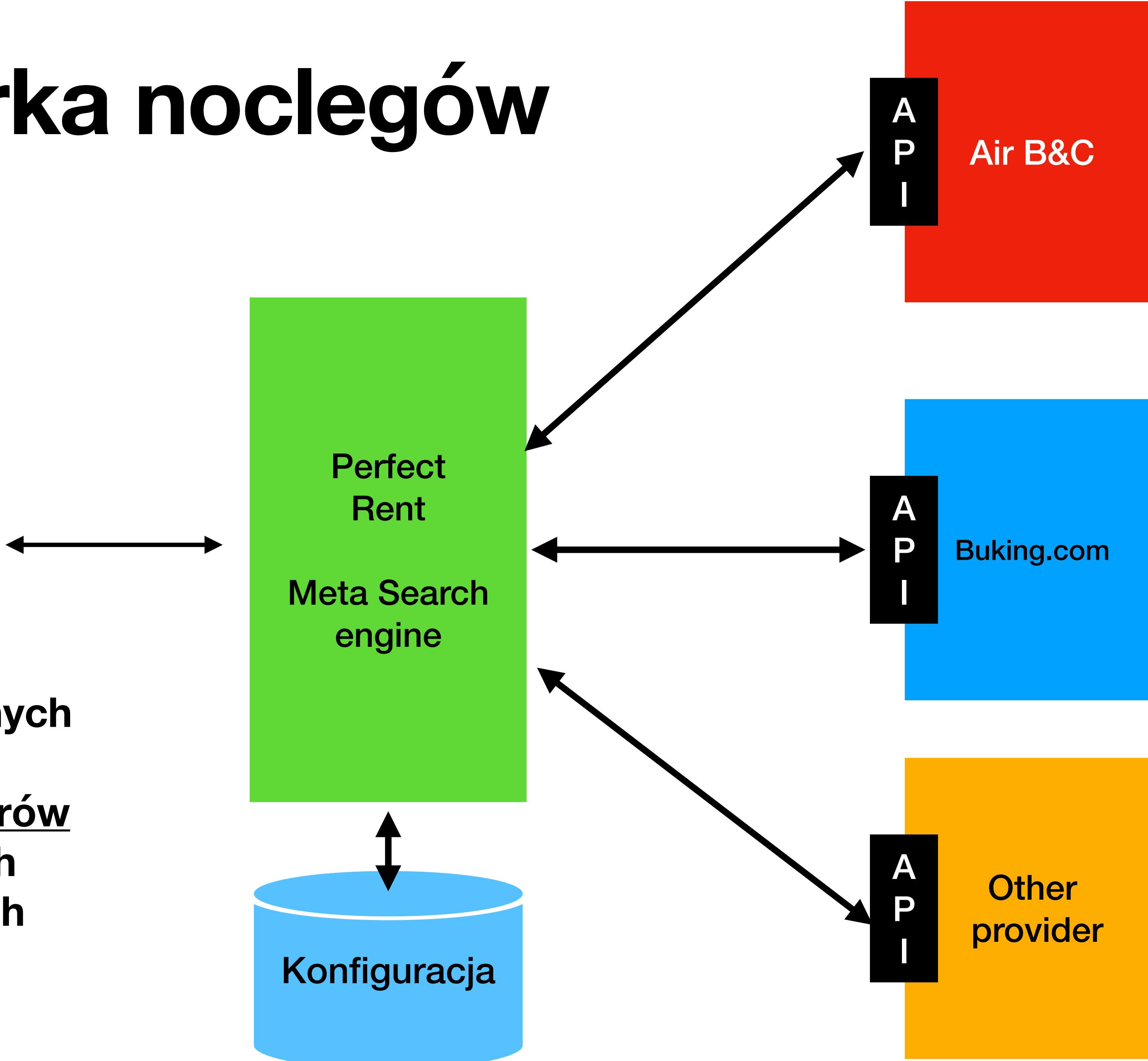


Porównywarka noclegów



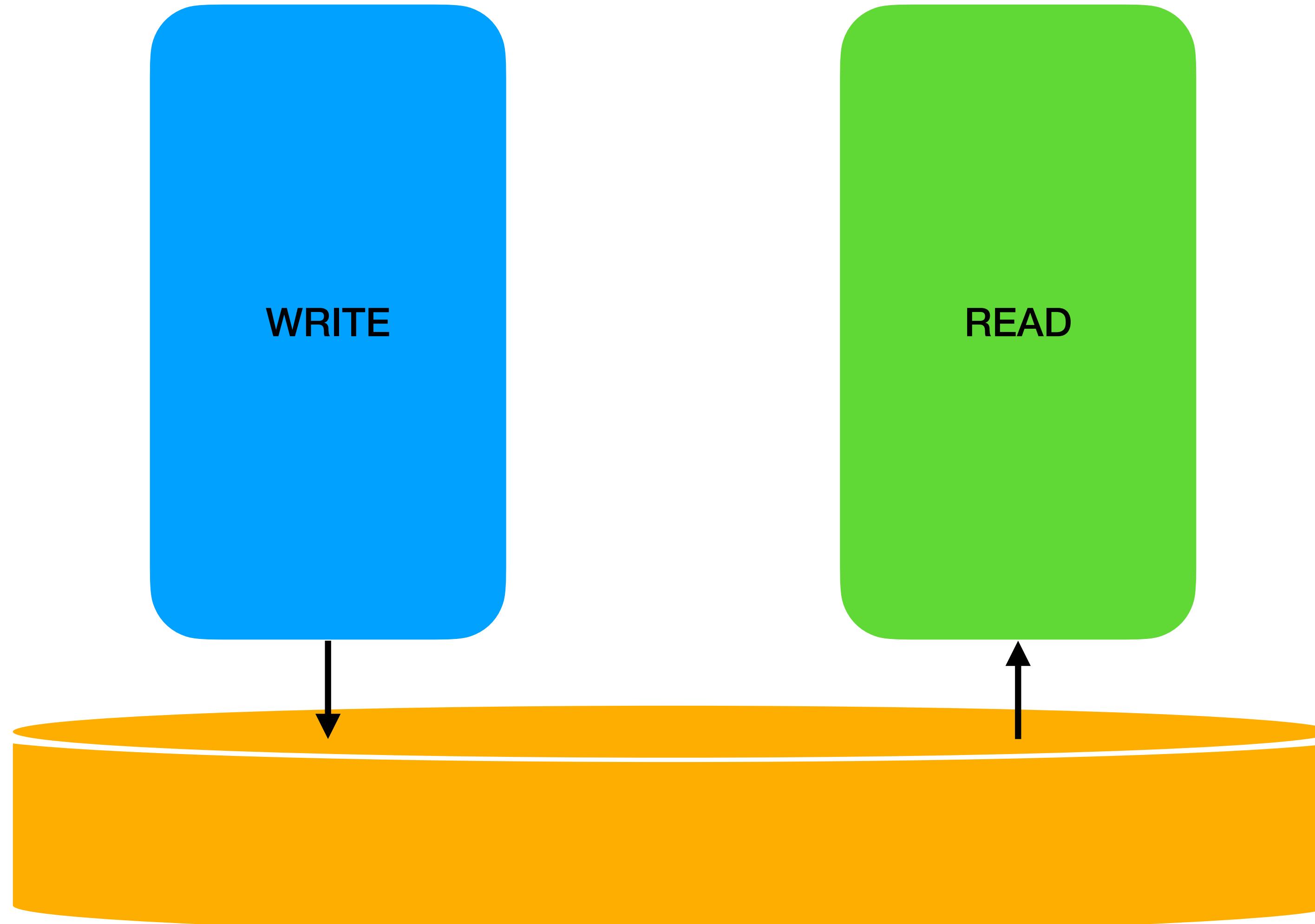
Porównanie ofert od różnych dostawców

Brak ryzykownych obszarów
+ ukrywanie niektórych dostawców przy dużych różnicach cen



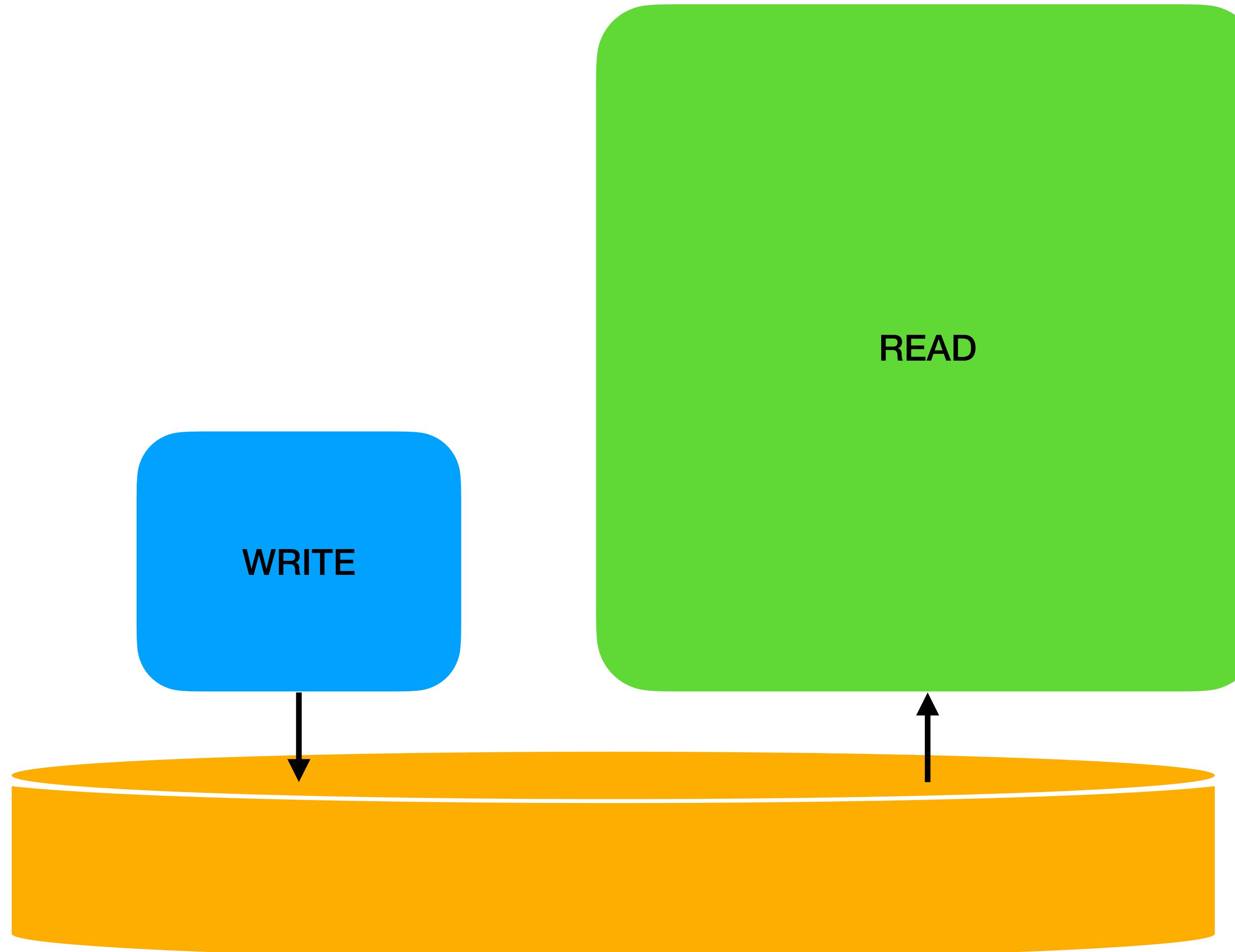
CQS

Command Query
Separation



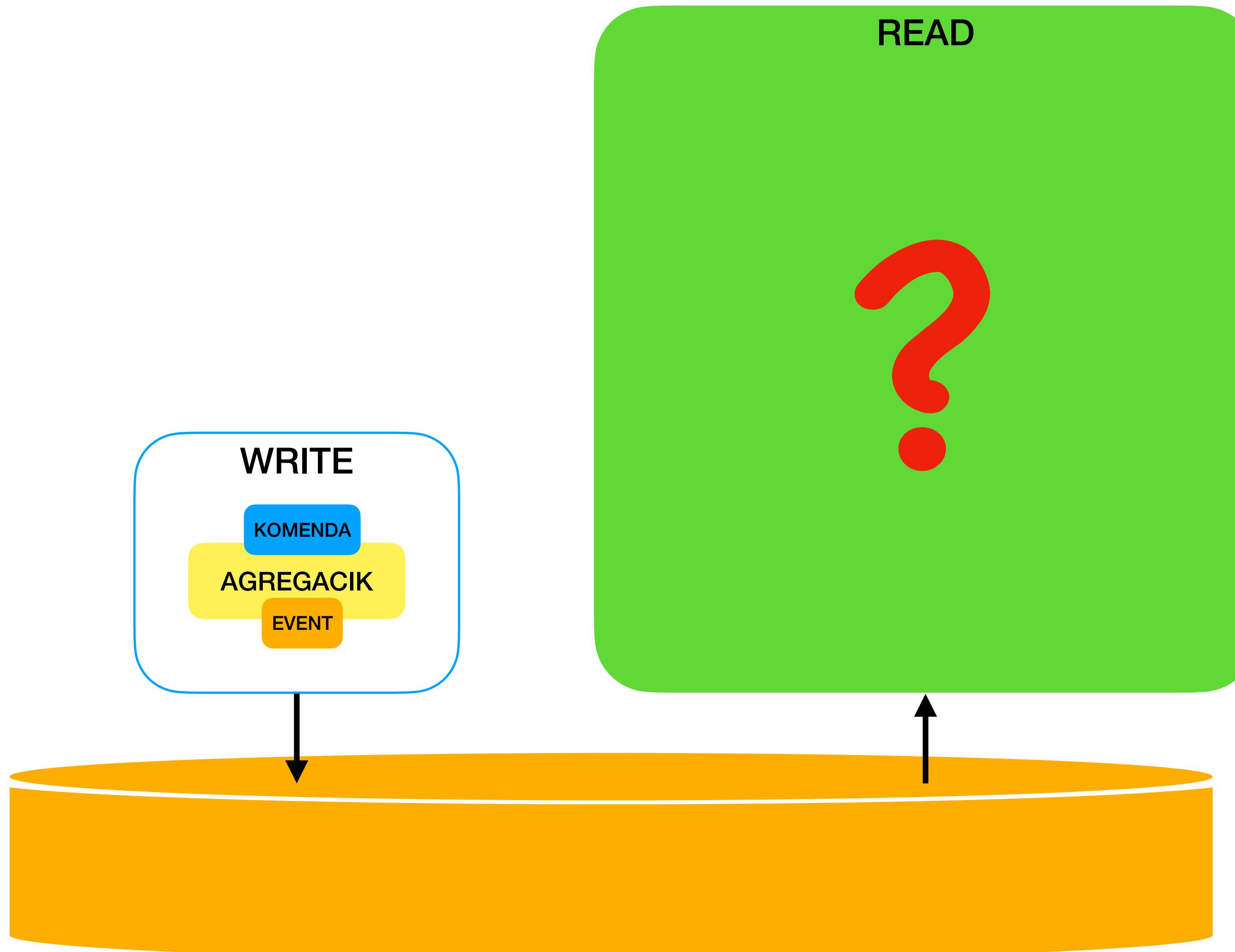
CQS

Command Query
Separation

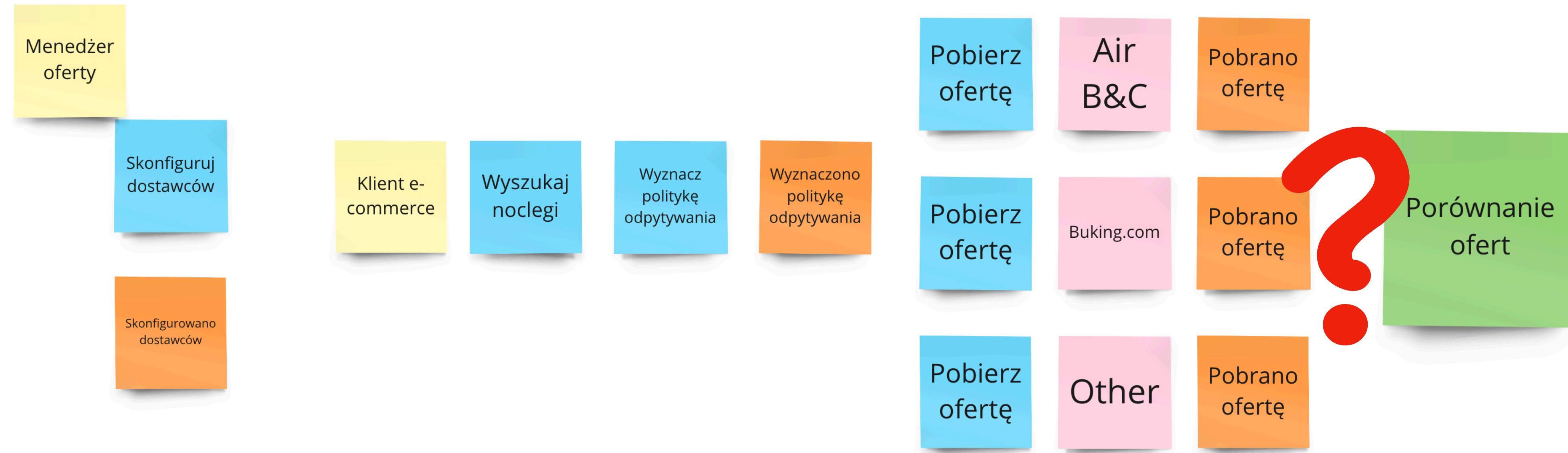


CQS

Command Query
Separation



Jak to wygląda na Event Stormingu?

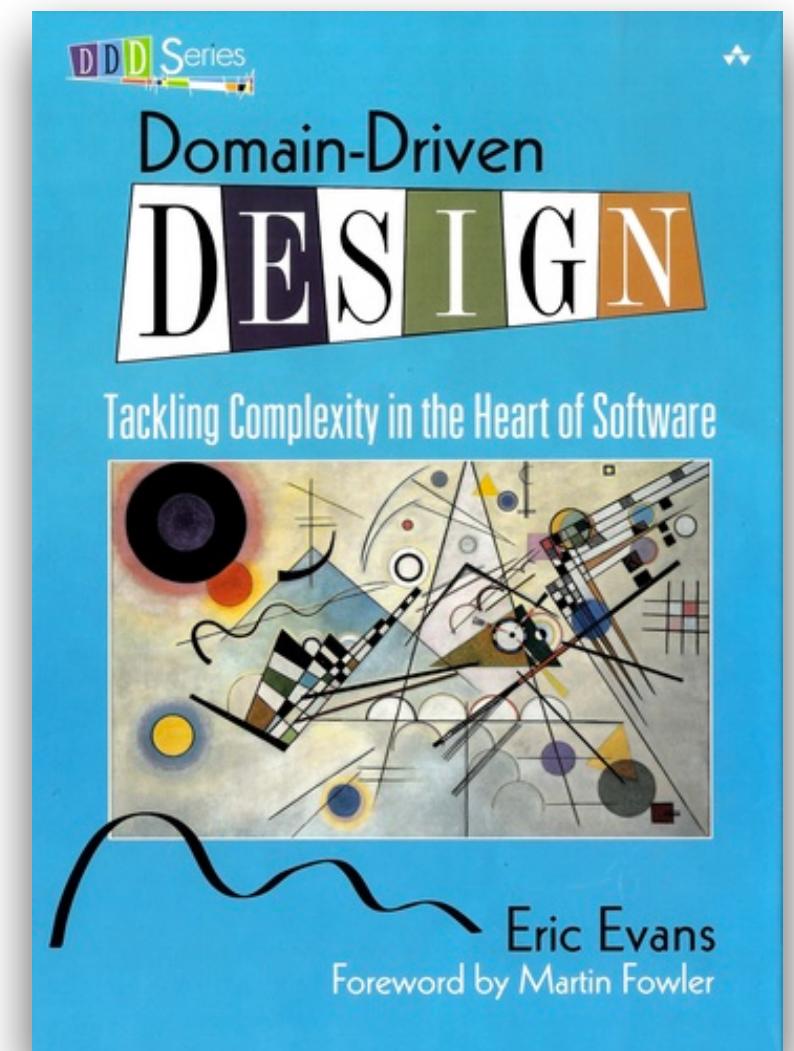


Jak to projektować?

Jaki powinien być MODEL?

„There are many (...) formalized conceptual frameworks, but my personal favorite is math. (...) Many domains include math somewhere. Look for it. Dig it out. Specialized math is clean, combinable by clear rules, and people find it easy to understand.”

Evans, Eric. Domain-Driven Design:
Tackling Complexity in the Heart of Software

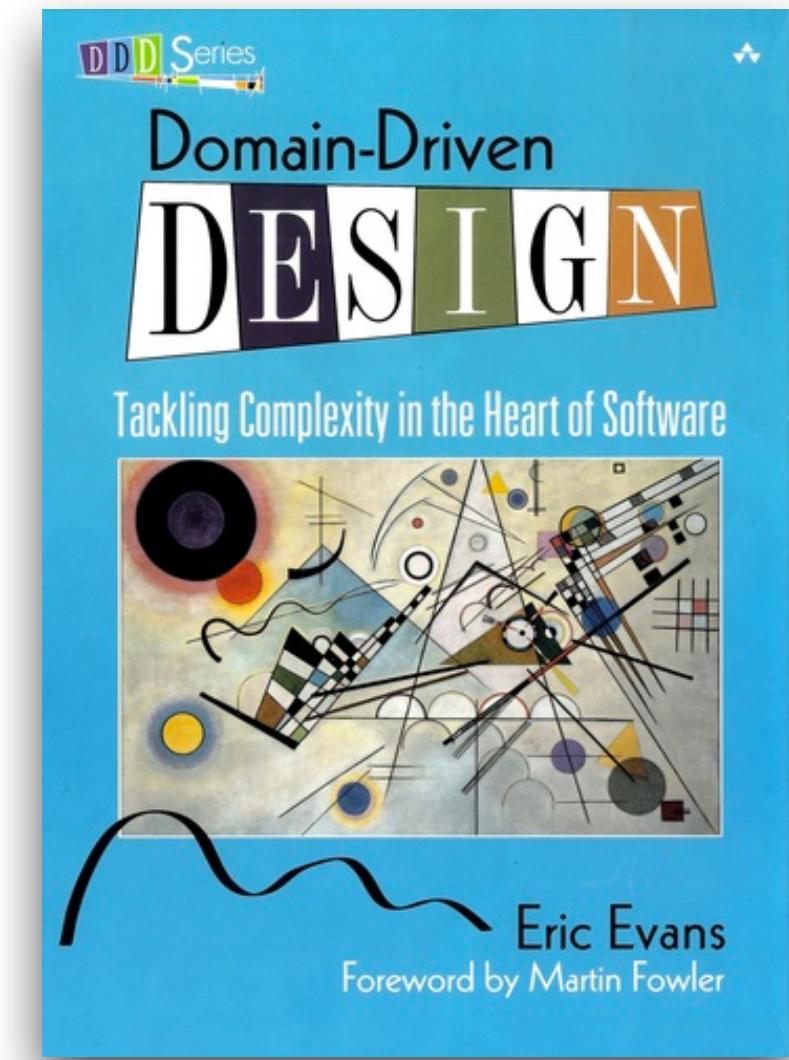


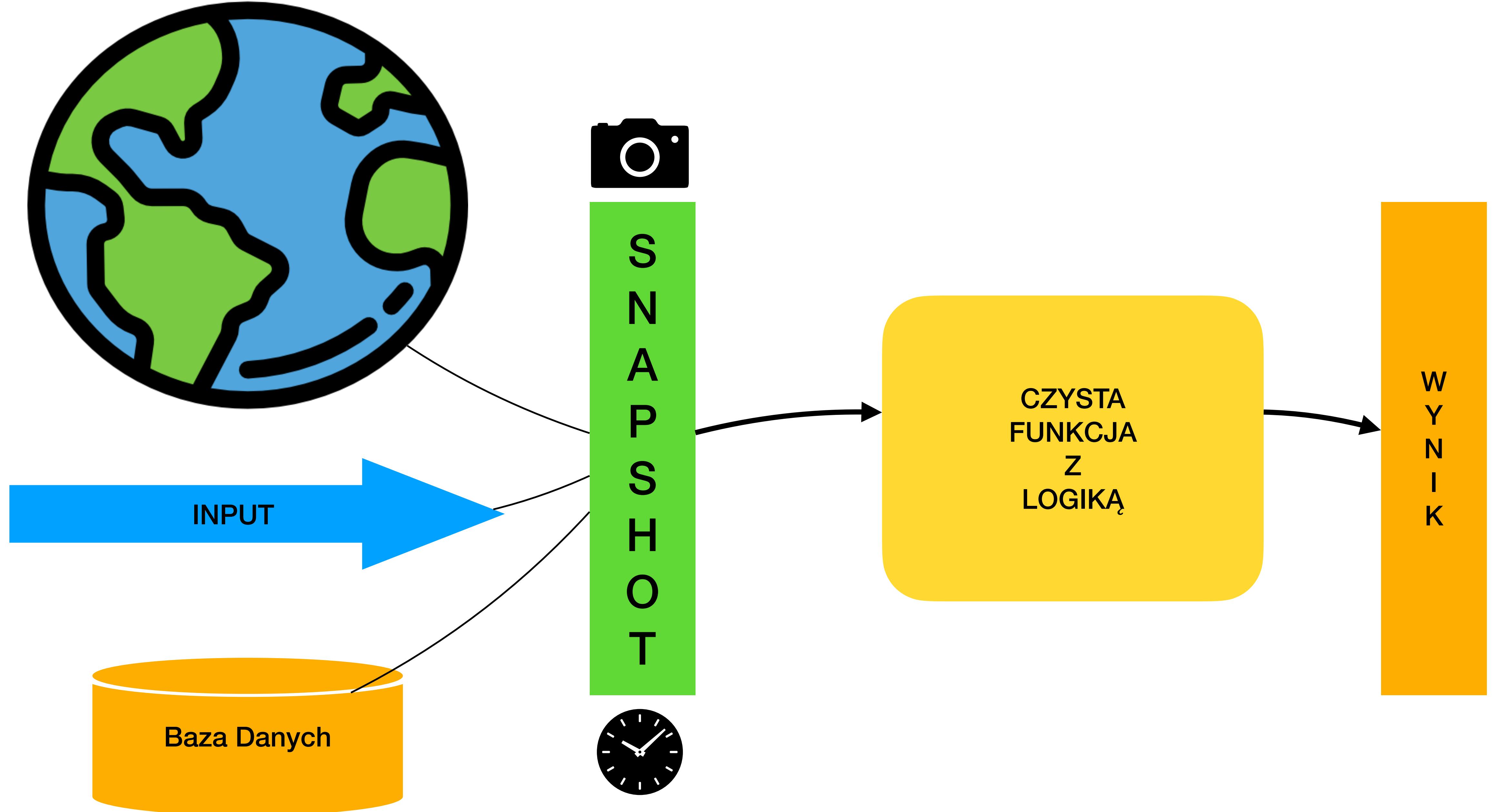
Projekcja, agregacja...

Najlepiej gdy zmaxsymalizujemy użycie
CZYSTYCH FUNKCJI!

„Functions lower risk”

**Evans, Eric. Domain-Driven Design:
Tackling Complexity in the Heart of Software**

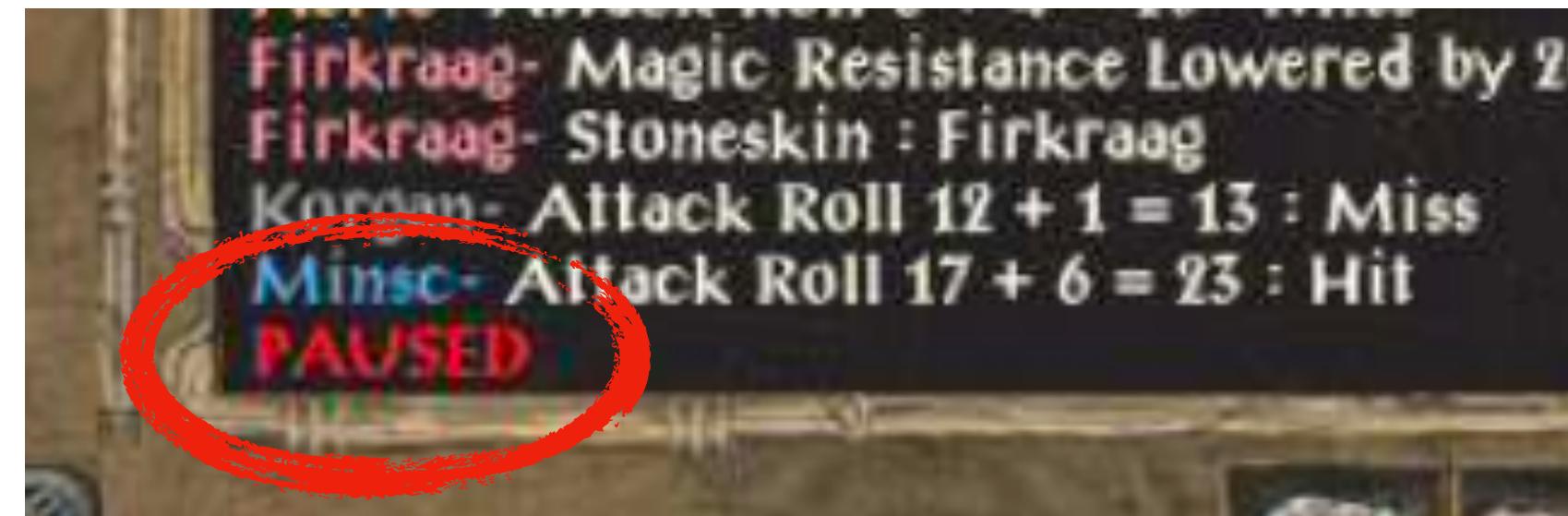




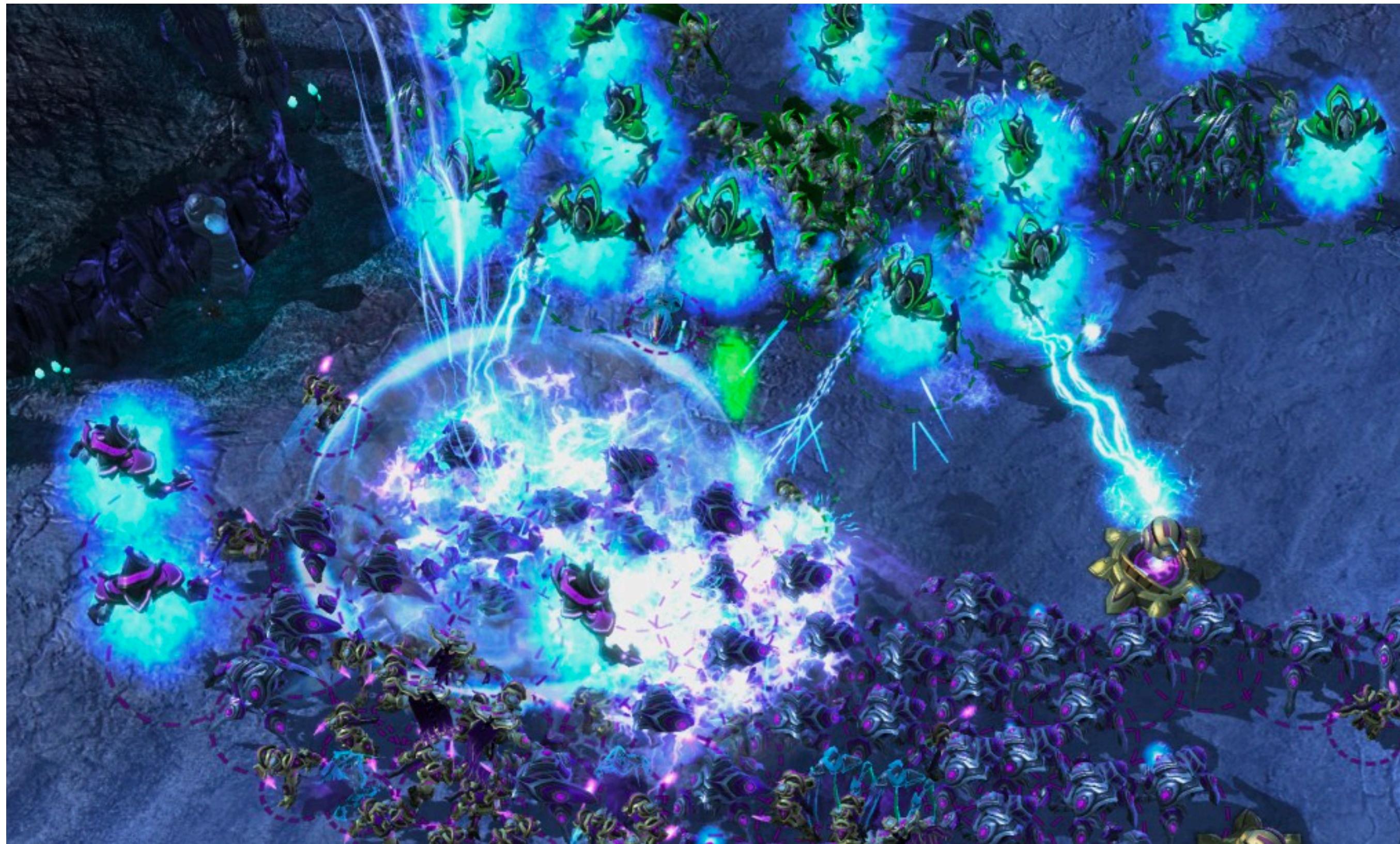
„Weźmy dowolne ale ustalone...”

Twój matematyk

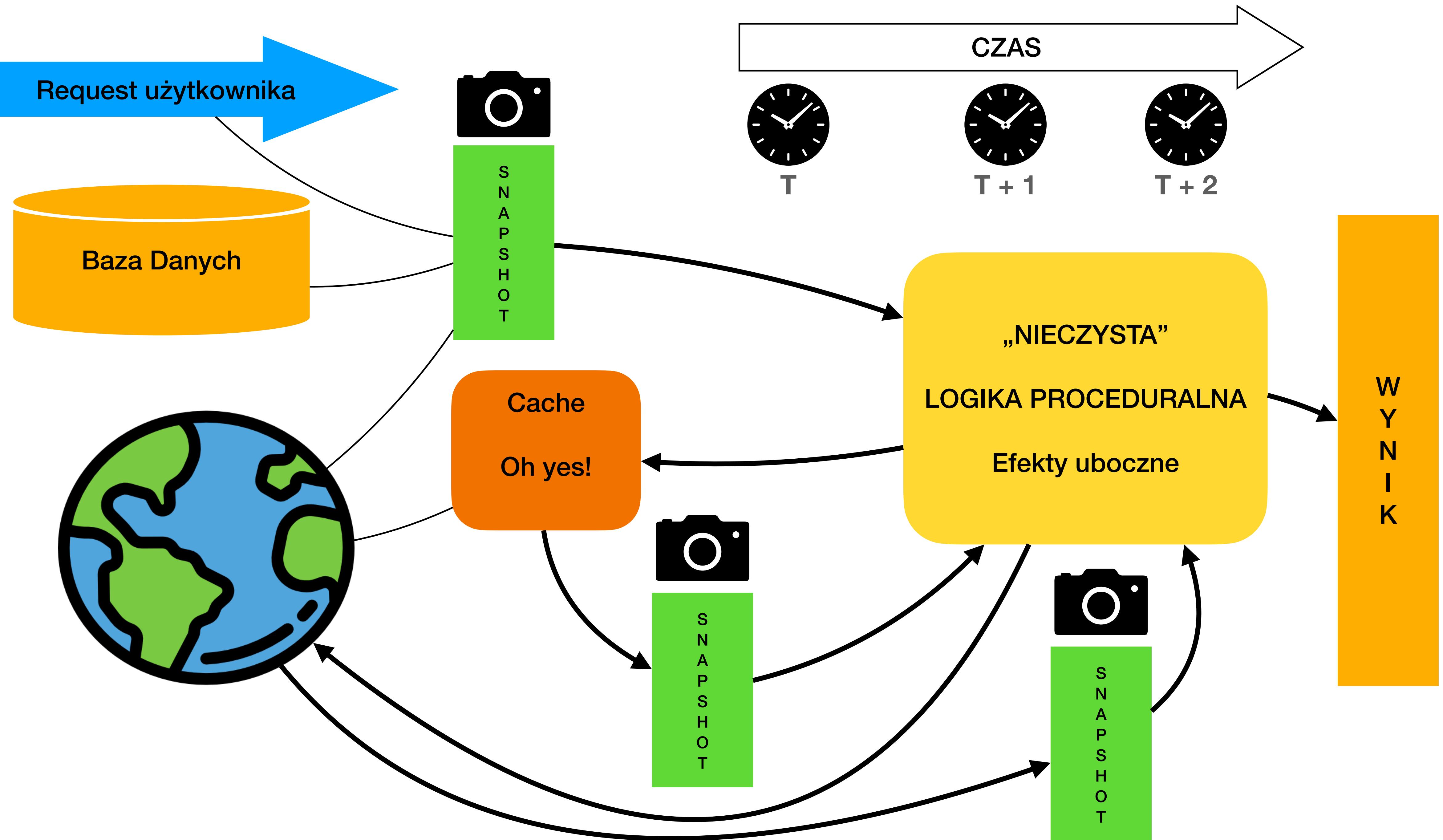
Auto-pauza 😊

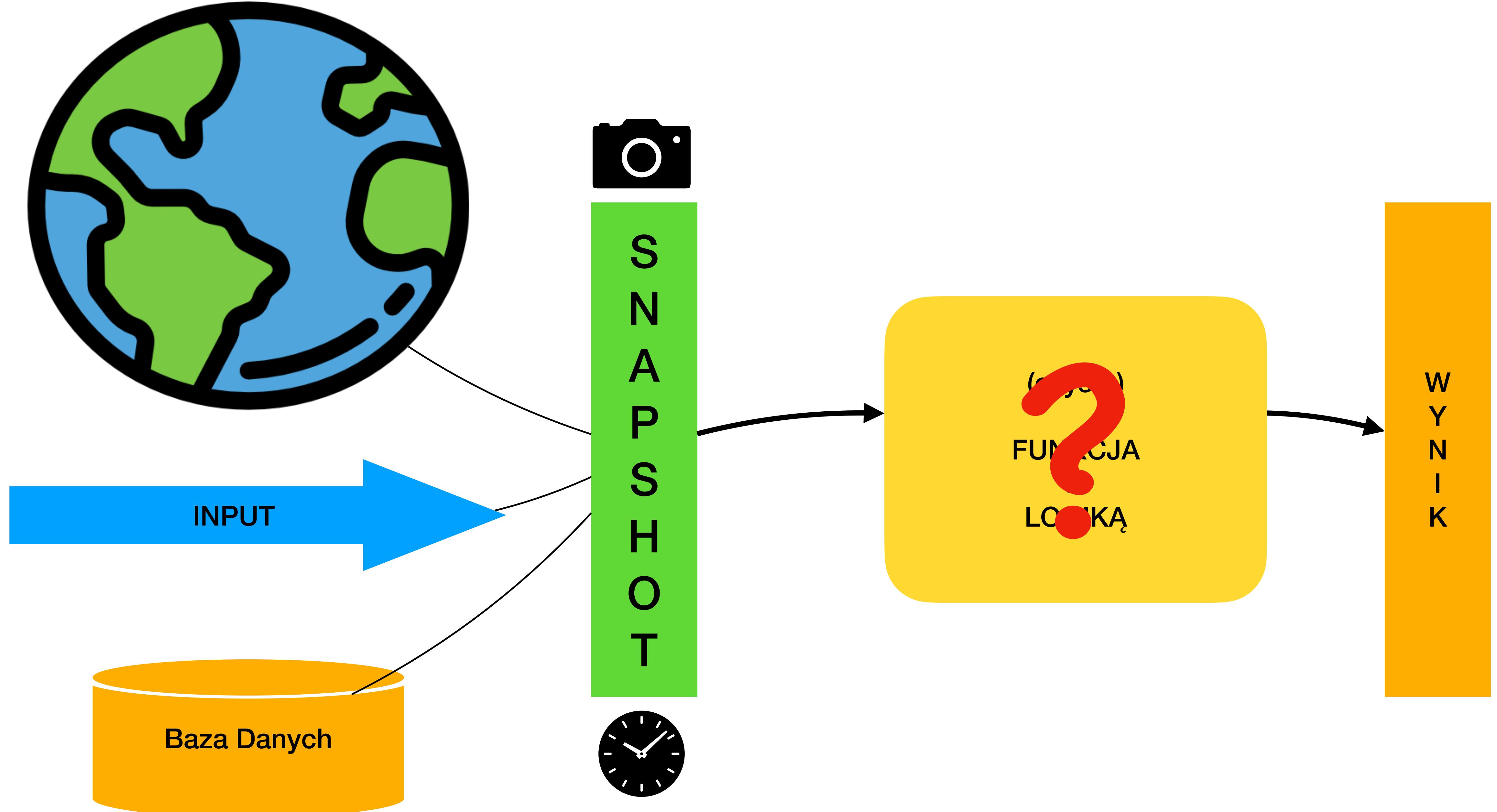


Możesz bawić się w RTS...



...ale czy chcesz? 😐





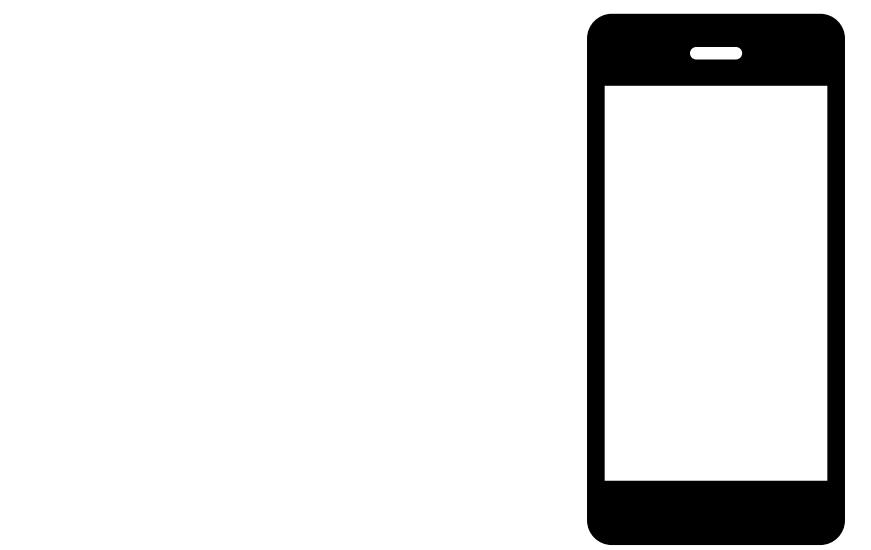
Jak to rozwiązać?

- Znaleźć **POWIĄZANIA** między danymi
- Opracować **MODEL** (algebraiczny, matematyczny, logiczny)
- Zadbać o **NAZEWNICTWO**

**Jakie powiązania w
porównywarkę?**

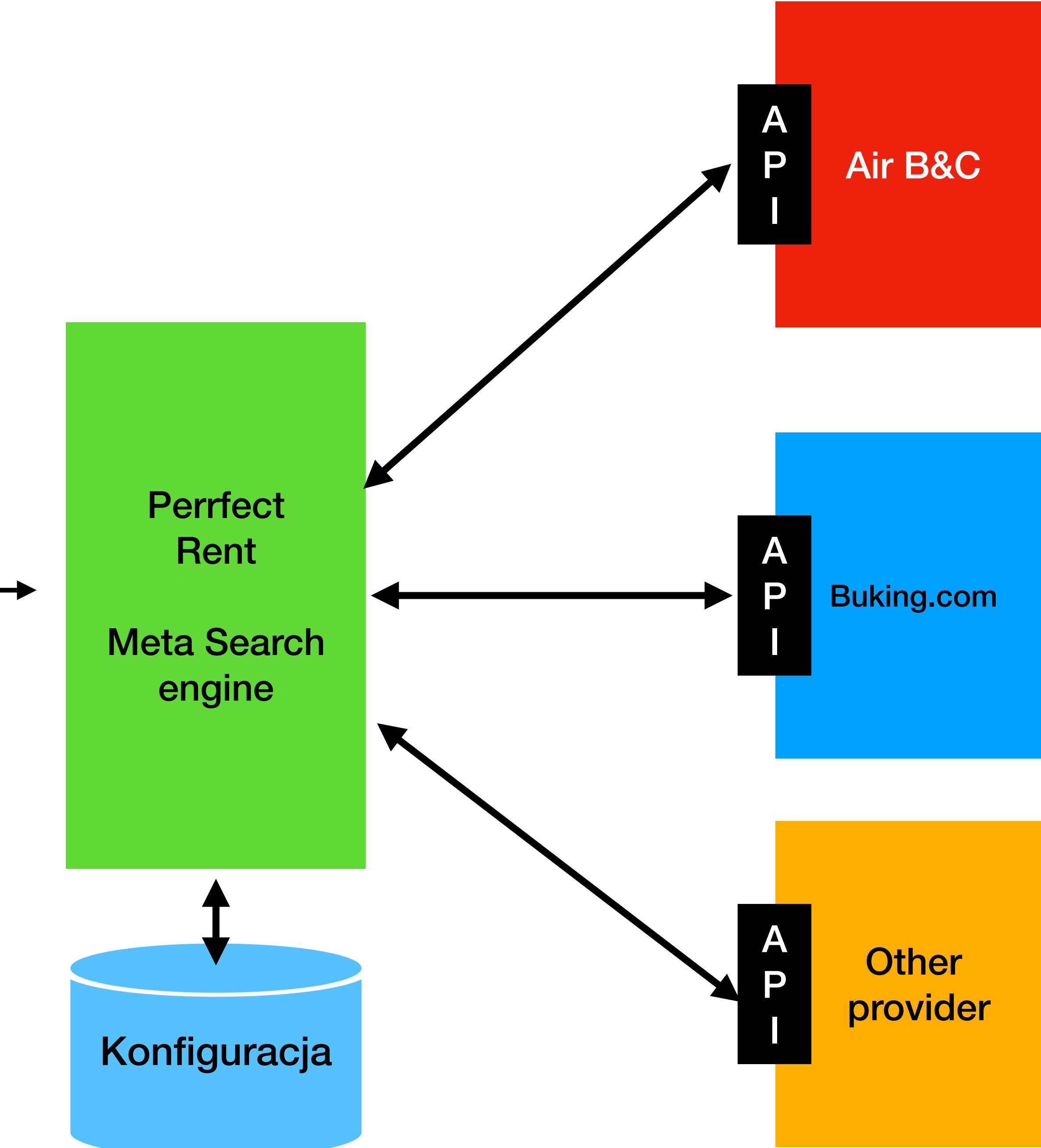
Jakie zależności między danymi?

QUESTION

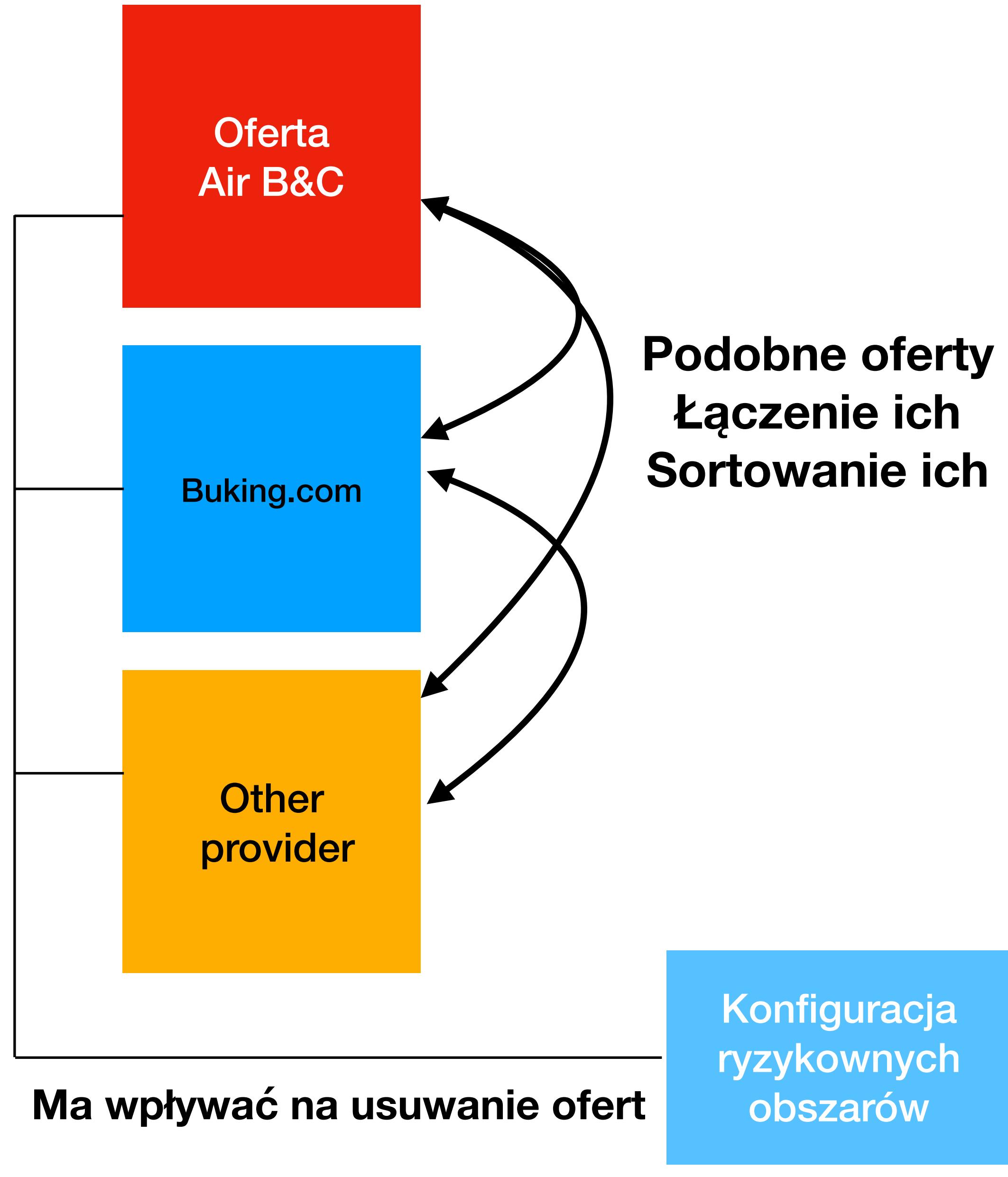


Porównanie ofert od różnych dostawców

Brak ryzykownych obszarów
+ ukrywanie niektórych dostawców przy dużych różnicach cen



SNAPSHOT



Najprościej zrobić to filtrem

Ustandaryzowane oferty od dostawcy Z

Ustandaryzowane oferty od dostawcy Y

Ustandaryzowane oferty od dostawcy X

Filtr ryzykownych obszarów

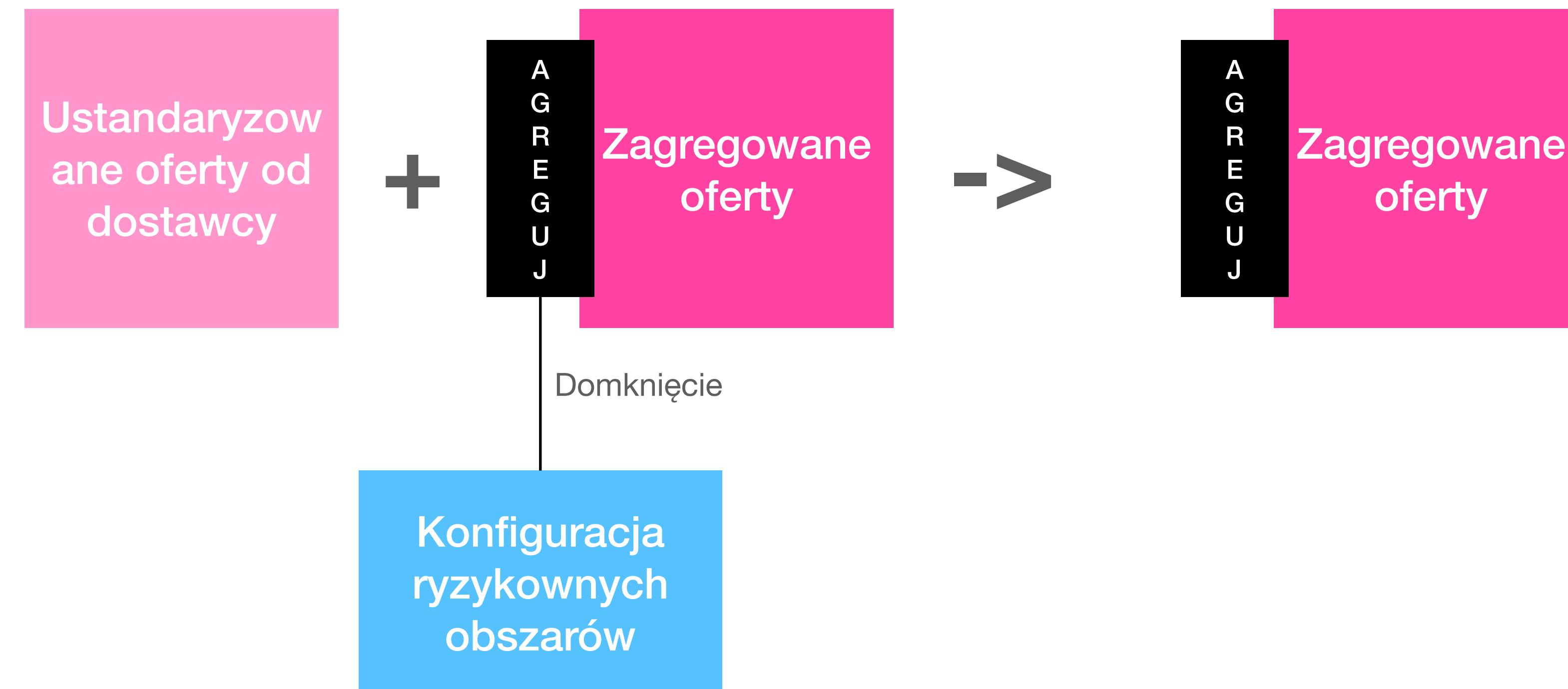
A
G
R
E
G
U
J

Zagregowane oferty

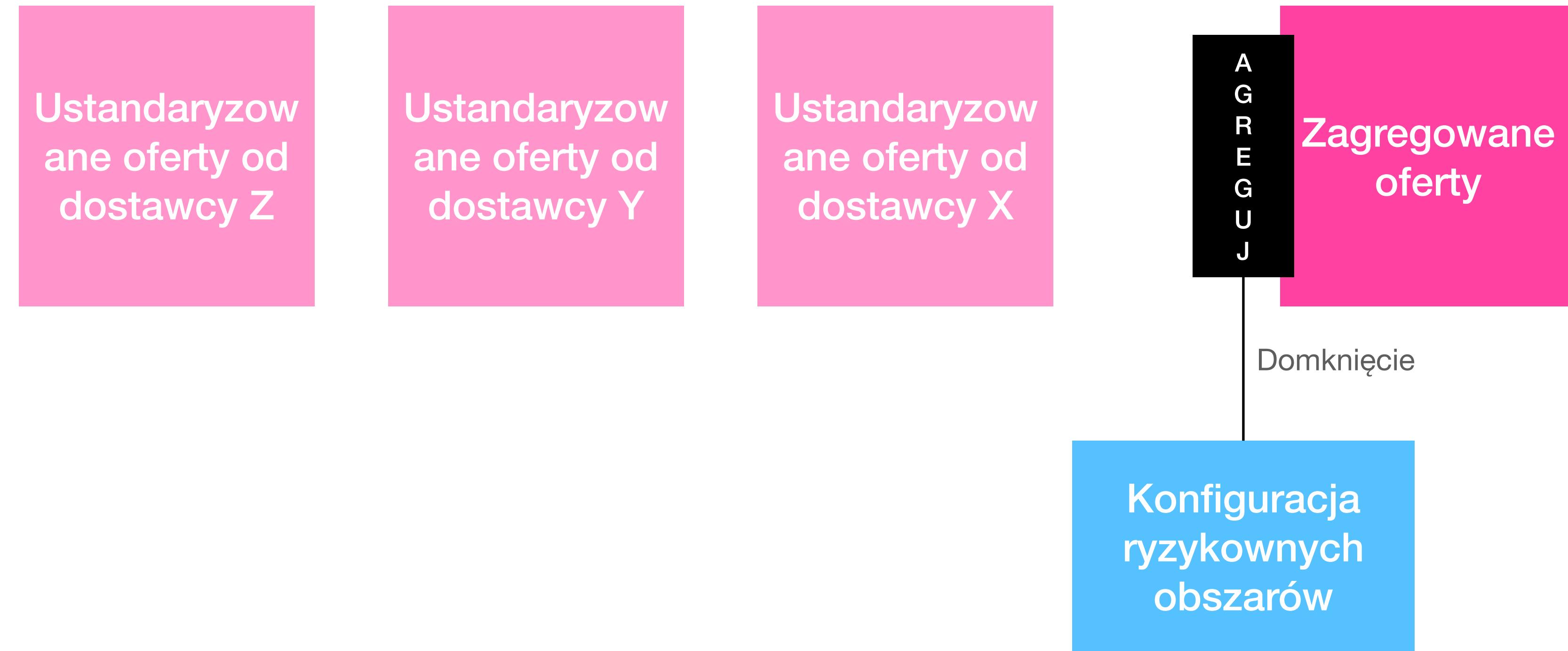
Domknięcie

Konfiguracja ryzykownych obszarów

Domknięcie + fold left - alternatywnie



Domknięcie + fold left



KOD

Wybieramy jednokrotne pobranie konfiguracji
na potrzeby obliczeń w kontekście obsługi
danej komendy
(DOMKNIĘCIE)

zamiast

Odpytywania o każdą lokalizację z osobna

Co już wykorzystaliśmy z programowania funkcyjnego?

- **Niemutowalność (Value Objects)** - do panowania nad przetwarzanymi danymi
- **Fold left (reduce)** - do modelowania pewnej „algebry” wbudowanej w Domenę
- **DOMKNIĘCIE** - pewne funkcje uzupełniamy niemutowalnymi danymi (snapshot świata)

Co już wykorzystaliśmy z programowania obiektowego?

- Modelowanie obiektów:
 - bogatych w zachowania (**Value Objects**)
 - posiadających odpowiednie NAZWY
- Interfejsy „zdradzające intencję” (**Intention revealing interfaces** - Beck / Evans) - obejmujące istotne fragmenty logiki
- Closure of operations (Evans) - operacje na obiekcie zwracają typ tego obiektu - np. Aggregate użyte w fold left
- Deklaratywny wzorzec Specyfikacji (**Predykatu**)

Jakie rodzaje struktur danych?

Jakie struktury danych?

Wejściowa

Wyjściowa
mutowalna

Wejściowa

Pośrednia
mutowalna

Wyjściowa

Wejściowa

Wiele pośrednich
niemutowalnych

Wyjściowa

Jakie struktury danych?

Wejściowa

Wyjściowa
mutowalna

Wejściowa

Pośrednia
mutowalna

Wyjściowa

Wejściowa

Wiele pośrednich
niemutowalnych

Wyjściowa

**Dodatkowe, „drobne” wymagania
filtrowania „odstających” ofert** 😈

Dodatkowe wymagania...

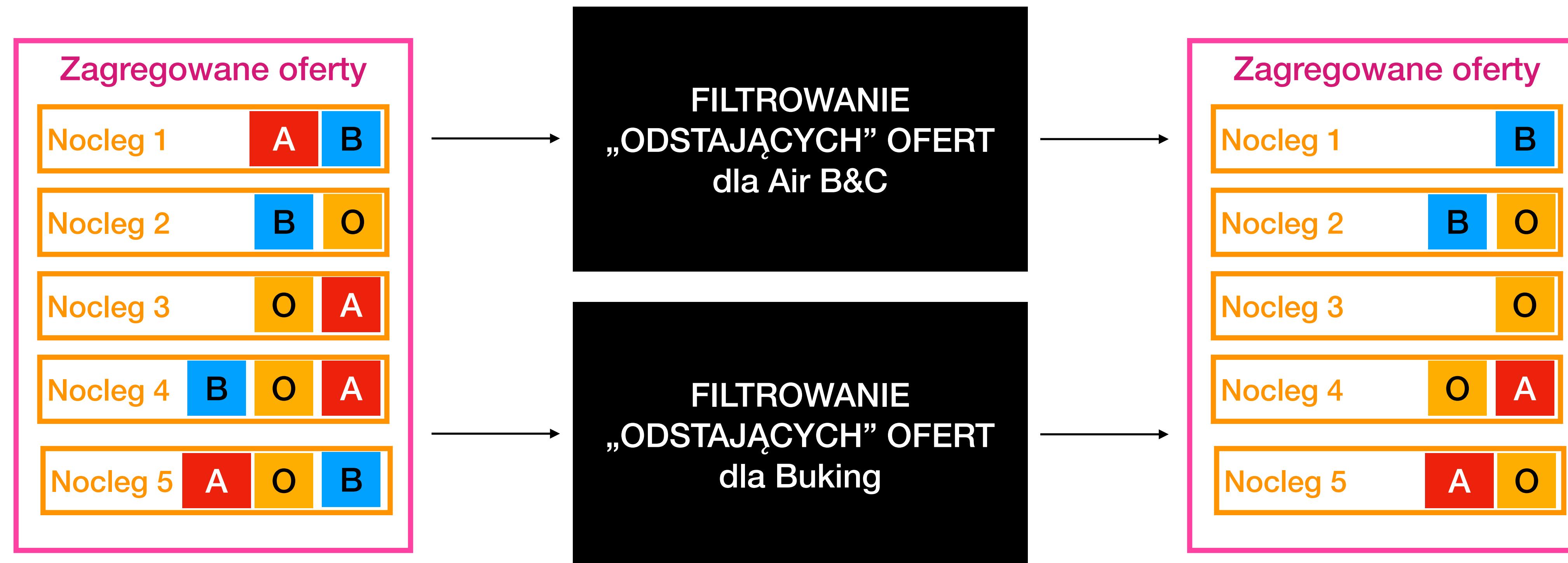
Zagregowane oferty			
Nocleg 1	A	B	
Nocleg 2	B	O	
Nocleg 3	O	A	
Nocleg 4	B	O	A
Nocleg 5	A	O	B

- Wybrani dostawcy życzą sobie, żeby ich oferta **nie pojawiała się** w rankingu jeśli jest **więcej niż 10% droższa** niż konkurencji.
- Jednocześnie wymagają, że ich **oferta ma się pojawić chociaż raz w pierwszych 3 wynikach** nawet jeśli jest droższa (o ile jakkolwiek zwróciły)
- Jednocześnie niektórzy wymagają, żeby **niektóre reguły nie były stosowane w weekendy**

Niby proste...

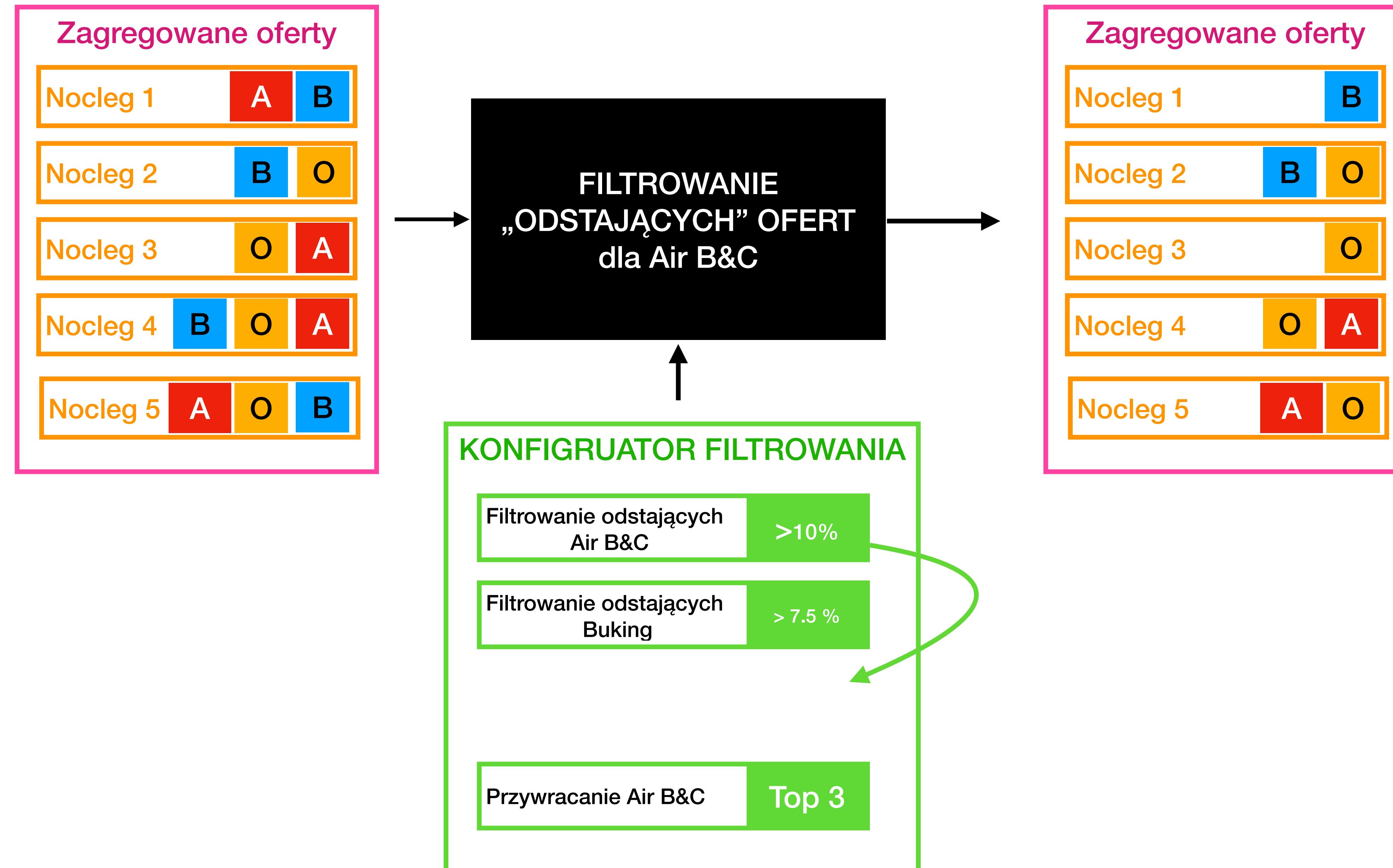


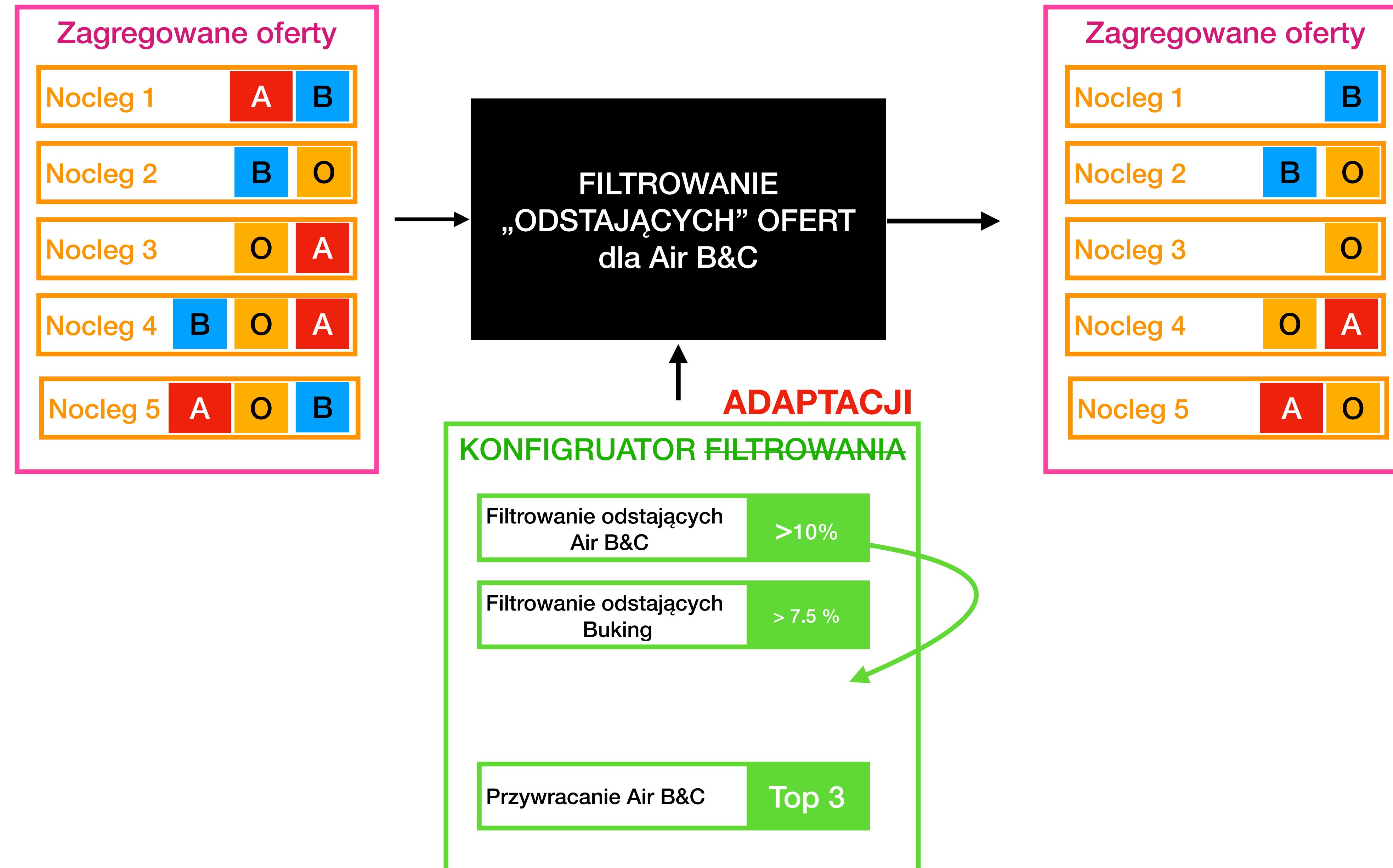
Ale czy tak się da?



JUŻ WYKLUCZONE OFERTY WPŁYWAJĄ
NA KOLEJNE WYKLUCZANE OFERTY!







SNAPSHOT

INPUT



KONFIGRUATOR
ADAPTACJI

Zagregowane oferty

Nocleg 1	A	B	
Nocleg 2	B	O	
Nocleg 3	O	A	
Nocleg 4	B	O	A
Nocleg 5	A	O	B

IsWeekend: true

Filtrowanie odstających
Air B&C >10%

Filtrowanie odstających
Buking > 7.5 %

Przywracanie Air B&C Top 3

(czysta)
**FUNKCJA
Z
LOGIKĄ**

Przefiltrowane oferty

Nocleg 1	B		
Nocleg 2	B	O	
Nocleg 3	O		
Nocleg 4	B	O	A
Nocleg 5	A	O	B



INPUT



Konfiguracja

KONFIGRUATOR
ADAPTACJI

SNAPSHOT

Zagregowane oferty

Nocleg 1	A	B	
Nocleg 2	B	O	
Nocleg 3	O	A	
Nocleg 4	B	O	A
Nocleg 5	A	O	B

KOLEJNOŚĆ
OFERT

IsWeekend: true

Filtrowanie odstających
Air B&C >10%

Filtrowanie odstających
Buking > 7.5 %

Przywracanie Air B&C Top 3

KOLEJNOŚĆ
ADAPTACJI

Wpływają na to jakie
filtry mają być zastosowane

Zadaptowane oferty

Nocleg 1	B		
Nocleg 2	B	O	
Nocleg 3	O		
Nocleg 4	B	O	A
Nocleg 5	A	O	B

JUŻ WYKLUCZONE OFERTY WPŁYWAJĄ
NA KOLEJNE WYKLUCZANE OFERTY!



INPUT



SNAPSHOT

Zagregowane oferty

Nocleg 1	A	B	
Nocleg 2	B	O	
Nocleg 3	O	A	
Nocleg 4	B	O	A
Nocleg 5	A	O	B

IsWeekend: true

Filtrowanie odstających
Air B&C >10%

Filtrowanie odstających
Buking > 7.5 %

Przywracanie Air B&C
Top 3

KONFIGRUATOR
ADAPTACJI

Oferty z WYKLUCZENIAMI

Nocleg 1	A	B	
Nocleg 2	B	O	
Nocleg 3	O	A	
Nocleg 4	B	O	A
Nocleg 5	A	O	B

APPLY

Filtrowanie odstających
Air B&C >10%

APPLY

Filtrowanie odstających
Buking > 7.5 %

APPLY

Przywracanie Air B&C Top 3

APLIKOWALNE ADAPTACJE

Adaptowane oferty

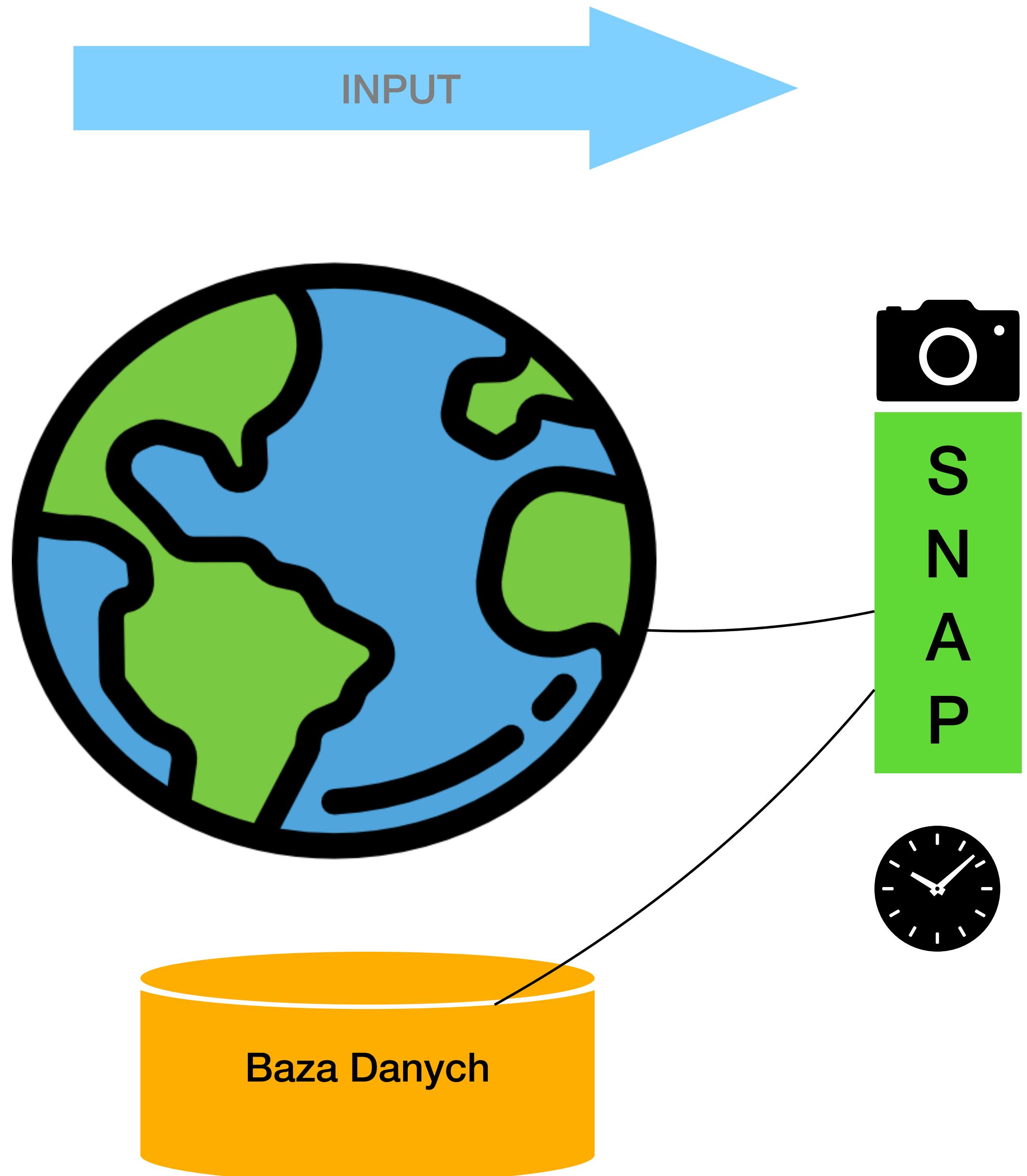
Nocleg 1	B		
Nocleg 2	B	O	
Nocleg 3	O		
Nocleg 4	B	O	A
Nocleg 5	A	O	B

Dobór
adaptacji

FABRYKA POLITYK

KOD

Fabryka polityk



POLITYKA
SPOSÓB DZIAŁANIA W
DANYM MOMENCIE
(Np. Aplikowalne adaptacje,
rabaty itp.)

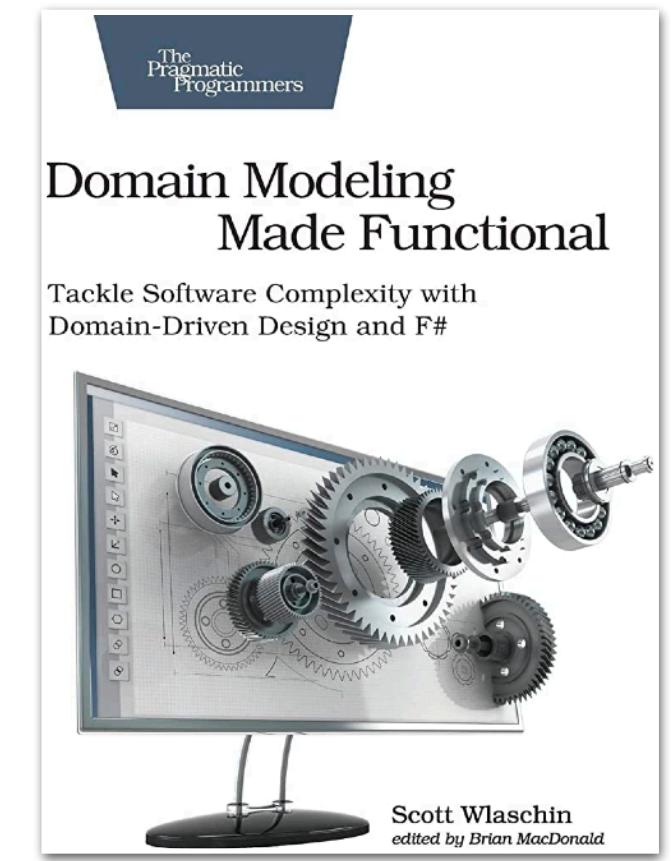
Co jeszcze wykorzystaliśmy z programowania funkcyjnego?

- Niemutowalność (Value Objects) - do panowania nad przetwarzanymi danymi
- Fold left (reduce) - do modelowania pewnej „algebry” wbudowanej w Domene
- DOMKNIĘCIE - pewne funkcje uzupełniamy niemutowalnymi danymi (snapshot świata)
- **Higher Order Functions** - funkcja zwracająca funkcję (politykę, strategię)
- do dostrojenia procesu
- **Pattern Matching** - do mapowania hierarchii klas w celu zachowania silniejszej enkapsulacji modułu

Co ta funkcja powinna zwracać?

Rodzaje błędów

- **BŁĘDY DOMENOWE** - brak dostępności, błędy walidacji etc.
- **BŁĘDY INFRASTRUKTURY** - błędy sieci, błędy uwierzytelniania etc. powodują potrzebę reakcji - np ponawiania.
- **PANIKI** - `NullPointerException`, `OutOfMemoryError` etc. nie da się z nich podnieść



Rzucać wyjątki?

Zwracać swój własny typ Result?

Zwracać Either?

Wygodnie jest używać hierarchii wyjątków!

Przykładowa hierarchia wyjątków

- **DomainException**
 - **NoAvailabilityException** - mapowany do 200 OK lub 409 Conflict
 - **RequestValidationException** - mapowany do 400 Bad Request
 - **NotEnoughRightsException** - mapowany do 403 Forbidden
 - ...
- **InfraException**
 - **NetworkTimeoutException** - powoduje ponowienie
- **PANICS** - każdy nieroznany Throwable

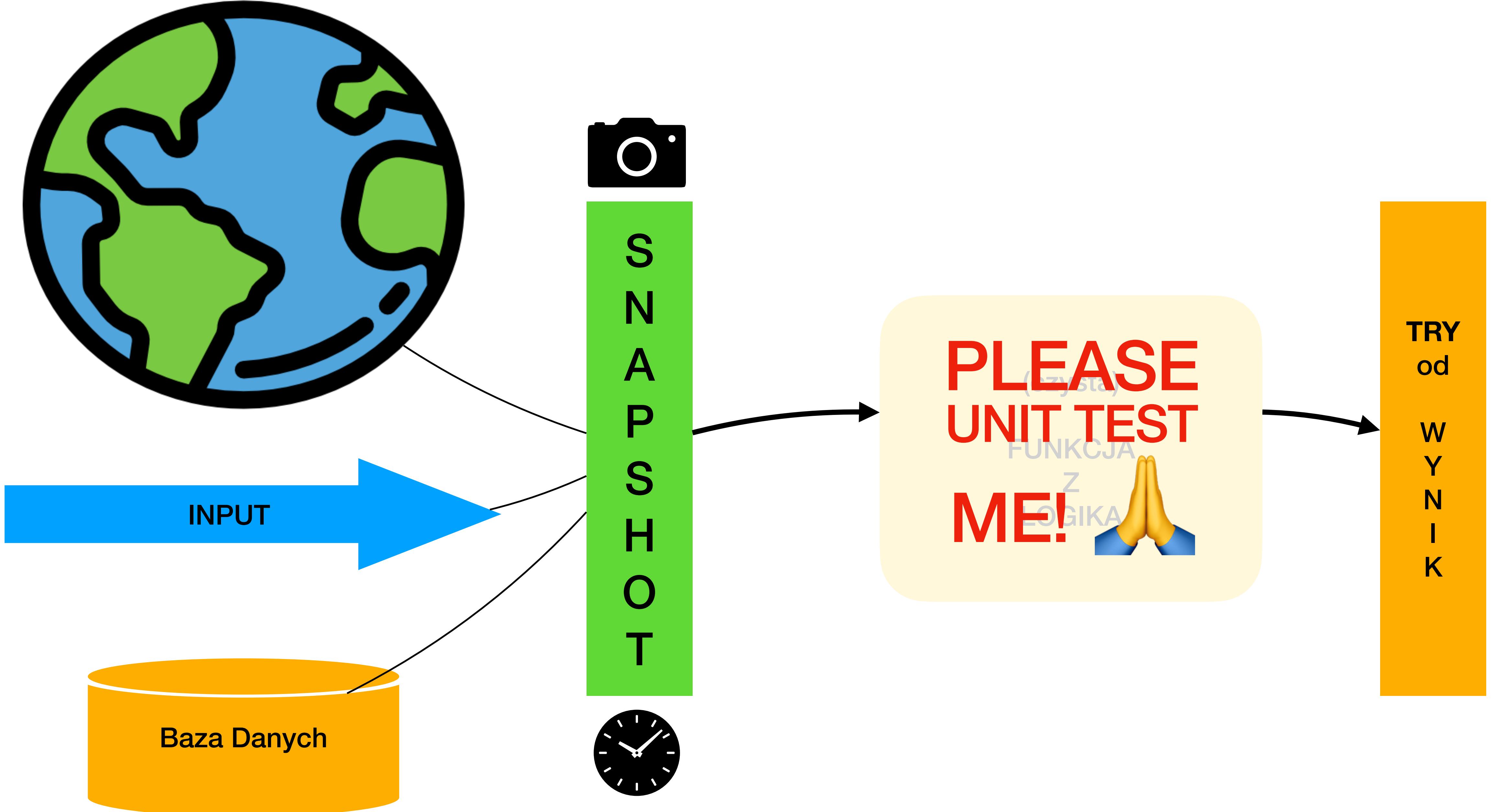
**Try<Typ dla Sukcesu>
- porażka to dowolny wyjątek**



vavr.io

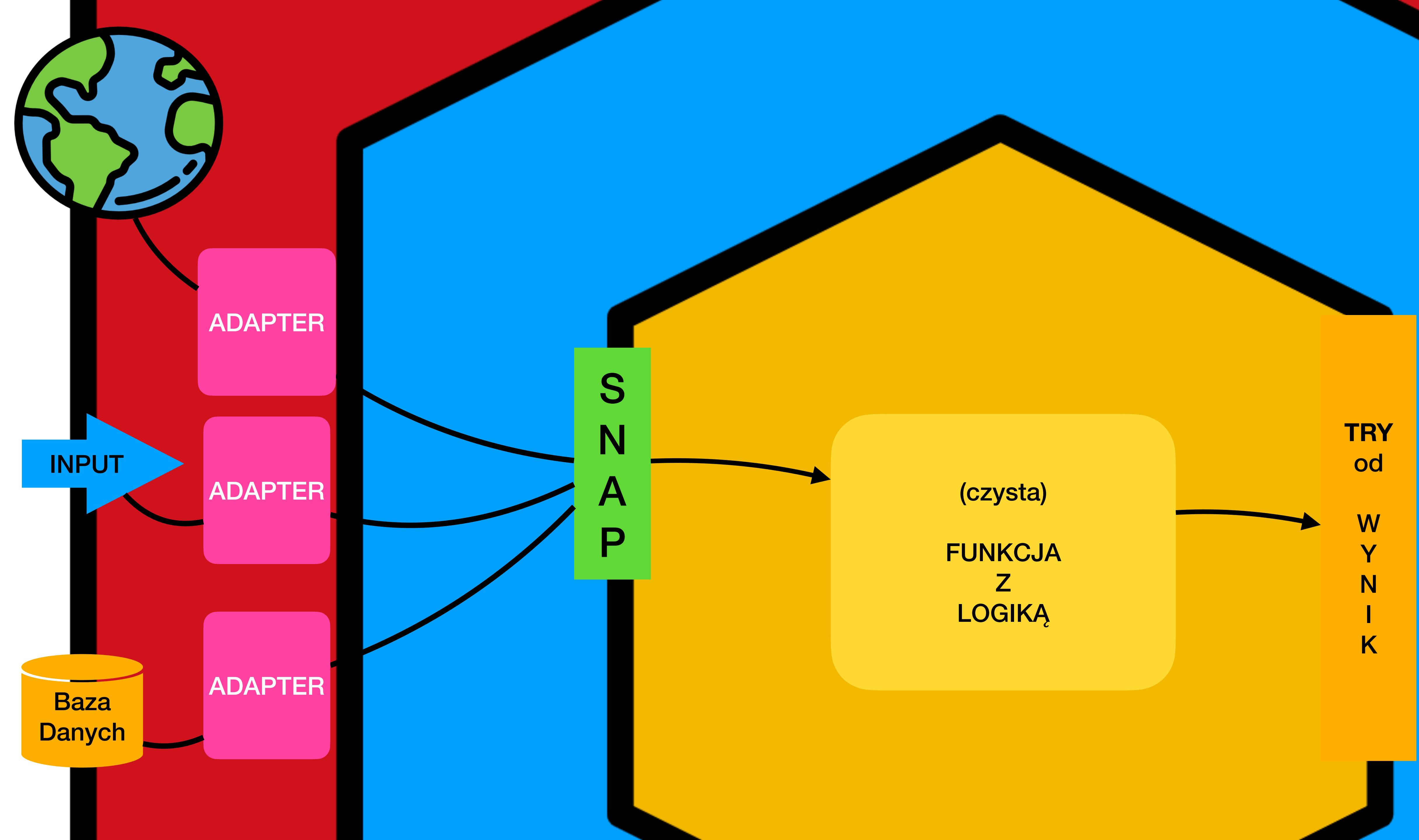
Mono<Typ dla sukcesu>
- porażka to bowiem wyjątkowy wyjątek

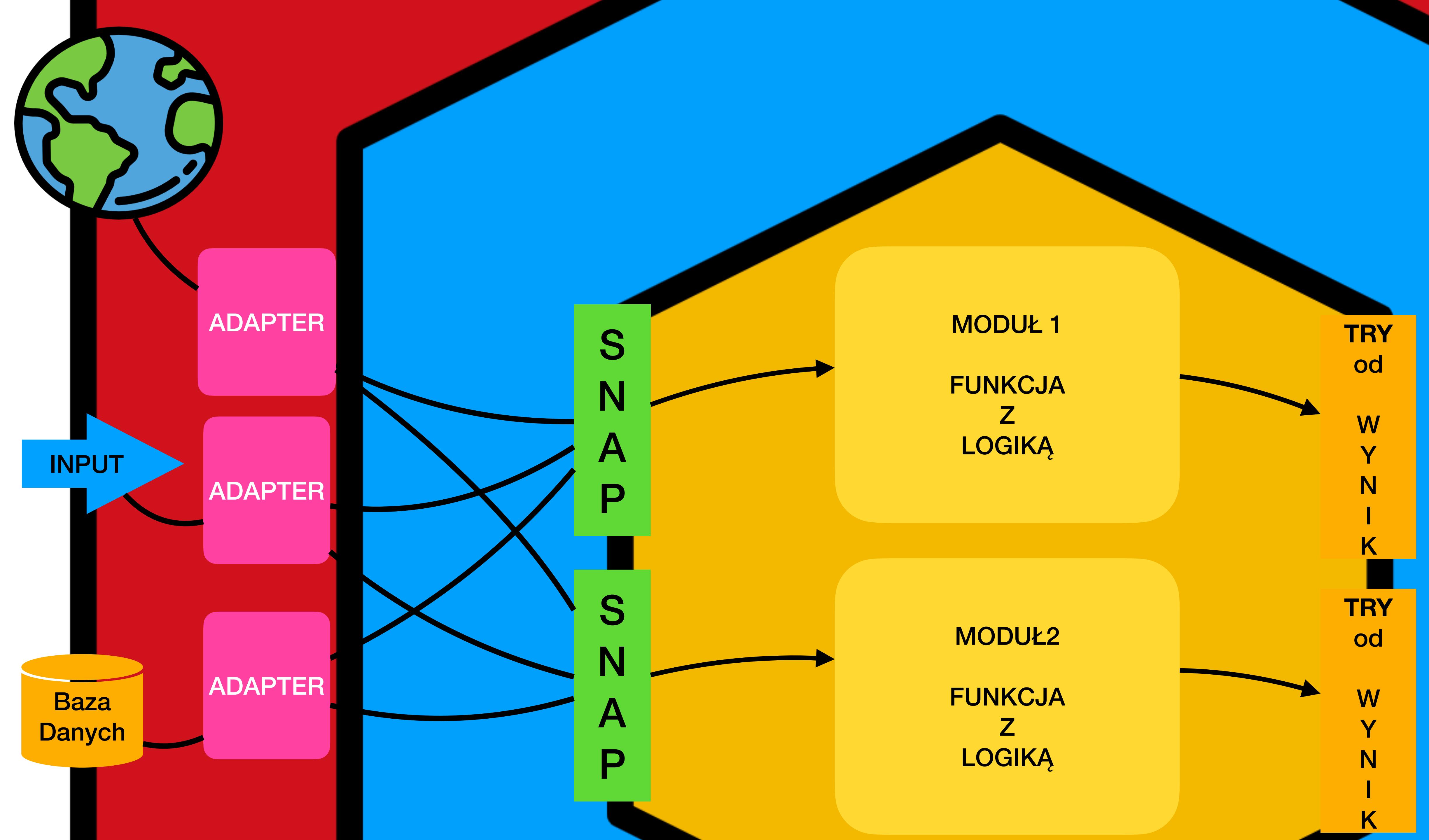
**KOMPATYBILNE Z PODEJŚCIEM
REAKTYWNYM**

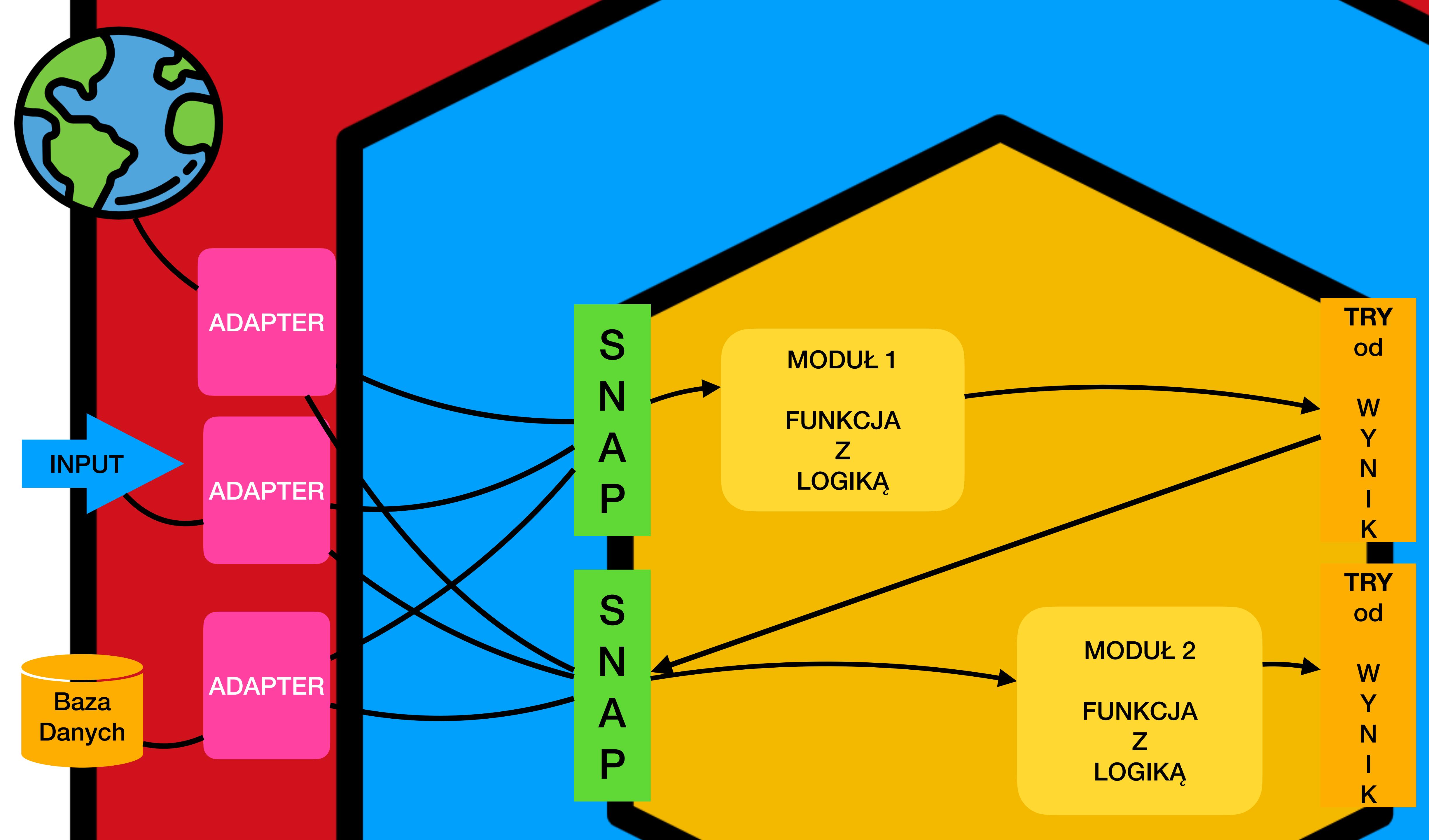


**Gdzie jest w tym wszystkim
framework?**

**Taka logika świetnie pasuje w
heksagonie!!**







Biznes często nie wie, że w niektórych fragmentach domeny myśli funkcyjnie...

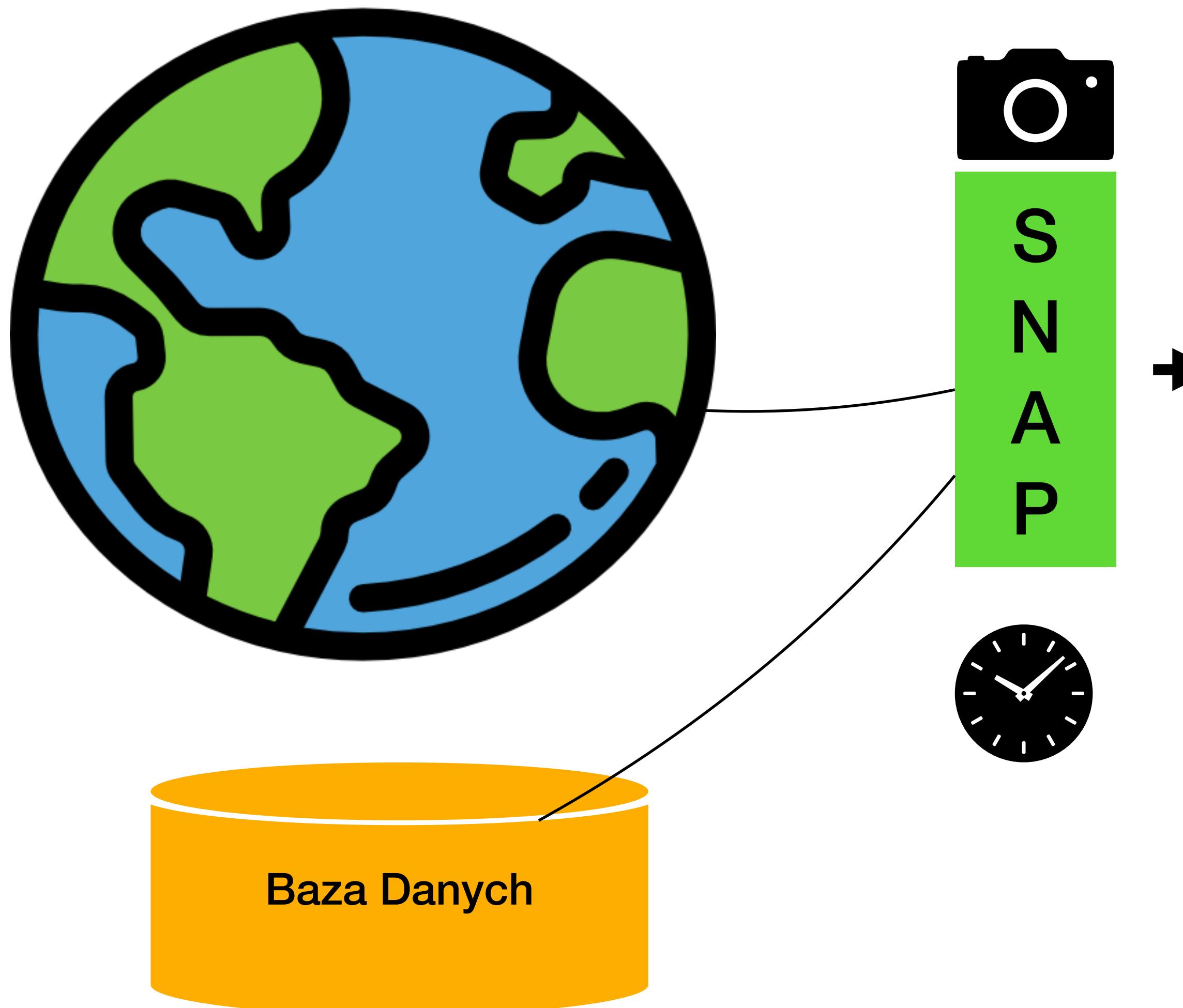
... ale wyczuwa, że potrzebuje rozwiązań na trudne „matematyczne” problemy...

... i jest wdzięczny gdy jest to dobrze zaimplementowane.

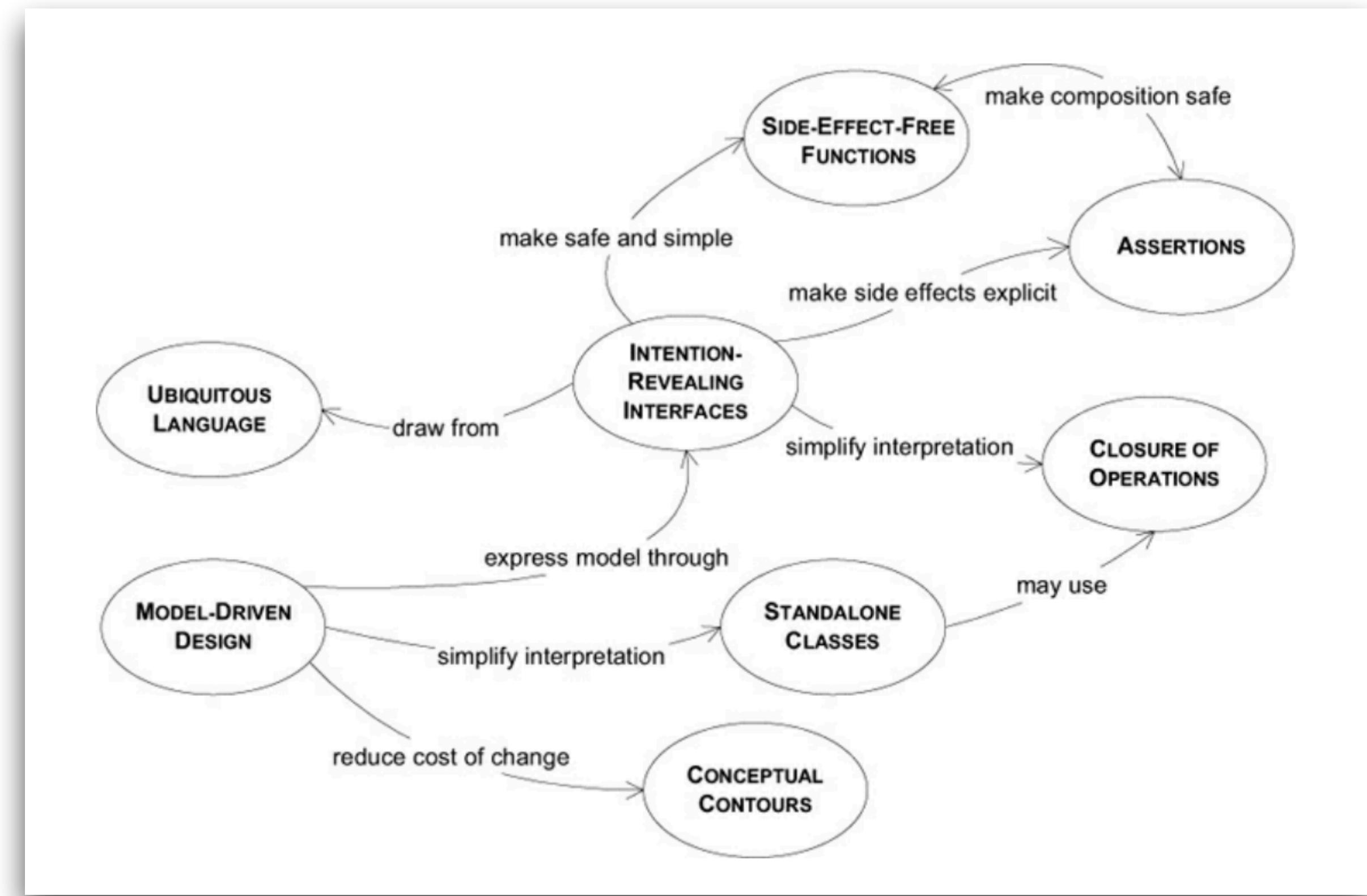
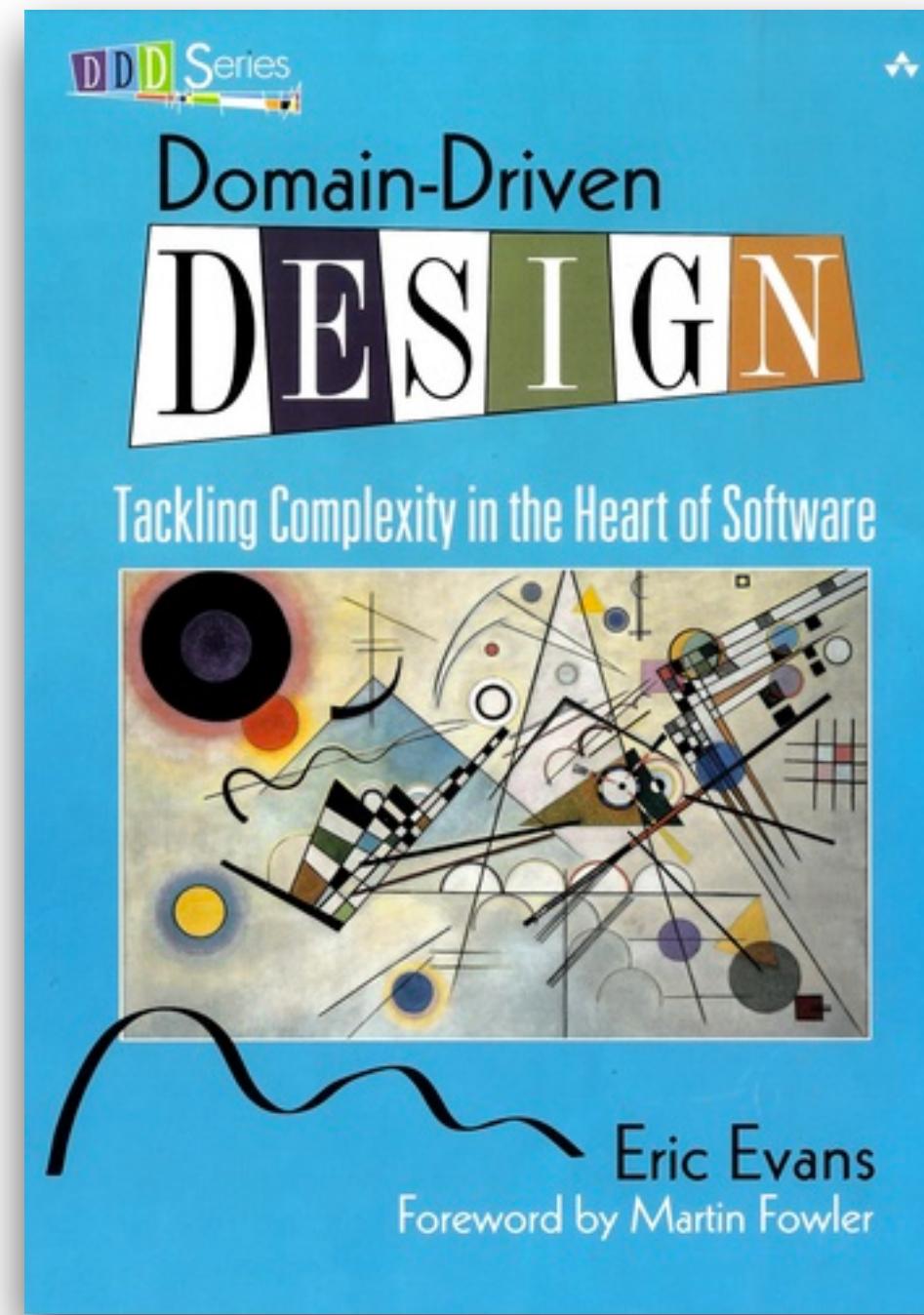
Co warto poznać z programowania funkcyjnego?

- **Niemutowalność (Value Objects, Snapshot stanu Świata)**
- **Higher Order Functions** - funkcja zwracająca funkcję (strategie)
- **Fold left** - jego uniwersalność i możliwości - rekursja w listach
- **DOMKNIĘCIE** - pewne funkcje uzupełniamy niemutowalnymi danymi (snapshot świata)
- **Monada Try<>** i na bardzo podstawowym poziomie pojęcie monady (aby zaprzyjaźnić się z nieszczęsnym flatMap)
- **Pattern matching** - mapowanie hierarchii klas

INPUT



Ważny patent!



Rozdział 10 - Supple design

Browse > Computer Science > Software Development

This course is part of the **Functional Programming in Scala Specialization**

Functional Programming Principles in Scala

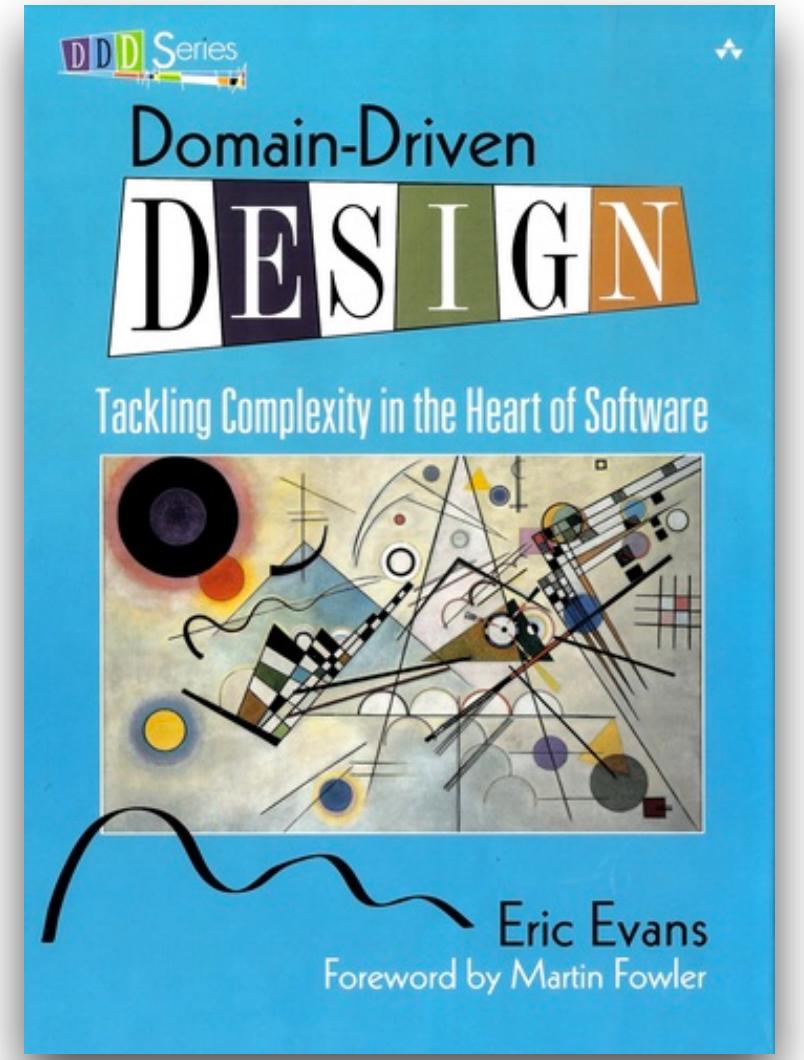
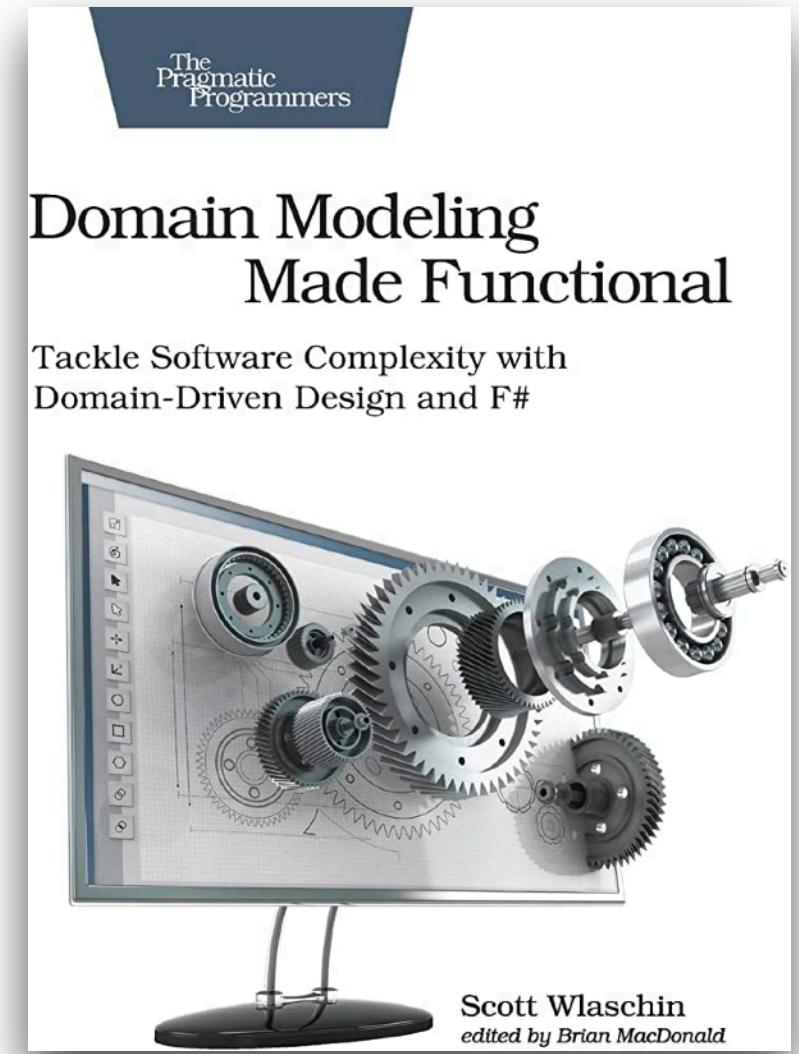
★★★★★ 4.8 7,738 ratings | 👍 92%

 Martin Odersky

Enroll for Free
Starts May 8

Financial aid available

187,827 already enrolled



Rozdział 10
Supple design

vavr

vavr.io

KOD na GitHub

Dzięki!



<https://github.com/szjanikowski/perfectrent>

Attributions

- Globe icons created by Freepik - Flaticon