

Wyświetlanie danych

Komponowanie Interfejsu Użytkownika, CQRS

Komponowanie Interfejsu Użytkownika

Komponowanie danych

Założenia

- Na potrzeby konkretnego widoku, składamy dane i wyświetlamy je użytkownikowi
- Złożenie danych może dobywać się na różnych warstwach
- Implementacja najprostsza, najtańsza
- Wydajność - szybko !!!

Kategorie

1. c R ud - jednego typu danych, pochodzących z jednego źródła
2. Komponowanie widoku z wielu źródeł
3. Wyszukiwanie, stronicowanie danych z wielu źródeł

Demo - która kategoria ?

Controller - SendNotificationsByUserController

Jako: Administrator

Chciałbym: wyświetlić, na moim panelu, użytkowników wraz z ilością maili do nich wysłanych

W celu: Poszukiwanie anomalii

Jak można komponować dane (Kategoria 2,3)

Kategoria 2 - Komponowanie widoku z różnych źródeł

- Warstwa UI (client side)
- Warstwa UI (server side)
- Warstwa aplikacyjna - dostawiamy osobny moduł który agreguje dane w locie
- Baza danych - replikacja
 - źródło prawdy
- Baza danych (select ... join)

Kategoria 3 - Wyszukiwanie, stronicowanie danych z różnych źródeł


- Warstwa aplikacyjna - CQRS (zasilany zdarzeniami)

Dyskusja - która to kategoria ?

by [Martin Fowler](#) (Author)

★★★★☆ 126 customer reviews


[Look inside](#)



ISBN-13: 978-0321127426
ISBN-10: 0321127420
[Why is ISBN important?](#)

Have one to sell? [Sell on Amazon](#)

[Add to List](#)

Kindle 

\$47.18

Hardcover

\$32.90 - \$40.38

Other Sellers

See all 3 versions

☐ Buy used \$32.90

☒ **Buy new** **\$40.38**

In Stock.

Ships from and sold by Amazon.com.

List Price: \$69.99
Save: \$29.61 (42%)
26 New from \$30.39

This item ships to **Poland**. Want it **Tuesday, Sept. 17**? Order within **3 hrs 48 mins** and choose **AmazonGlobal Priority Shipping** at checkout. [Learn more](#)

[Deliver to Poland](#)

Qty: 1

[Add to Cart](#)

[Buy Now](#)

More Buying Choices

26 New from \$30.39 | 34 Used from \$32.90 | 1 Collectible from \$26.29

61 used & new from \$26.29

[See All Buying Options](#)

Ćwiczenie:

Administrator przegląda użytkowników na panelu administratorskim. Zaimplementuj możliwość stronicowania danych dla każdego użytkownika.

Controller - AlertNotificationsWithUserDataController Test -

Panel administracyjny:

Dla administratora udostępniony jest specjalny dashboard prezentujący

- login użytkownika
- ilość założonych alarmów

CQRS

CAP Theorem

- Consistency
- Availability
- Partition tolerance



ACID vs BASE



- Atomic
- Consistent
- Isolated
- Durable



- Basically Available
- Soft state
- Eventual consistency

Eventual consistency



Copyright 2023 © Szymon Szyllabel

CQRS - Ogólne założenia

Stos Command

- Servisy stają się command-handlerami (tylko logika aplikacji)
- logika biznesowa delegowana do obiektów domenowych
- Encje tylko do zapisu, zorientowane na zachowanie – enkapsulacja stanu

Stos Query

- Cienka warstwa nad bazą danych – nie musi być złożona
- Zwraca DTO – encje zwykle nie nadają się do prezentacji (grids)

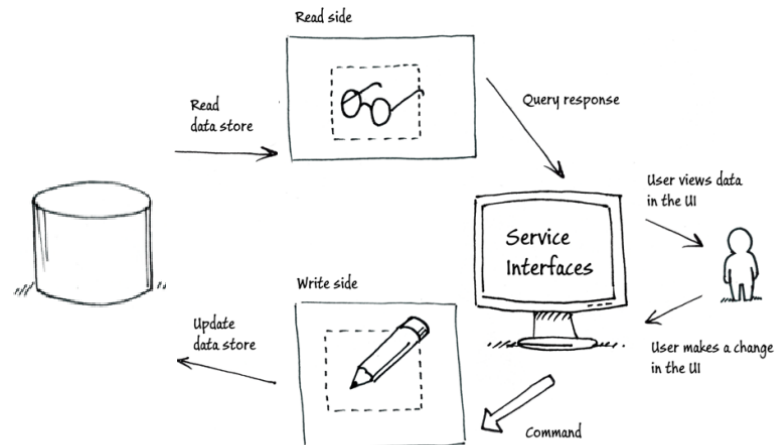
Level 0

Założenia (Read side)

- Funkcje / Widoki
- Widoki zmaterializowane

Założenia (Write side)

- Dane zapisywane w formie relacyjnej



Read side – przykładowa implementacja

```
public class SearchDocumentsQuery
{
    public int Type { get; }
}
```

```
public class DocumentQueries : IDocumentQueries
{
    public List<DocumentDTO> Search(SearchDocumentsQuery query)
    {
        using (SqlConnection connection = new SqlConnection(_connectionString))
        {
            return connection.Query<int>(
                "select Name,Status from Documents where Type = @type",new { type = query.Type});
        }
    }
}
```

Write side – przykładowa implementacja

Demo

- Jak wygląda podział na 2 stosy
- Dapper
- Solucja gdzie tu jest hexagon
- Memento
- Autofac (automatyczne rejestracje)
- Jak skonstruowana jest perspektywa
- Jak wyglądają testy
- Gdzie będziemy testować wyjątki od happy scenario
- Jak obsłużony jest problem czasu
- Jak wygląda bootstrap
- Jak wygląda automatyczna migracja

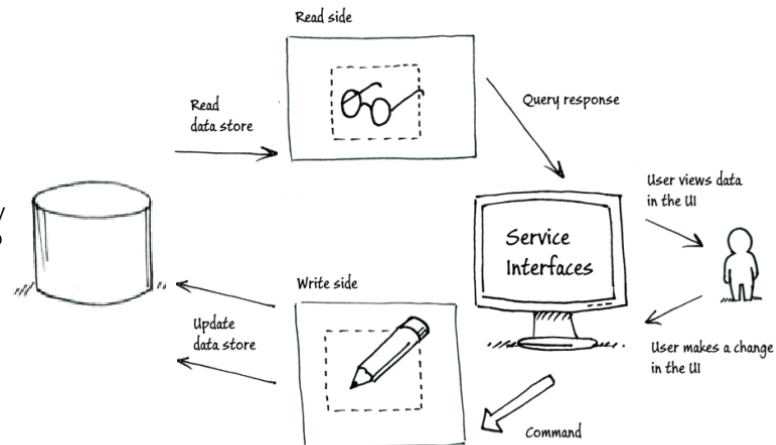
Level 1

Założenia (Read side)

- Odczyty z osobnej tabeli

Założenia (Write side)

- Dane zapisywane w formie relacyjnej
- Synchroniczny, transakcyjny zapis do bazy relacyjnej i do read side



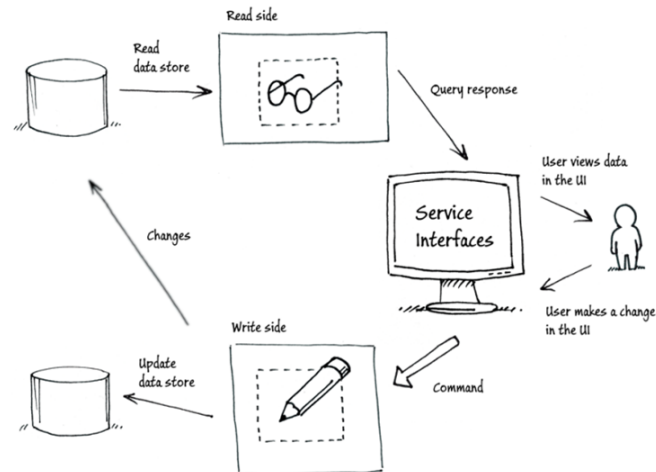
Level 2

Założenia (Read side)

- Odczyty z osobnej tabeli
- zasilana asynchronicznie zdarzeniami

Założenia (Write side)

- Dane zapisywane w formie relacyjnej
- Przy zmianach publikowane zdarzenia



Ćwiczenie: Wyszukiwanie użytkowników którzy mają zadaną ilość alarmów

Administrator często wyszukuje użytkowników którzy mają zarejestrowanych wiele alarmów. Zaimplementuj możliwość filtrowania po ilości zarejestrowanych alarmów.

Controller - SearchableDashboardController Test - SearchableDashboardTest

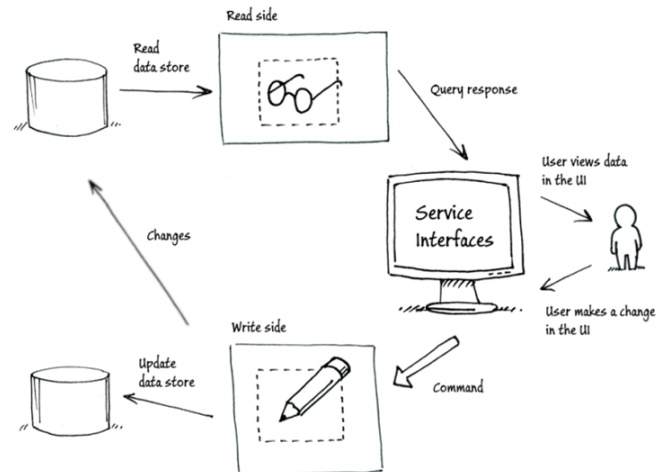
Level 3 - Event Sourcing

Założenia (Read side)

- Odczyty z osobnej tabeli
- zasilana asynchronicznie zdarzeniami

Założenia (Write side)

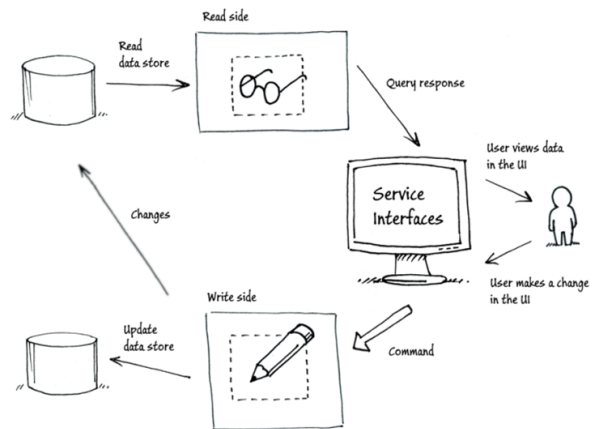
- Dane zapisywane w formie zdarzeń (EventSourcing)
- Przy zmianach publikowane zdarzenia



Level 3 - Event Sourcing

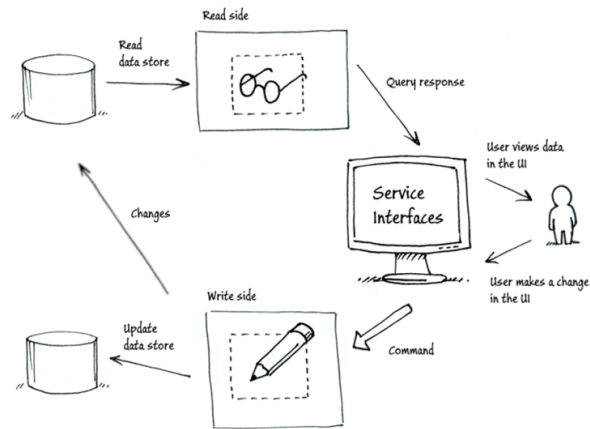
- Poisson Pill Architecture
- Snapshoting
- Zasilanie zdarzeniami
- Mapping'i - ORMy
- Wiele widoków (analizy, utrzymanie)

CQRS



- CAP
- BASE
- Scale Up/Down

CQRS vs Outsourcing



Które elementy są naszym „Know-How”?

Które elementy są łatwe w implementacji?

Jakie są konsekwencje słabej jakości kodu?

CQRS podsumowanie

- Paradygmat CQS
- Twierdzenie CAP
- Scale Out/Up
- ACID vs BASE
- Typowa architektura vs CQRS
- CQRS
 - Level 0 - Jedna reprezentacja (model synchroniczny)
 - Level 1 - Osobna tabela (model synchroniczny)
 - Level 2 - Read model zasilany zdarzeniami, Write model - relacyjny
 - Level 3 - Read model zasilany zdarzeniami, Write model - EventStore
- Events Sourcing
 - Poison pill architecture

Źródła (CQRS)

Artykuły

- CQRS Journey (Microsoft) <https://msdn.microsoft.com/en-us/library/jj554200.aspx>
- Udi Dahan - Clarified CQRS - <http://udidahan.com/2009/12/09/clarified-cqrs/>
- Microsoft - Tackle Business Complexity in a Microservice with DDD and CQRS Patterns - <https://docs.microsoft.com/en-us/dotnet/standard/microservices-architecture/microservice-ddd-cqrs-patterns/eshoponcontainers-cqrs-ddd-microservice>

Kod

- EventFlow CQRS framework - <https://github.com/rasmus/EventFlow/tree/develop/Source>
- Greg Young - Simple CQRS - <https://github.com/gregoryyoung/m-r>

Narzędzia

- EventStore - <https://eventstore.org/>
- Marten - <https://jasperfx.github.io/marten/>
- StreamStone - Azure Table Storage

Komponowanie interfejsu użytkownika (podsumowanie)

Kategorie

1. c R ud - jednego typu danych, pochodzących z jednego źródła
2. Komponowanie widoku z wielu źródeł
3. Wyszukiwanie, stronicowanie danych z wielu źródeł - CQRS

Antypattern

- sql join

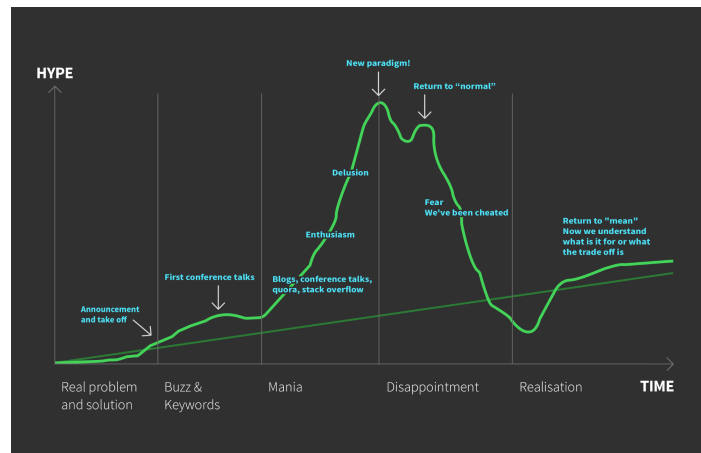
Szkolenie się skończyło.

Co dla ciebie było szczególnie interesujące?

Co chciałbyś "jutro" spróbować

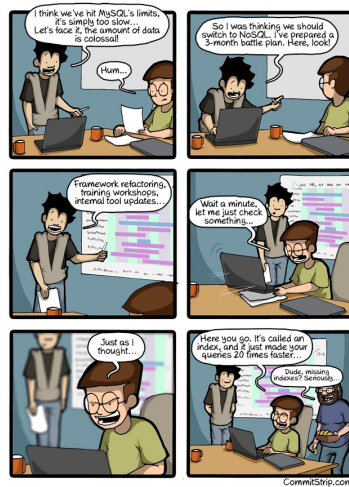
Jak dalej mam rozwijać swoje kompetencje ?

HDD



Copyright 2023 © Szymon Szyllhabel

Lepsze (nowe) jest wrogiem dobrego



Model kompetencji braci Dreyfus

Expert	Głównie intuicja, wszystko zależy od kontekstu
Proficient	Szerszy kontekst, metafory, antywzorce. Pojawia się instynkt. Wiem, że nic nie wiem.
Competent	Nastawienie na cel, samodzielny dobór kroków. Pierwsze wzorce. Wyżej == podjęcie wysiłu poznawczego
Advanced beginner	Wciąż nastawienie na zadania, jednak umiejętność składania kroków. Tendencja do zawyżania samooceny.
Novice	Nastawienie na zadania, potrzeba określenia prostych kroków

Bloggers

- [Uncle Bob Martin](#)
- [Greg Young](#)
- [Udi Dahan](#)
- [Nick Tune](#)
- [Martin Fowler](#)
- [Scott Hanselman](#)
- [Sandi Metz](#)
- [Venkat Subramaniam](#)
- [Michael Feathers](#)
- [Rob Eisenberg](#)
- [Neal Ford](#)
- [Kent Beck]
- [Alistair Cockburn]
- [Jimmy Bogard]
- [Dan North]
- [Jeremy Clark]
- [Scott Alen]
- [Simon Brown]
- [Gojko Adzic](#)
- [Troy Hunt](#)

Conferences

International

- NDC ([Oslo](#)/[London](#)/[Sydney](#)) - [videos](#)
- [goto](#);
- [Qredev](#)
- [Build Stuff](#)

Polish

- [4 developers](#)
- [Boiling Frogs](#)
- [Developer Days](#)