DOMAIN DRIVEN DESIGN

Dla opornych

ILE PIENIĘDZY NA KONCIE?

- dzwoni Staś do banku i pyta ile ma kasy na koncie
- "80 złotych" odpowiada pani w banku

- Hola, gdzie jest moje 300 złotych?

- Oto pańskie wydatki z ostatnich 2 dni
  - 200 na payu
  - 300 wyciągnięte z bankomatu
  - 100 na amazonie
- a tak, zapomniałem o bankomacie ...

CZY SALDO KONTA W SYSTEMIE BANKOWYM JEST KOLUMNĄ W TABELI?

MOŻLIWE, ALE NAJPEWNIEJ NIE JEST TO "ŹRÓDŁEM PRAWDY"

PRZYKŁAD IMPLEMENTACJI

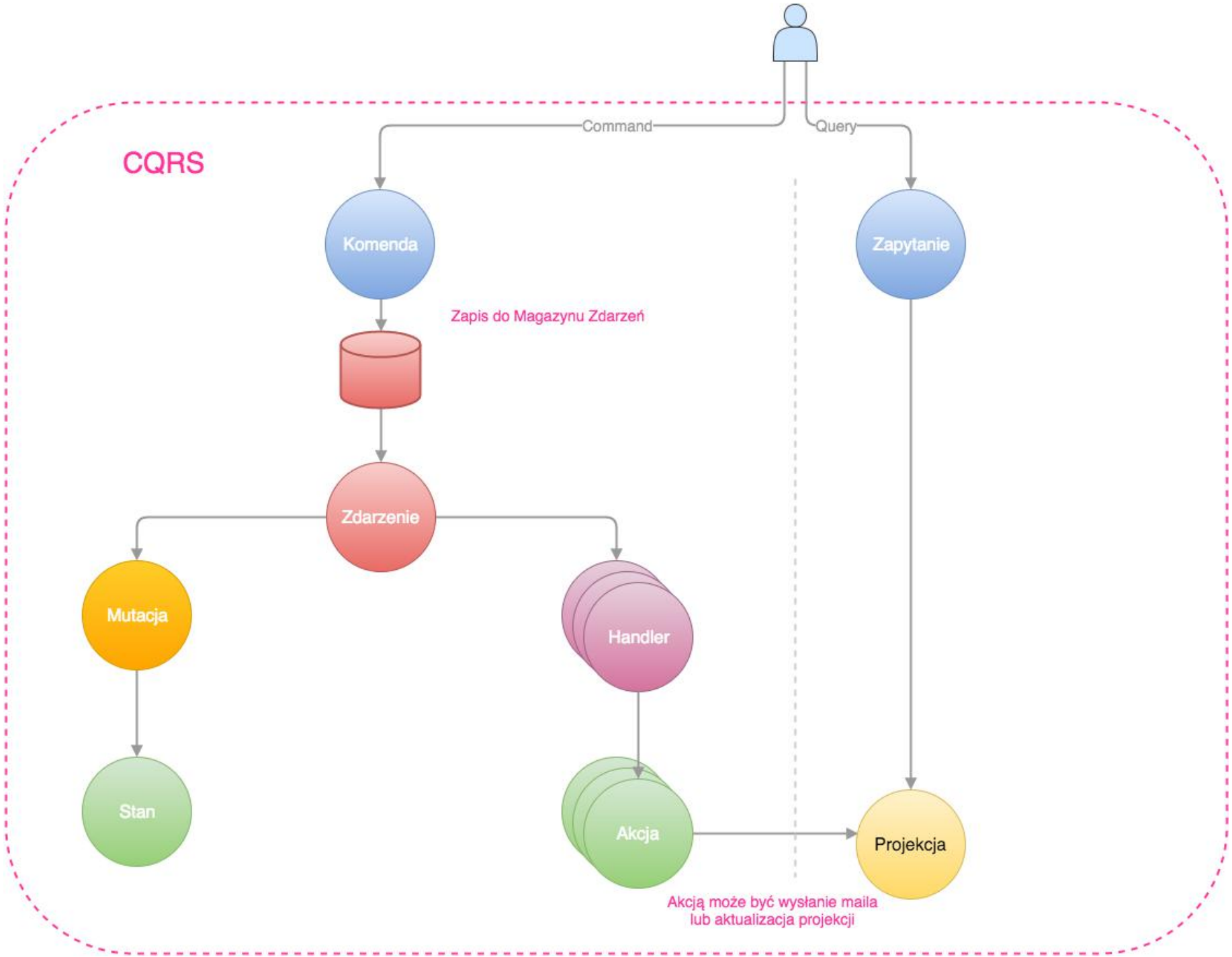
```
def wire_money(request):
    data = validate_wire_money(request)
    account = get_account(request.user)
    send_money(account.id, data['target'], data['value'])
    return Response(status=200)
```

```
def send_money(account_id, target, value):
    sql = '''
    UPDATE account
    SET current_val = (account.current_val - %s)
    WHERE account.id = %s
    '''
    perform_query(sql, value, target)
    save_history_entry(
        name='wire',
        account_id=account_id,
        target=target,
        value=value,
    )
```

PROBLEMY?

- transakcja
- błąd w zapisie wpisu historii

Mutowanie danych


CQRS
Command
Query
Komenda
Zapytanie
Zapis do Magazynu Zdarzeń
Zdarzenie
Mutacja
Handler
Stan
Akcja
Projekcja
Akcją może być wysłanie maila
lub aktualizacja projekcji

COMMAND QUERY RESPONSIBILITY SEPARATION

DDD OD POCZĄTKU

RUPY 2013

RUBY LOVES DDD

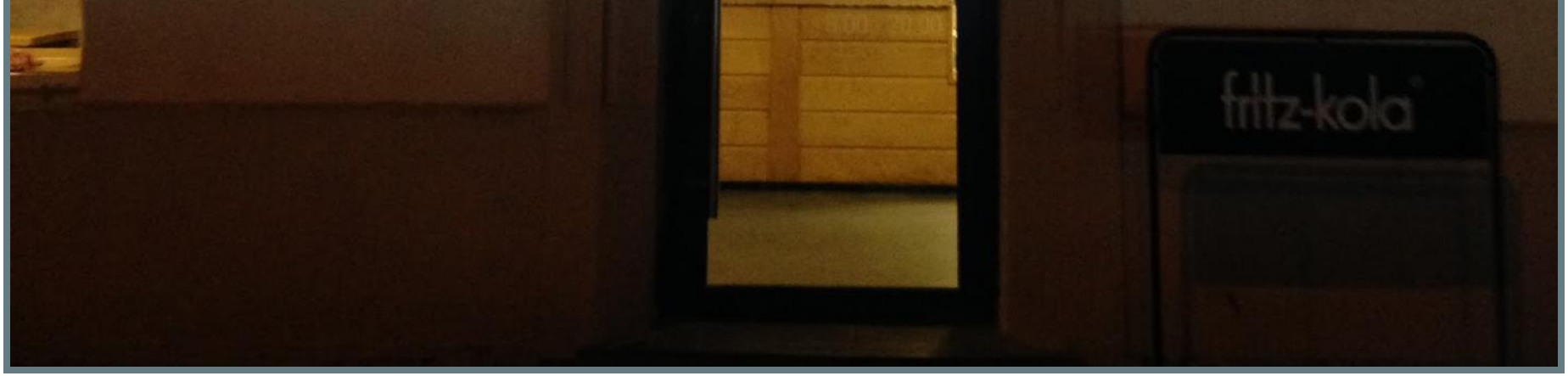
https://www.youtube.com/watch?v=Vn34CdkLa3g

Moja pierwsza próba implementacji skończyła się totalną porażką

Zwalam na terminy i comfort zone

Dużo dużo później, właściwie całkiem niedawno


moko
tuff
moko
tuff
moko
22.00


fritz-kola

- Python
- Django
- DRF
- REST

PYTANIE

- dlaczego serializator wie jak zapisać obiekt?

PYTANIE

- dlaczego serializator wie jak zapisać obiekt?
- dlaczego logika endpointu wie jak zapisać dane?

PYTANIE

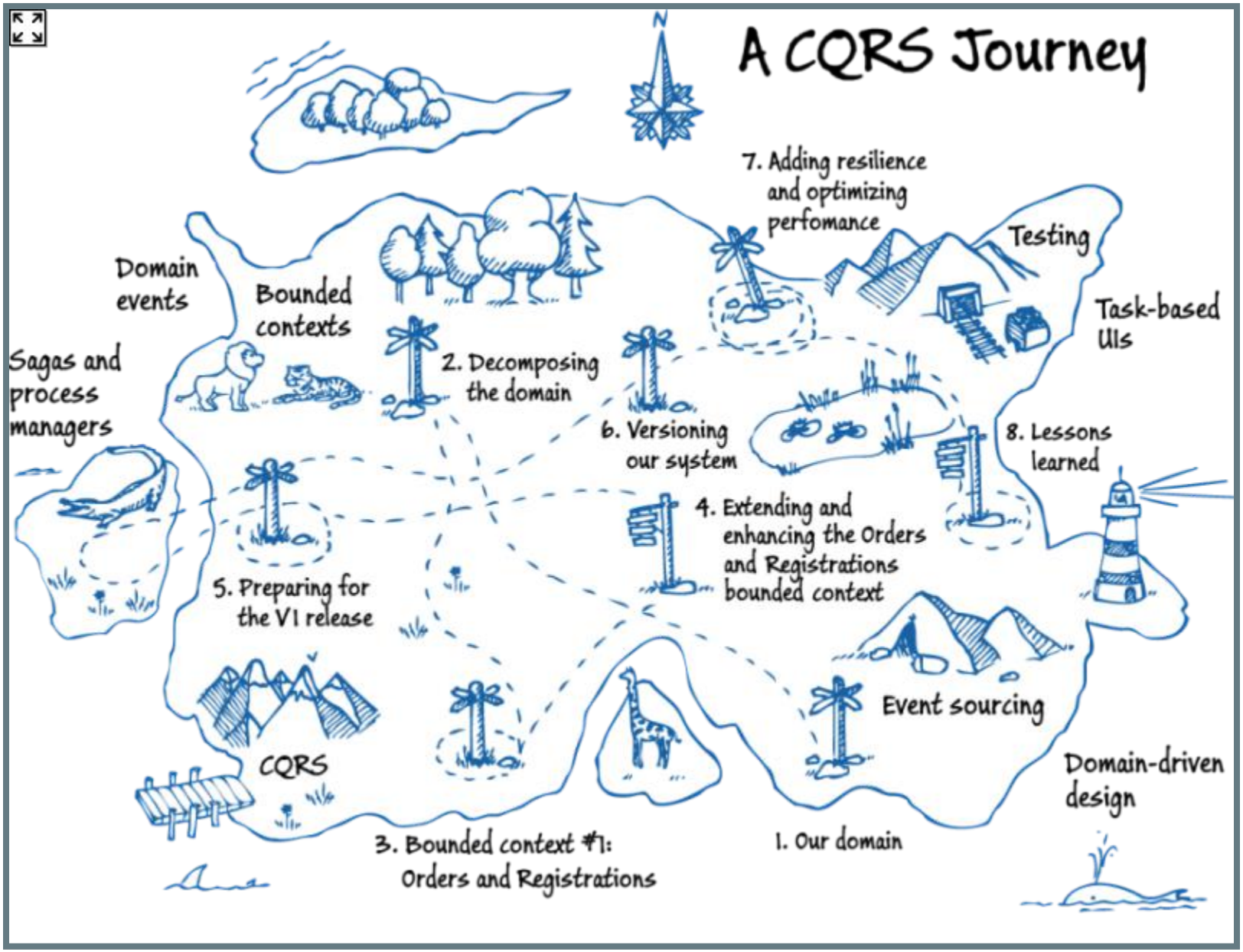
- dlaczego serializator wie jak zapisać obiekt?
- dlaczego logika endpointu wie jak zapisać dane?
- Model.save() vs Manager.create()/.update() ?

- Java
- Data Transfer Object

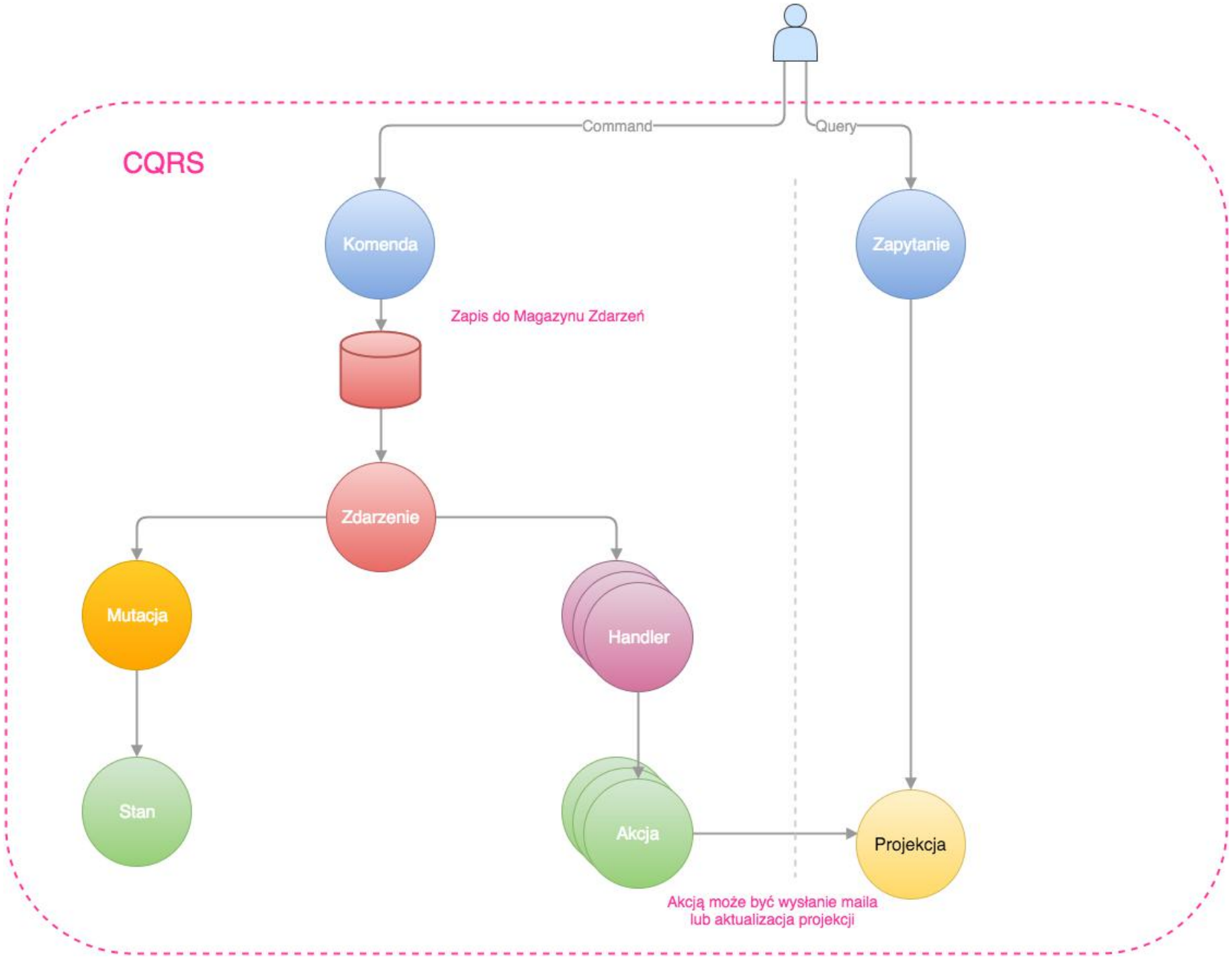
WNIOSEK

- Zbyt duża elastyczność
- Brak jasnego podziału odpowiedzialności
- Zmiana stanu danych jest (zbyt) łatwa

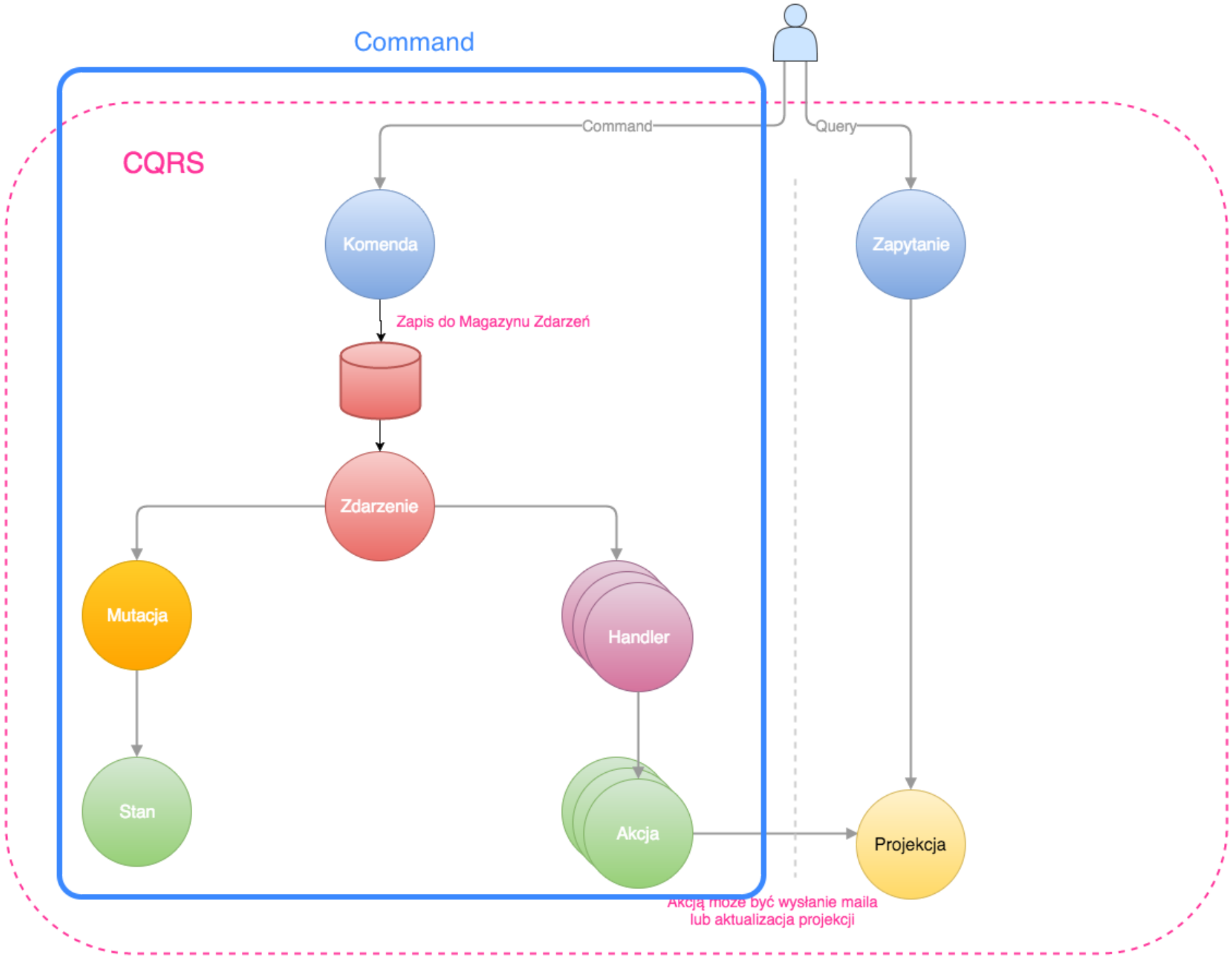

APPLAUSE®

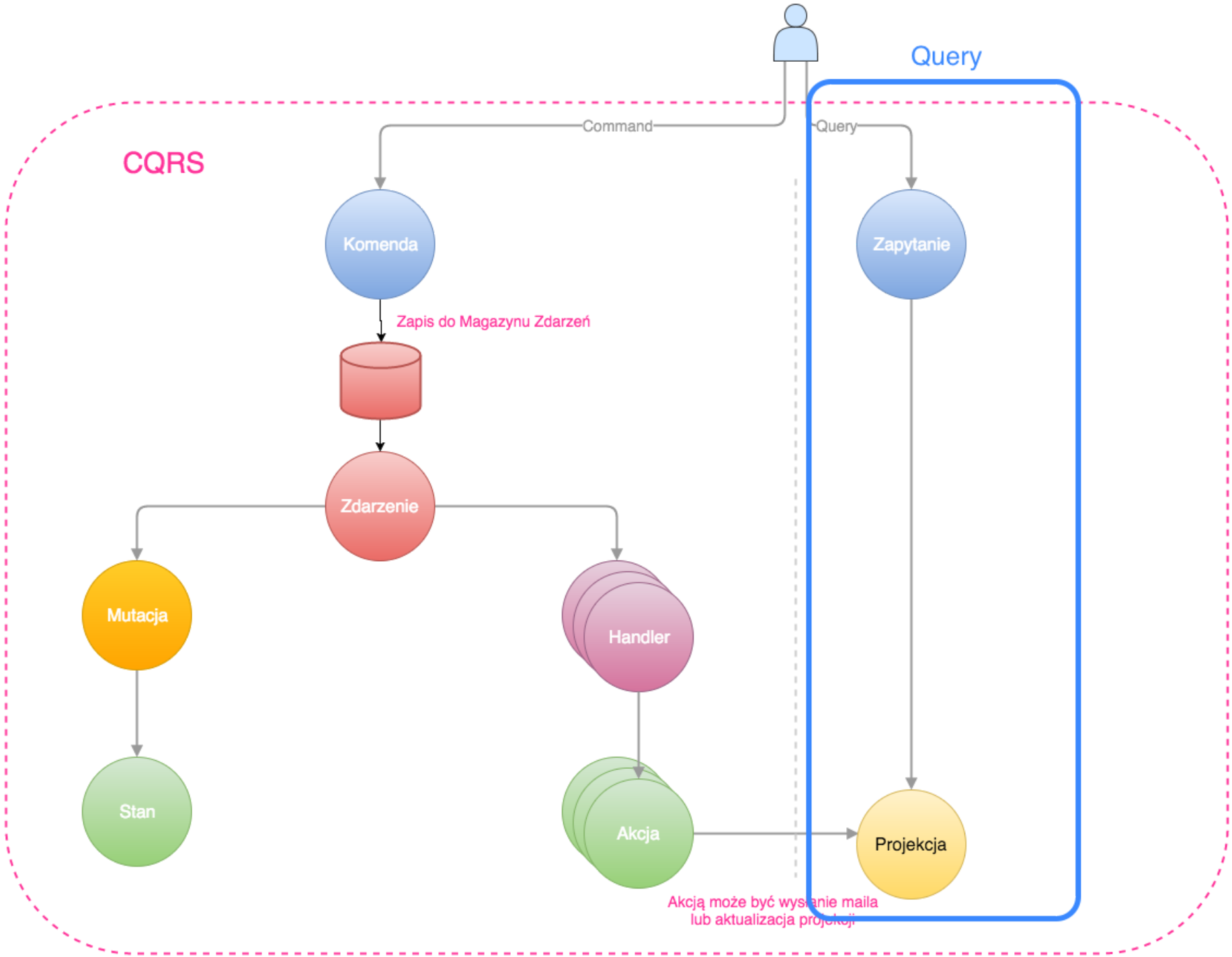

A CQRS Journey
N
7. Adding resilience and optimizing perfomance
Testing
Task-based UIs
Domain events
Bounded contexts
2. Decomposing the domain
Sagas and process managers
6. Versioning our system
8. Lessons learned
4. Extending and enhancing the Orders and Registrations bounded context
5. Preparing for the V1 release
Event sourcing
CQRS
Domain-driven design
3. Bounded context #1: Orders and Registrations
1. Our domain

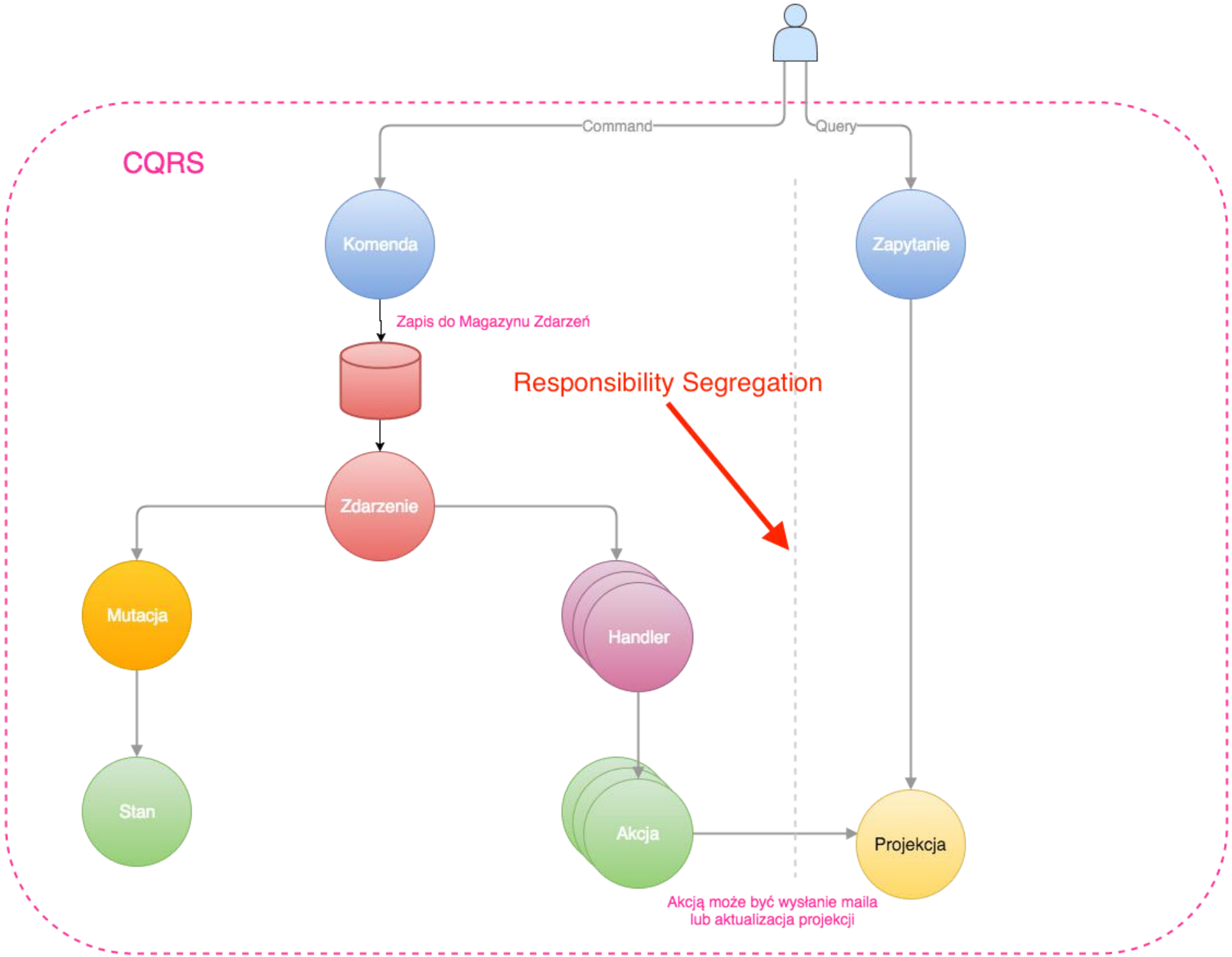
- CQRS
- ES
- DDD
- Microsoft


CQRS
Command
Query
Komenda
Zapytanie
Zapis do Magazynu Zdarzeń
Zdarzenie
Mutacja
Handler
Stan
Akcja
Projekcja
Akcją może być wysłanie maila
lub aktualizacja projekcji

COMMAND QUERY RESPONSIBILITY SEGREGATION


Command
CQRS
Command
Query
Komenda
Zapytanie
Zapis do Magazynu Zdarzeń
Zdarzenie
Mutacja
Handler
Stan
Akcja
Projekcja
Akcją może być wysłanie maila
lub aktualizacja projekcji


Query
Command
Query
CQRS
Komenda
Zapytanie
Zapis do Magazynu Zdarzeń
Zdarzenie
Mutacja
Handler
Stan
Akcja
Projekcja
Akcją może być wysłanie maila
lub aktualizacja projekcji


CQRS
Command
Query
Komenda
Zapytanie
Zapis do Magazynu Zdarzeń
Responsibility Segregation
Zdarzenie
Mutacja
Handler
Stan
Akcja
Projekcja
Akcją może być wysłanie maila
lub aktualizacja projekcji

PRZYKŁAD APLIKACJI TODO

https://github.com/lukaszb/cq

```
from todos.cqrs import app

app.add(label='Prepare PyWaw presentation')
app.add(label='Find funny pictures')

app.query()

# fa56bc | Prepare PyWaw presentation
# 50dd61 | Find funny pictures

app.finish('fa56bc')
```

```
from .aggregates import Todo
from cq.app import BaseApp
from cq.contrib.sqlalchemy.storage import SqlAlchemyStorage


class TodoApp(BaseApp):
    storage_class = SqlAlchemyStorage
    storage_kwargs = {'db_uri': 'sqlite:///:memory:'}

    def __init__(self):
        super().__init__()
        self.repo = self.get_repo_for_aggregate(Todo)
```

```
    def add(self, label):
        uuid = self.genuuid()  # can be provided by the client as well
        return self.repo.store('Todo.Added', uuid, data={'label': label})

    def finish(self, todo_id):
        self.repo.get_aggregate(todo_id)
        self.repo.store('Todo.Finished', todo_id)
```

```
from cq import aggregates


class Todo(aggregates.Aggregate):
    __slots__ = ('label', 'done')


@aggregates.register_mutator(Todo, 'Todo.Added')
def mutate_added(todo, event, data):
    todo.label = data['label']
    todo.done = False


@aggregates.register_mutator(Todo, 'Todo.Finished')
def mutate_finished(todo, event, data):
    todo.done = True
```

```
from todos.projections import Todo

    # ...
    def query(self):
        session = self.storage.get_session()
        return session.query(Todo)
```

HANDLERS.PY

```
from todos.projections import Todo
from cq.handlers import register_handler


@register_handler('Todo.Added')
def todo_added(event):
    todo = Todo(
        id=event.aggregate_id,
        label=event.data['label'],
    )
    session = get_session()
    session.add(todo)
    session.commit()

@register_handler('Todo.Finished')
def todo_finished(event):
    session = get_session()
    session.query(Todo).filter(Todo.id == event.aggregate_id).update({'d
```

KOMUNIKACJA MIĘDZY APLIKACJAMI / USŁUGAMI

- komendy i zapytania jako główny interfejs
- aplikacja może nasłuchiwać na zdarzenia z innych aplikacji

ZALETY

- Podział obowiązków
- Interfejs całej aplikacji/usługi
- Komunikacja między usługami

WADY

- Złożoność systemu
- Duplikacja części logiki

MATERIAŁY

- Greg Young, "CQRS and ES": https://www.youtube.com/watch?v=JHGkaShoyNs
- Greg Young, "CQRS Documents": https://cqrs.files.wordpress.com/2010/11/cqrs_documents.pdf
- Martin Fowler: https://martinfowler.com/eaaDev/EventSourcing.html
- V. Vernon, "DDD dla architektów oprogramowania"
- Microsoft CQRS Journey: https://msdn.microsoft.com/en-us/library/jj554200.aspx
- Dariusz Pawlukiewicz: https://bulldogjob.pl/articles/122-cqrs-i-event-sourcing-czyli-latwa-droga-do-skalowalnosci-naszych-systemow

NARZĘDZIA

- Kafka (https://kafka.apache.org/)
- Maxwell (https://github.com/zendesk/maxwell)

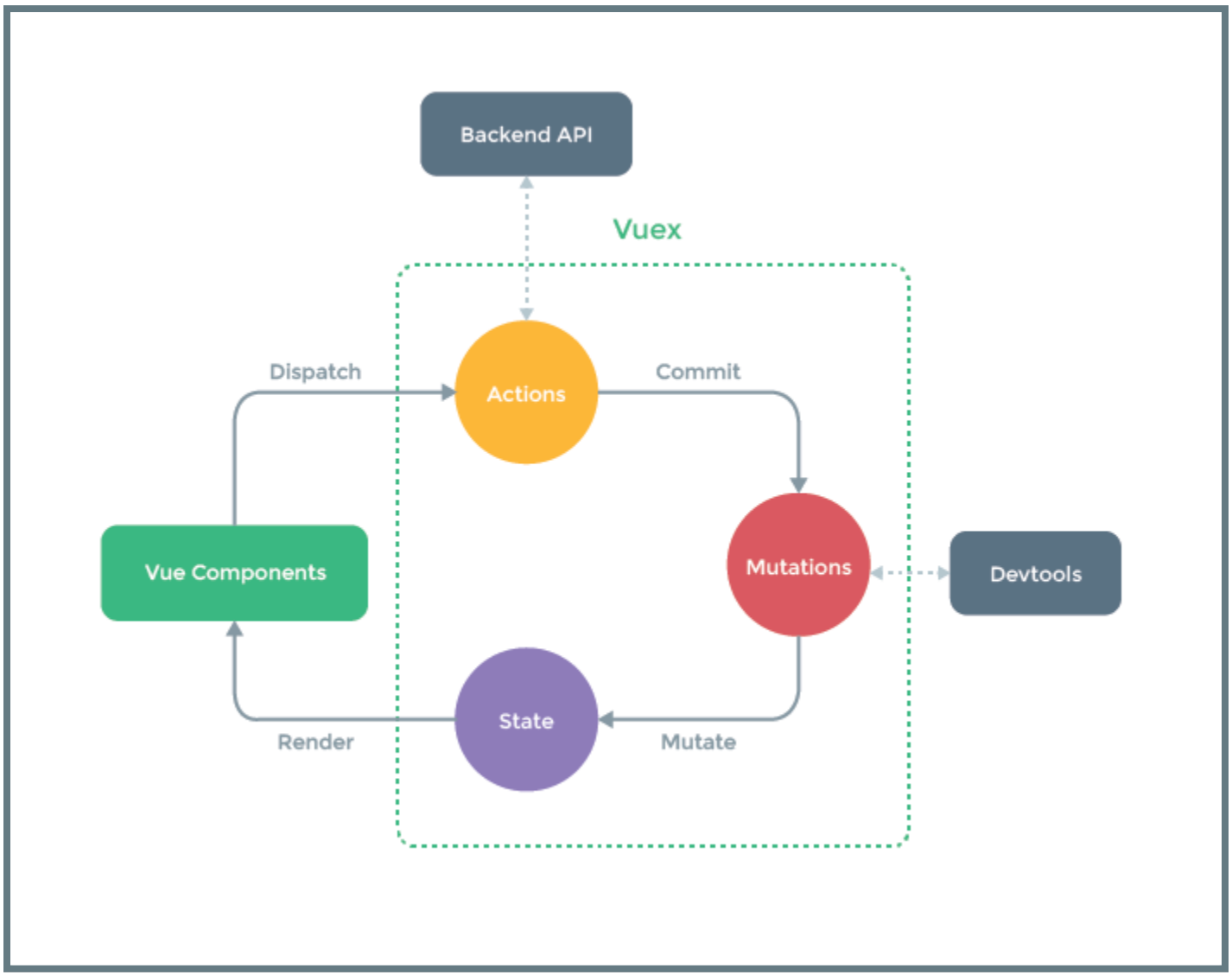
PACZKI

- cq (https://github.com/lukaszb/cq)
- eventsourcing (https://github.com/johnbywater/eventsourcing)
- django-eventsourcing (soon)
- django-cqrs (soon)

FRONTEND

- http://vuex.vuejs.org

VUE.JS


Backend API
Vuex
Dispatch
Actions
Commit
Vue Components
Mutations
Devtools
State
Render
Mutate