Рефакторинг модели домена

- CQRS
- DDD
- Expression Trees
- Lifehacks

Максим Аршинов

предприниматель, спикер, преподаватель

max@hightech.group
https://habr.com/users/marshinov

Мотивация

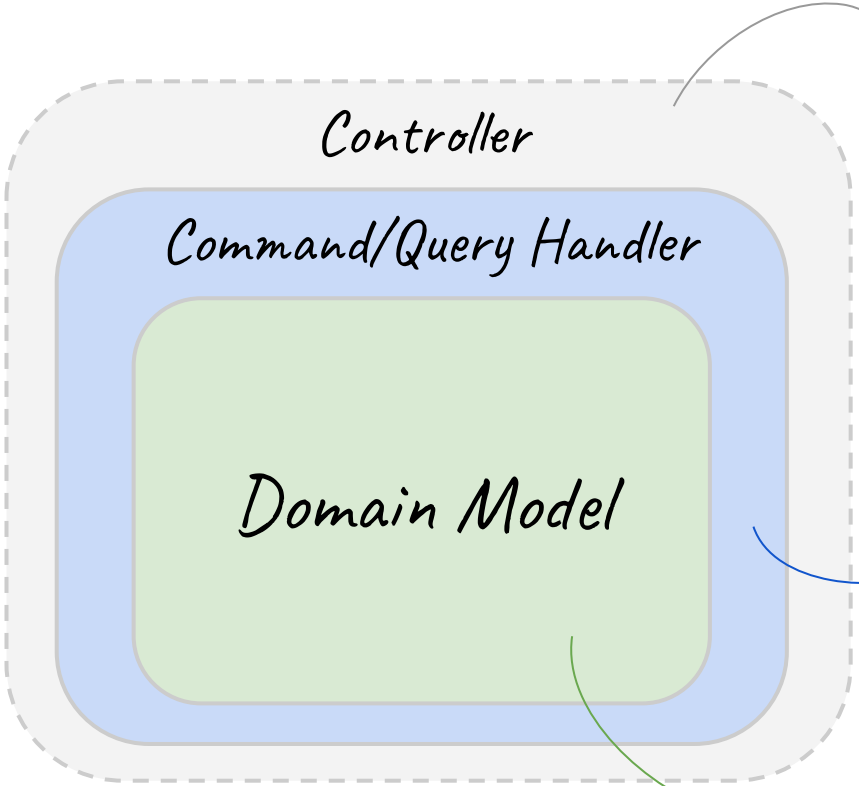
1. Меньше однообразных задач
2. Меньше кода
3. Меньше багов
4. Больше фич
5. Выше прогнозируемость

Инструменты

1. Refactoring
2. Функциональное программирование
3. DDD
4. CQRS/Vertical Slices
5. Expression Trees
6. Декларативный стиль

Альтернатива

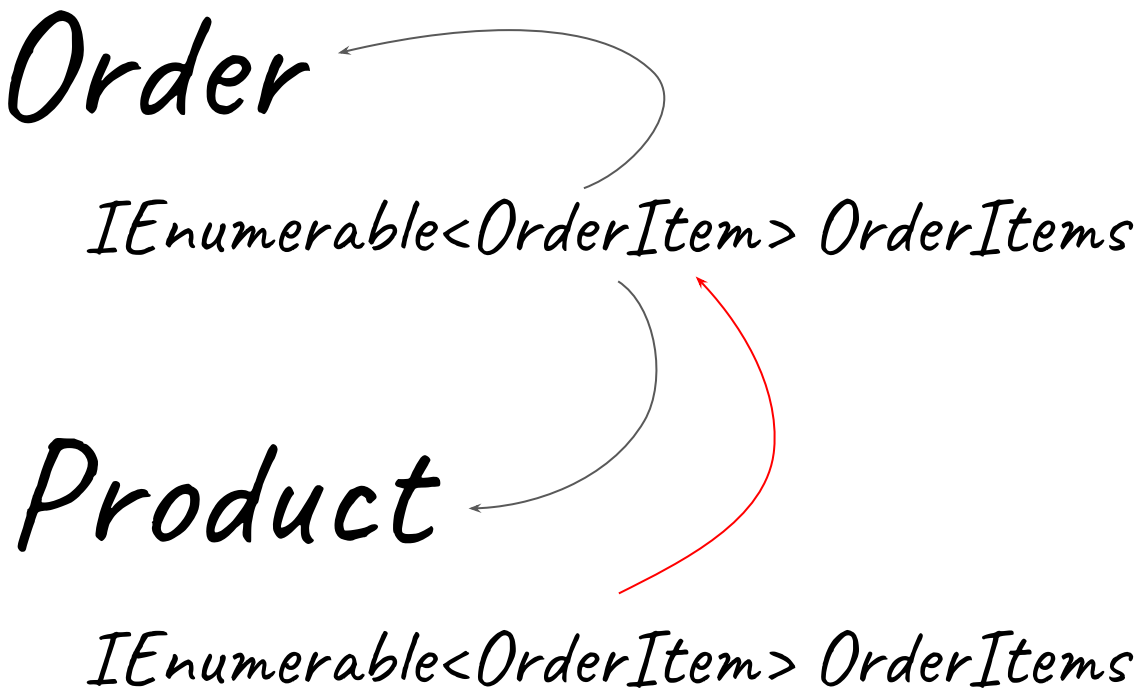
Варианты использования вместо слоев

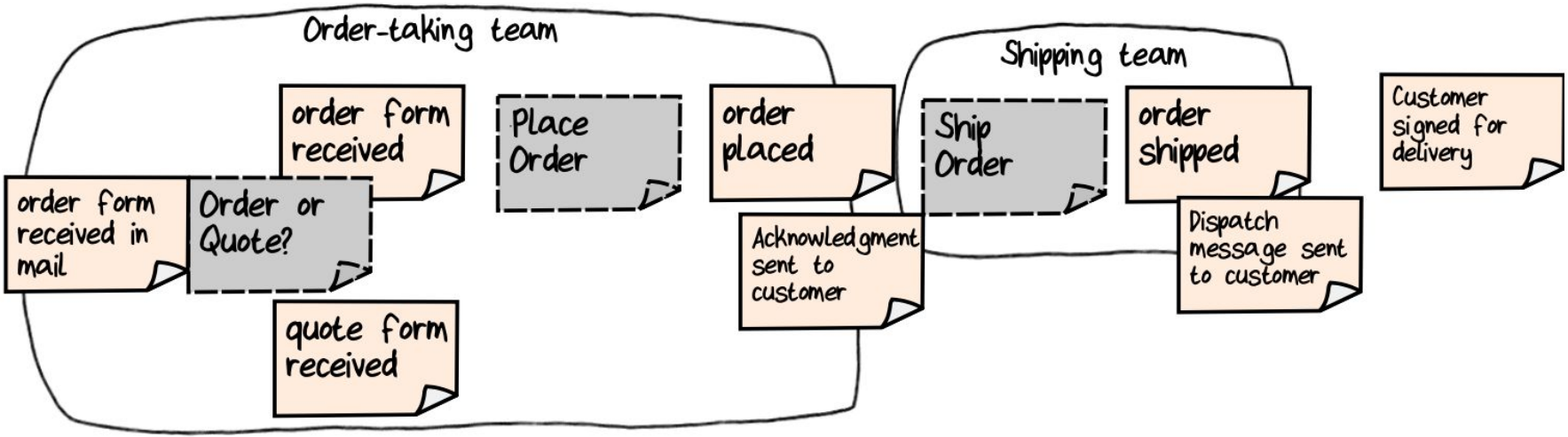


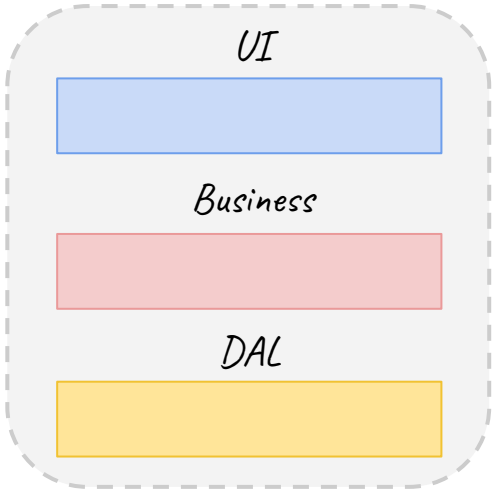
Роутинг
Сериализация/десериализация
Model Binding
Swagger
Простая валидация
Простая проверка прав доступа

Application Layer
Pipeline
Cross-cutting concerns
Интеграционные тесты

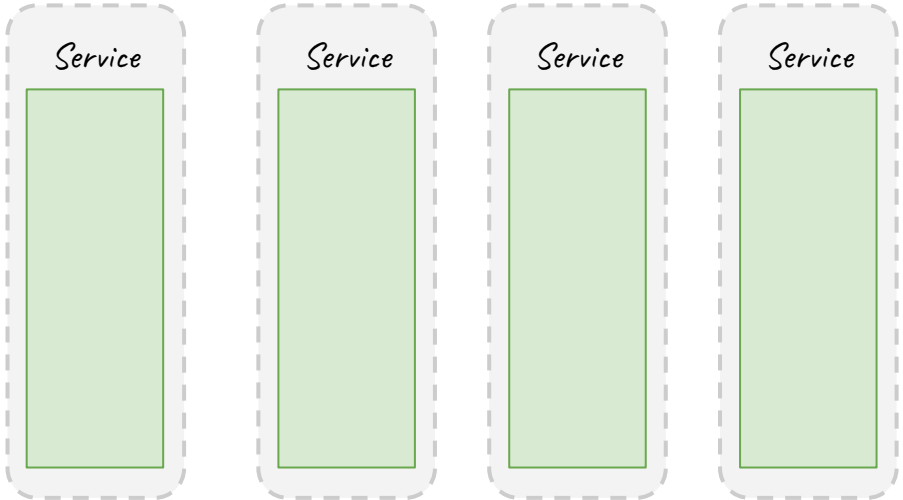
Бизнес-логика
Юнит-тесты


Order
IEnumerable<OrderItem> OrderItems
Product
IEnumerable<OrderItem> OrderItems


Order-taking team
Shipping team
order form received in mail
Order or Quote?
order form received
quote form received
Place Order
order placed
Acknowledgment sent to customer
Ship Order
order shipped
Dispatch message sent to customer
Customer signed for delivery

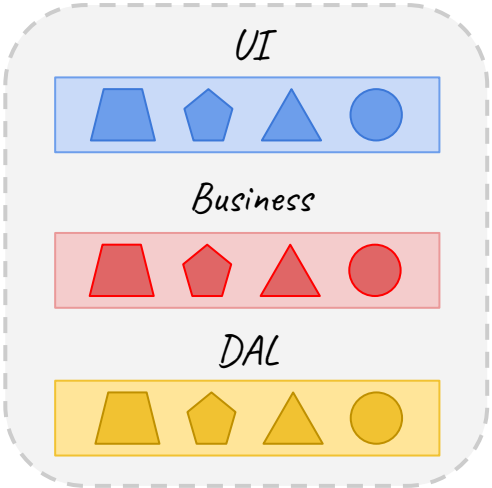


«Ужасный» монолит

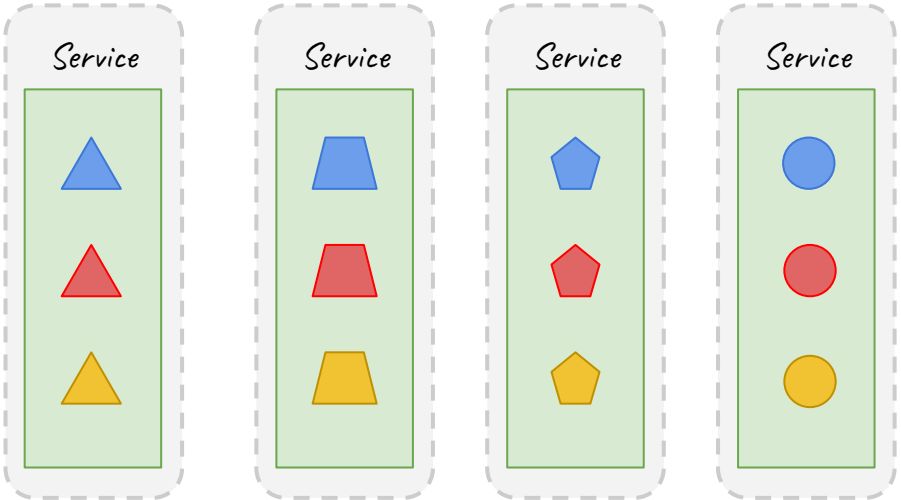


«Прекрасные» микросервисы

с очаровательной сложностью инфраструктуры

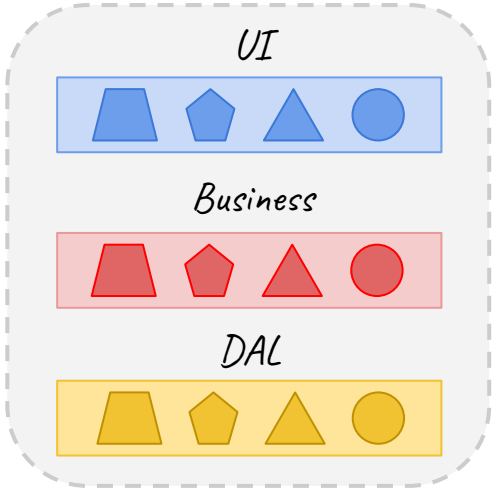


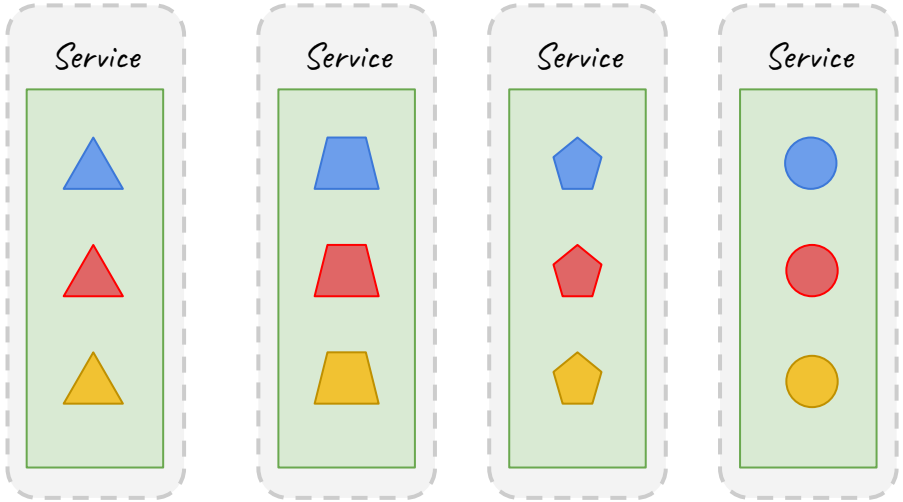
«Ужасный» монолит



«Прекрасные» микросервисы

с очаровательной сложностью инфраструктуры



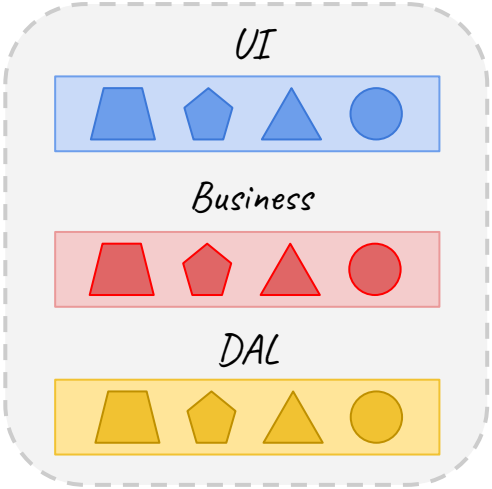


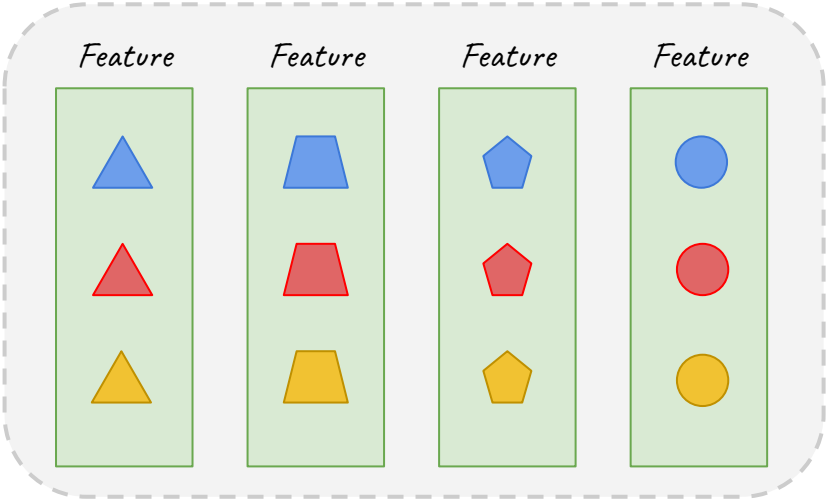
«Ужасный» монолит

Cargo-cult

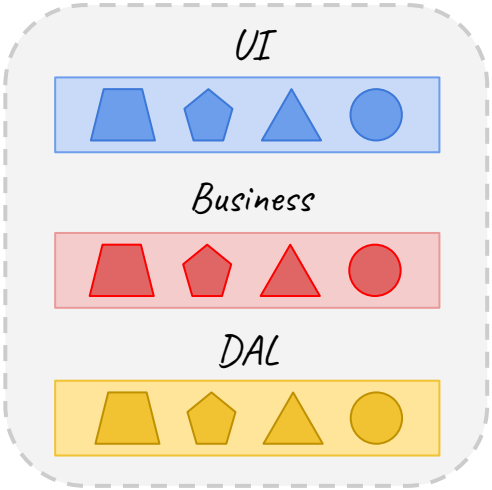
«Прекрасные» микросервисы

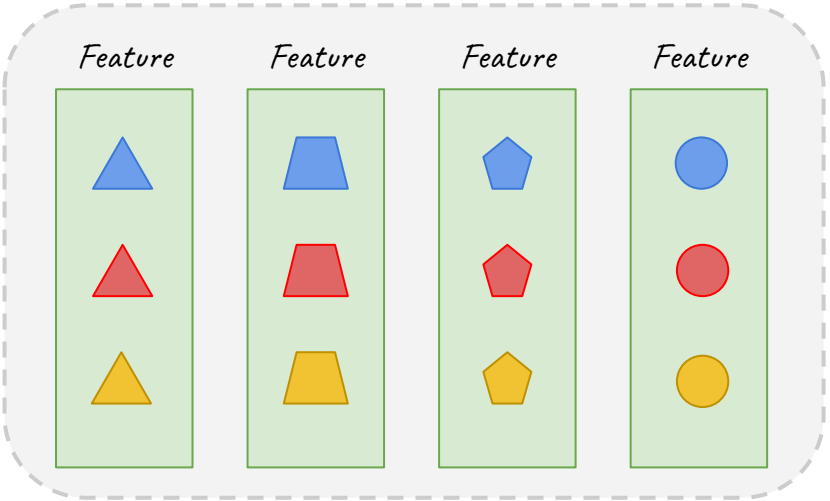
с очаровательной сложностью инфраструктуры


UI
Business
DAL


Feature
Feature
Feature
Feature


Cargo-cult



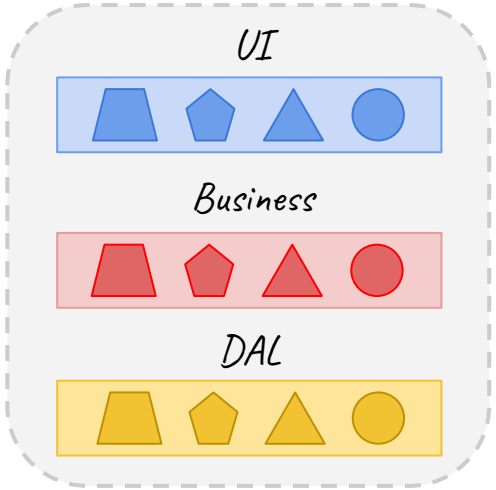


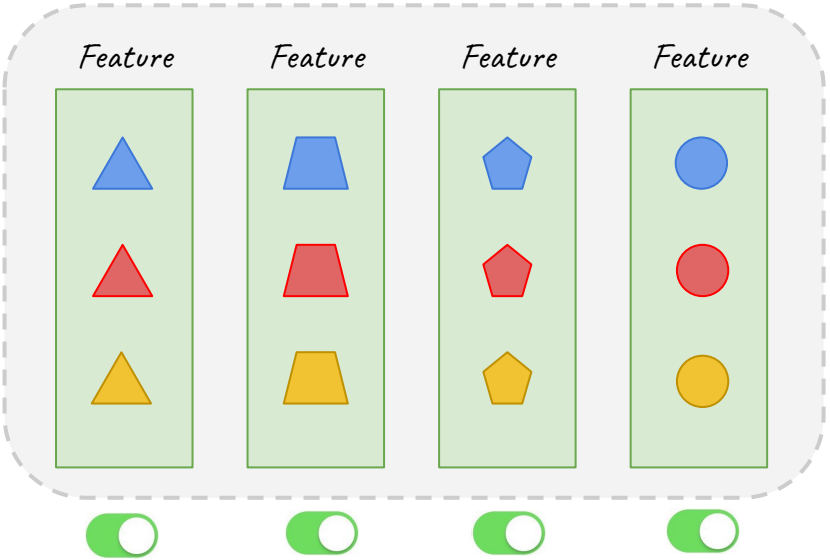
By Layer

Cargo-cult

By Feature

Cross-cutting concerns в подарок (без регистрации и смс)



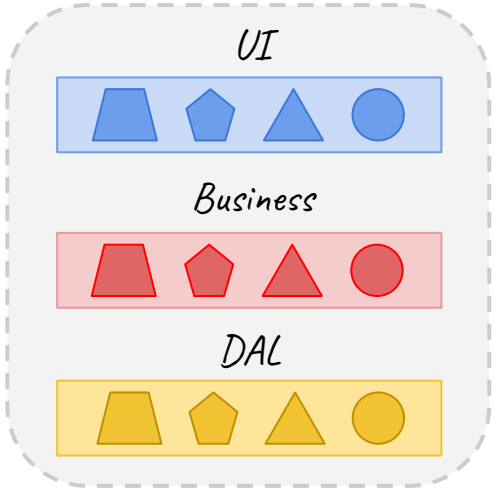


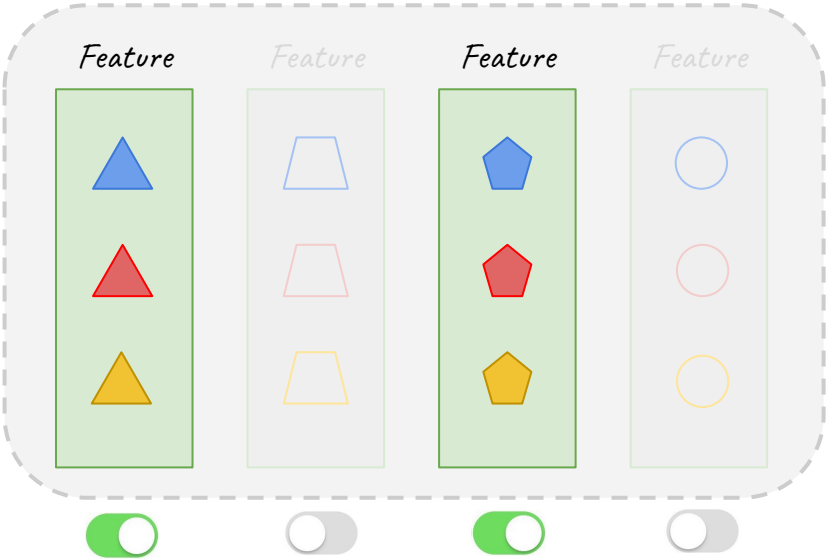
By Layer

Cargo-cult

By Feature

Cross-cutting concerns в подарок (без регистрации и смс)



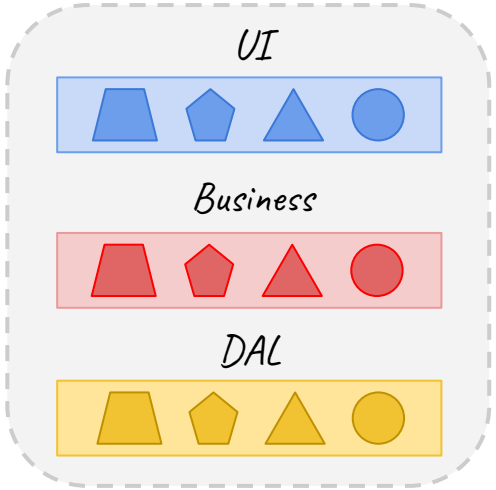


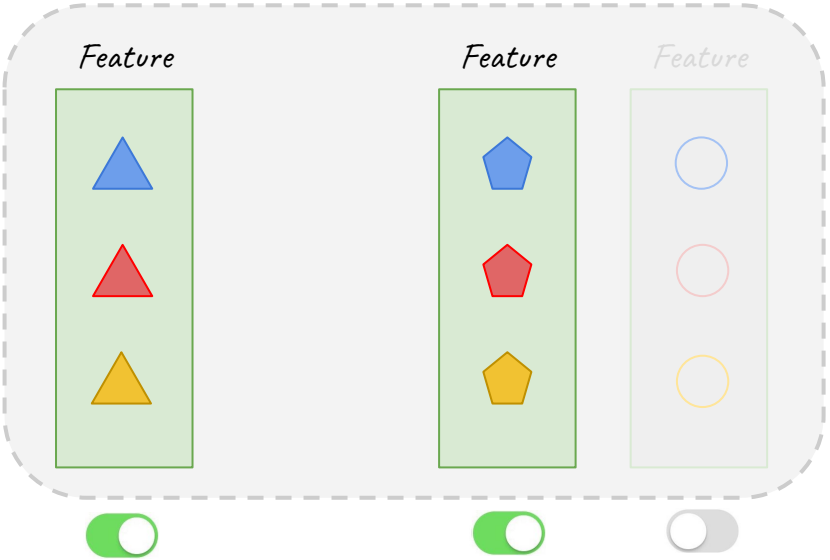
By Layer

Cargo-cult

By Feature

Cross-cutting concerns в подарок (без регистрации и смс)



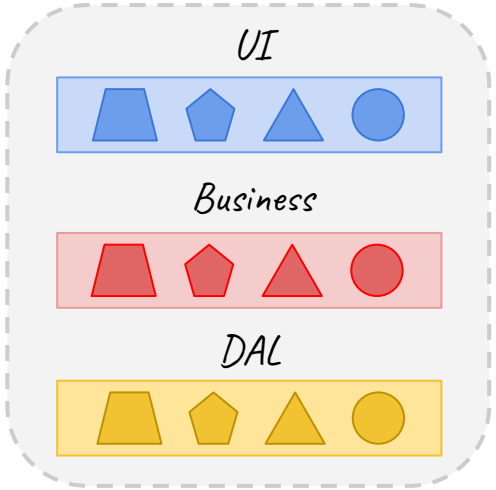


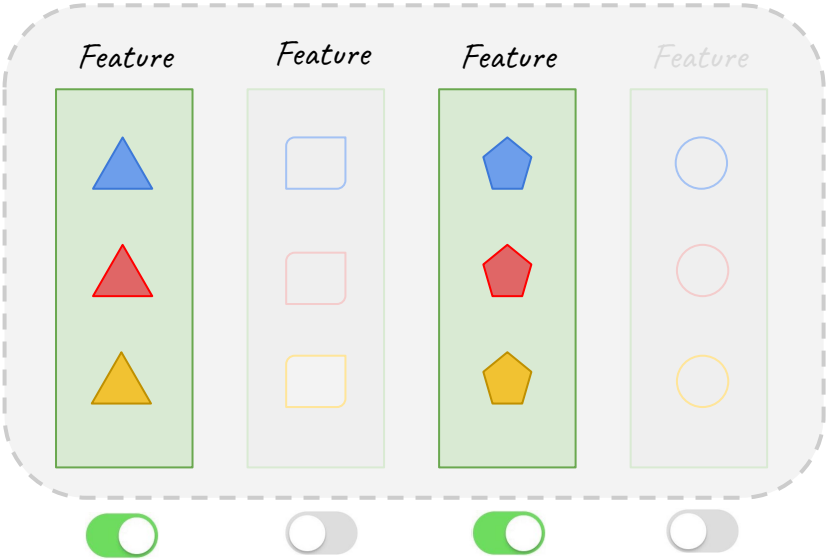
By Layer

Cargo-cult

By Feature

Cross-cutting concerns в подарок (без регистрации и смс)



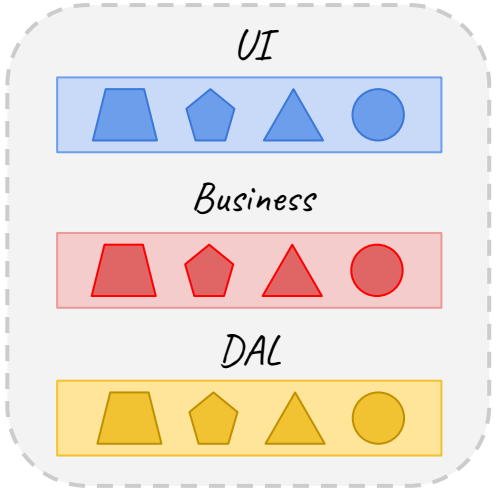


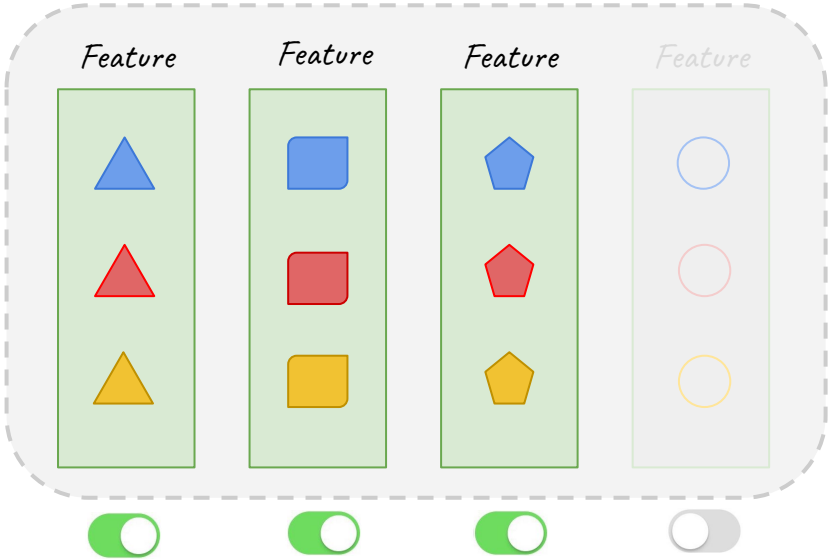
By Layer

Cargo-cult

By Feature

Cross-cutting concerns в подарок (без регистрации и смс)



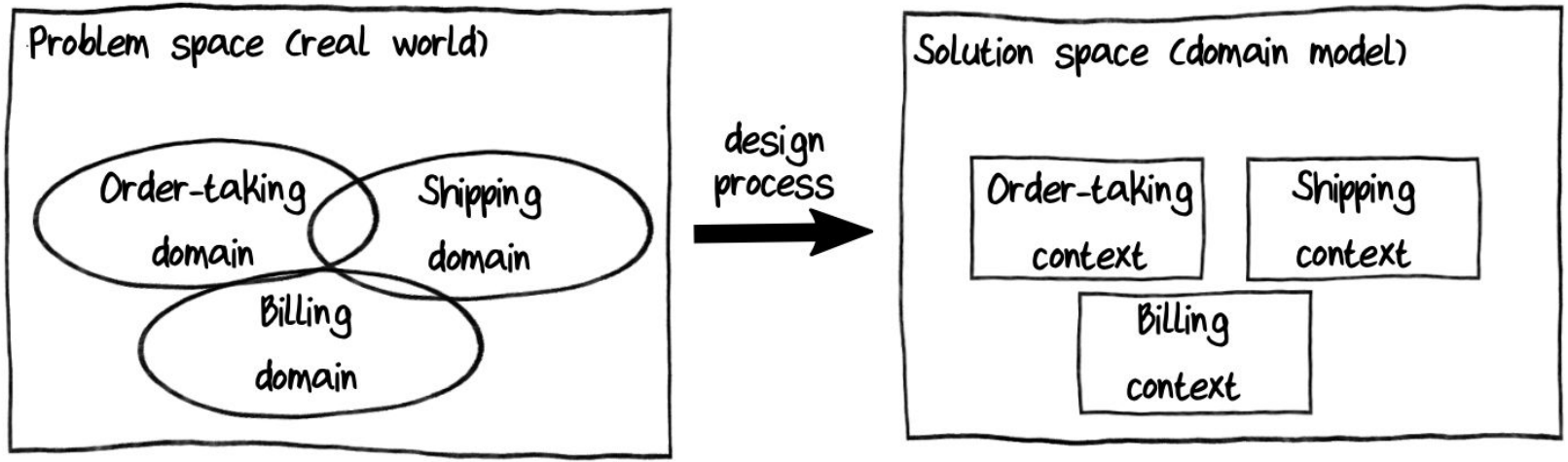


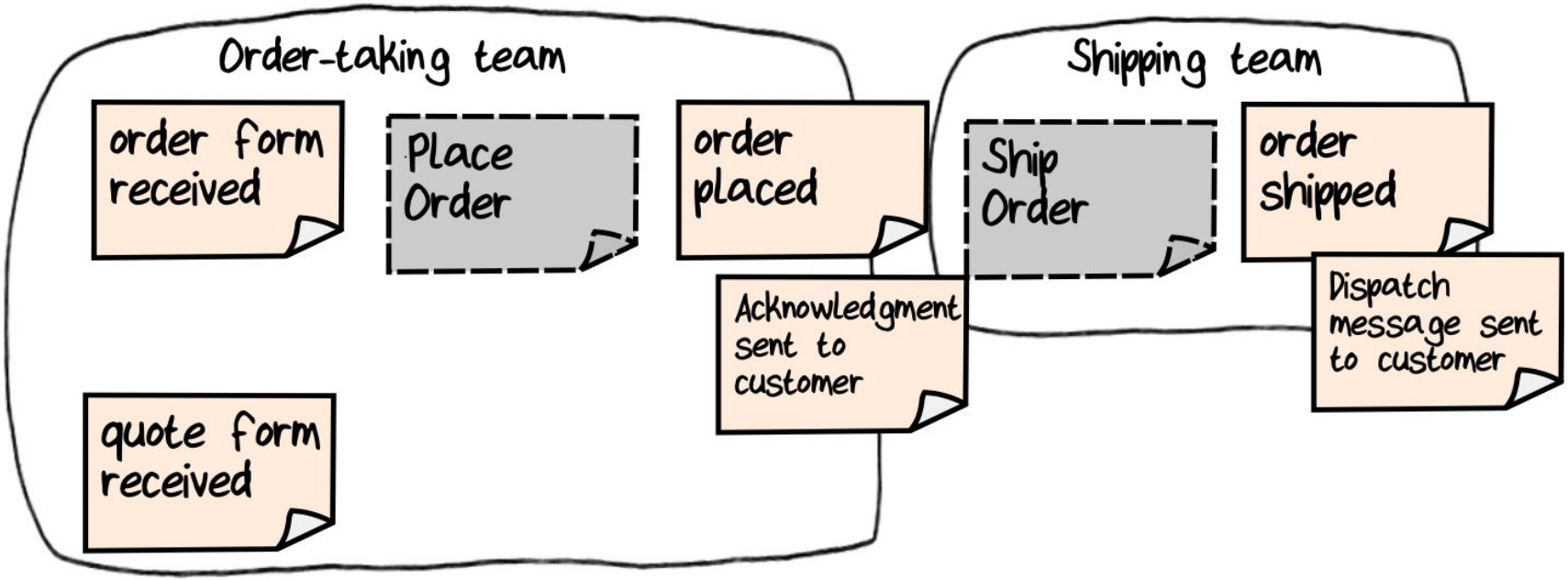
By Layer

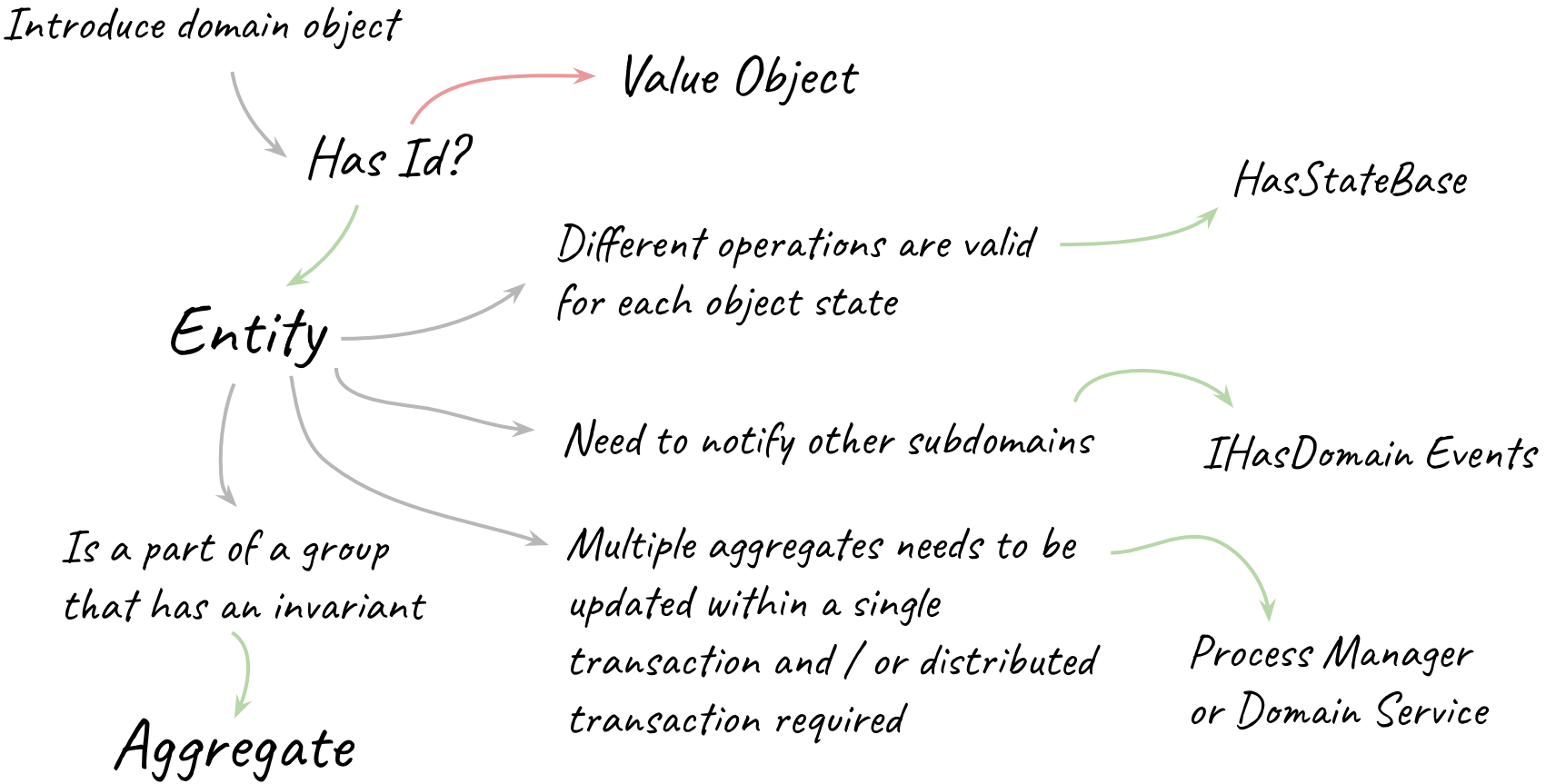
Cargo-cult

By Feature

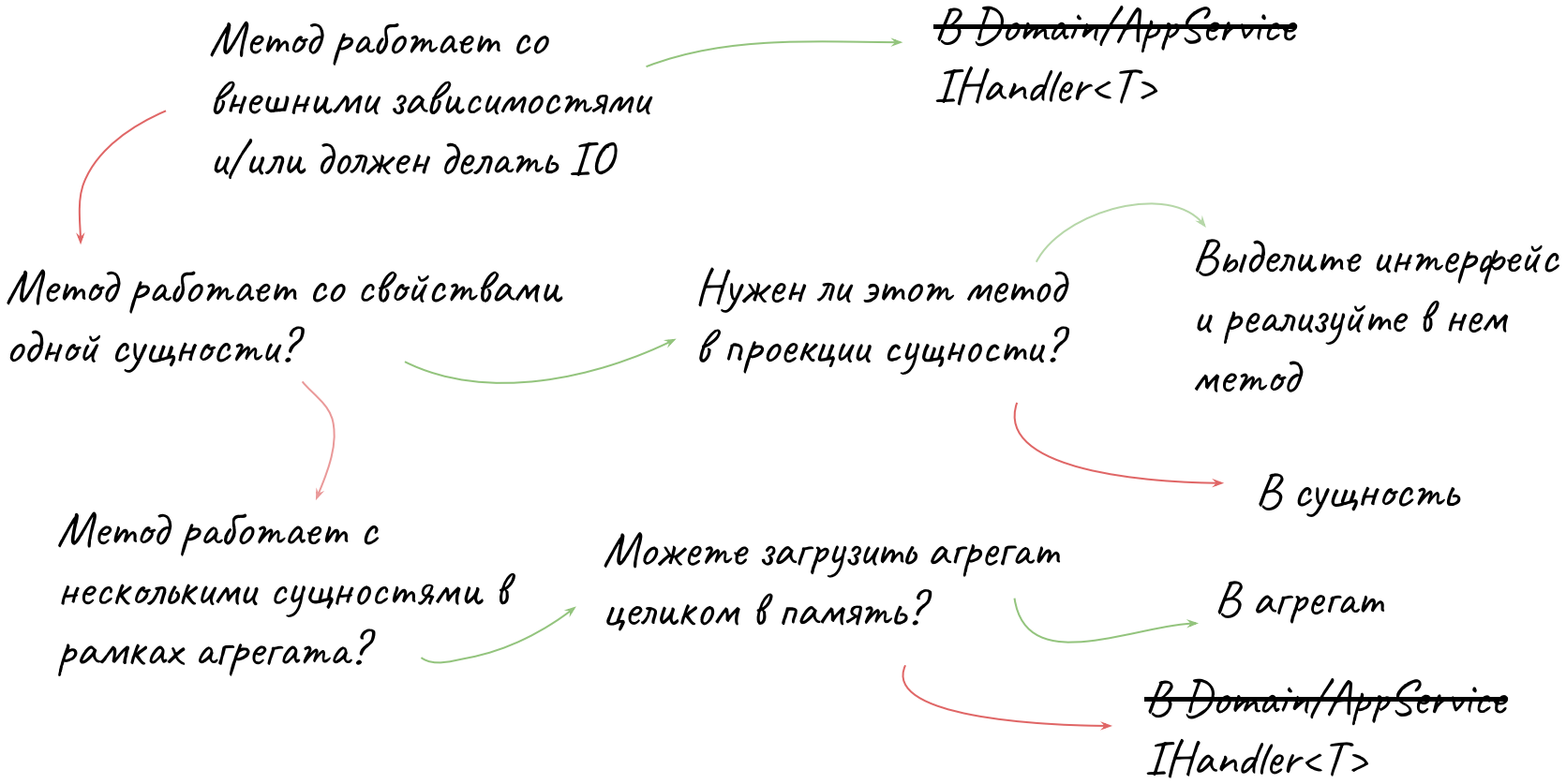
Cross-cutting concerns в подарок (без регистрации и смс)

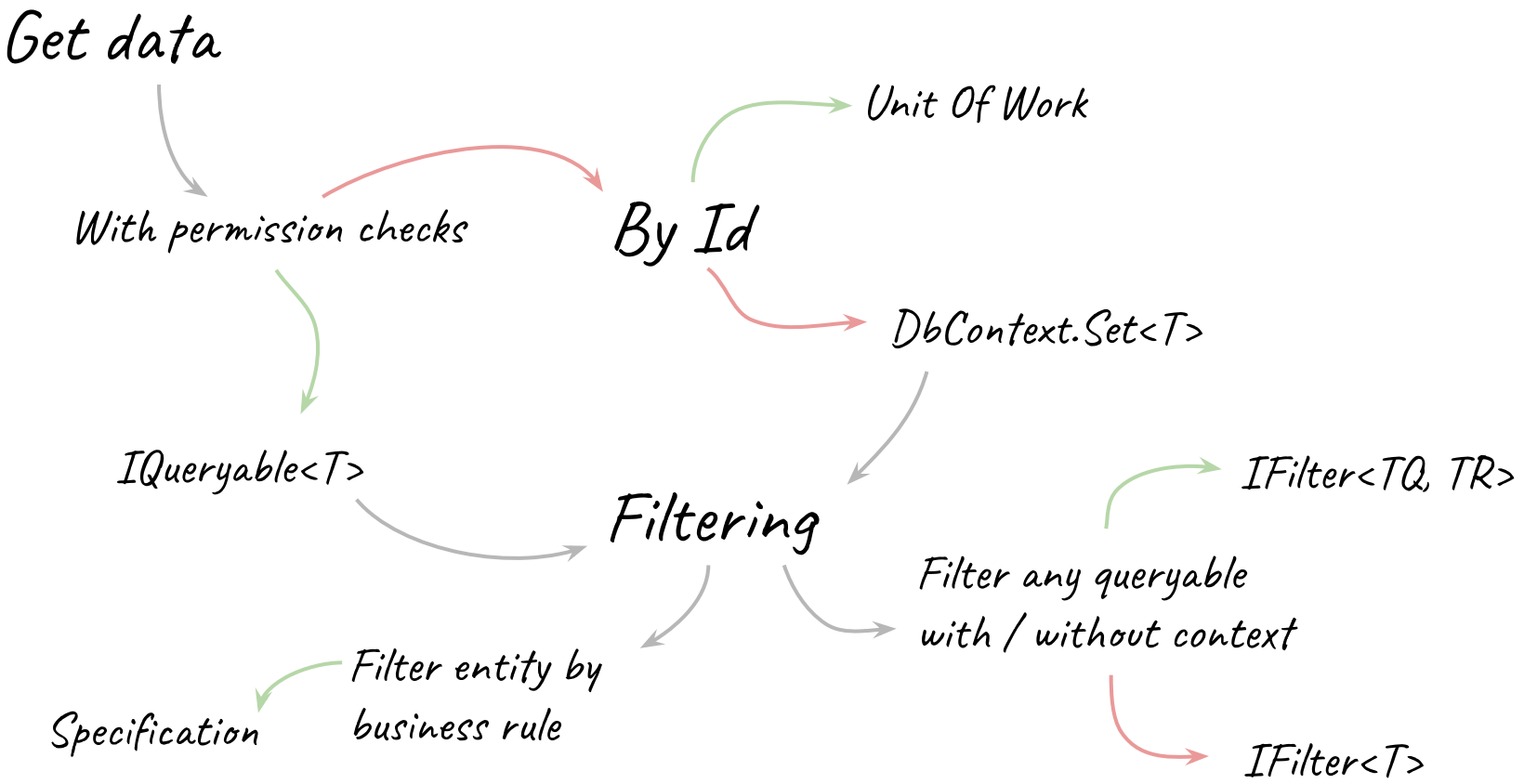

Problem space (real world)
Order-taking domain
Shipping domain
Billing domain
design process
Solution space (domain model)
Order-taking context
Shipping context
Billing context

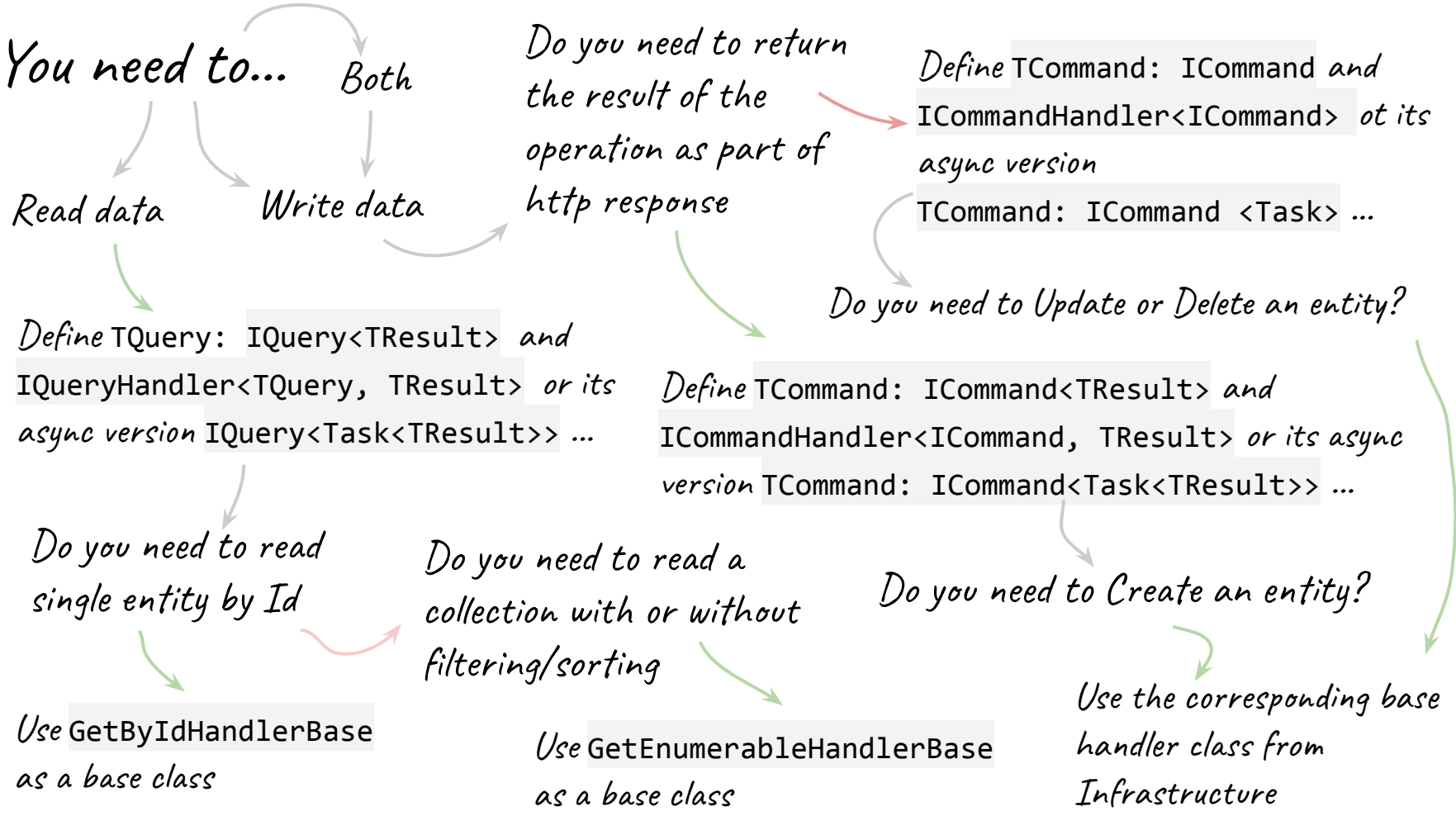

Order-taking team
order form received
Place Order
order placed
quote form received
Acknowledgment sent to customer
Shipping team
Ship Order
order shipped
Dispatch message sent to customer

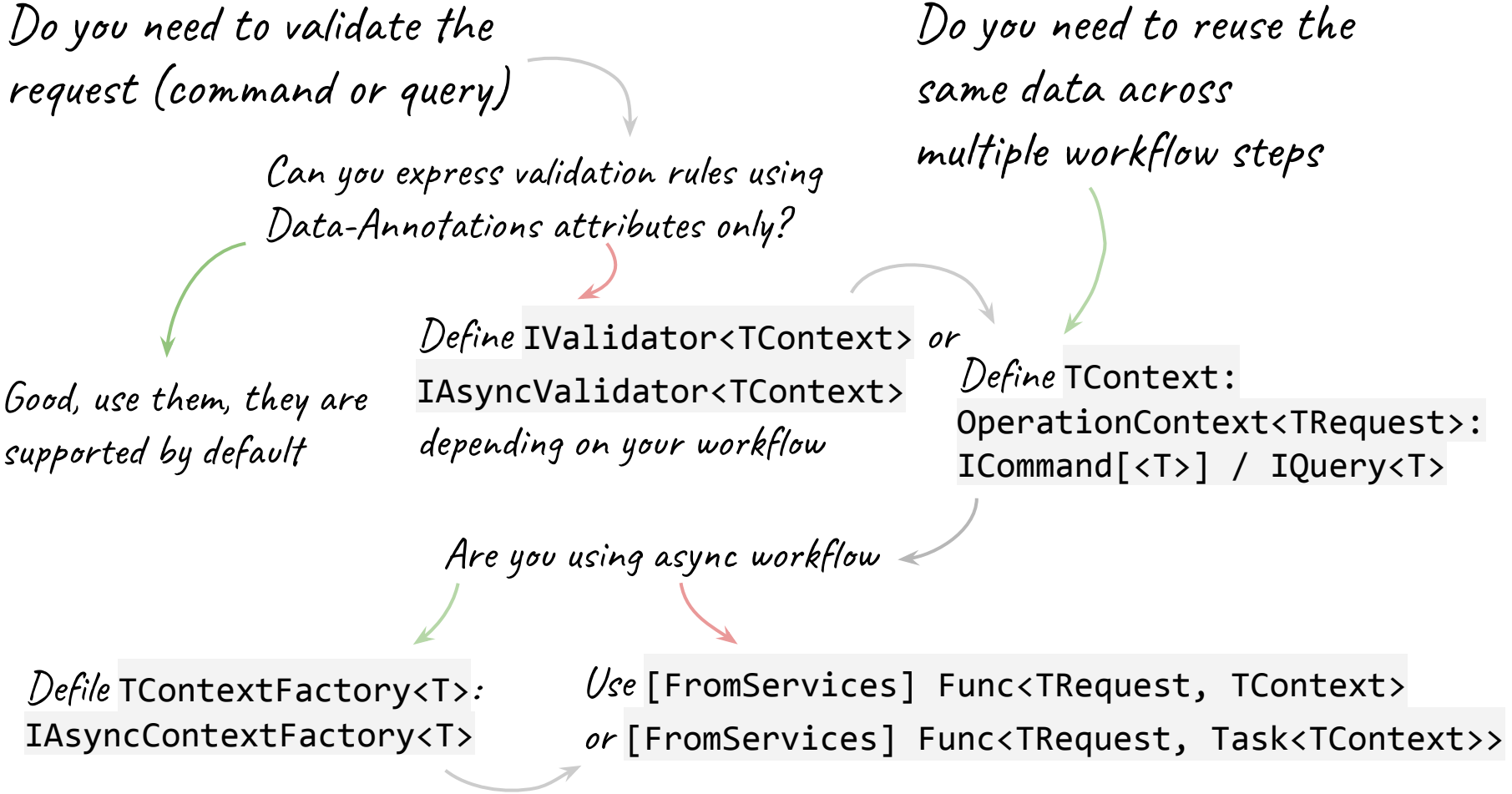

Introduce domain object
Has Id?
Value Object
Entity
Different operations are valid for each object state
HasStateBase
Need to notify other subdomains
IHasDomain Events
Is a part of a group that has an invariant
Aggregate
Multiple aggregates needs to be updated within a single transaction and / or distributed transaction required
Process Manager or Domain Service

Куда мне положить метод?




Get data
With permission checks
By Id
Unit Of Work
DbContext.Set<T>
IQueryable<T>
Filtering
IFilter<TQ, TR>
Filter any queryable
with / without context
IFilter<T>
Filter entity by
business rule
Specification


You need to...
Both
Read data
Write data
Do you need to return the result of the operation as part of http response
Define TCommand: ICommand and ICommandHandler<ICommand> ot its async version TCommand: ICommand <Task> ...
Do you need to Update or Delete an entity?
Define TQuery: IQuery<TResult> and IQueryHandler<TQuery, TResult> or its async version IQuery<Task<TResult>> ...
Define TCommand: ICommand<TResult> and ICommandHandler<ICommand, TResult> or its async version TCommand: ICommand<Task<TResult>> ...
Do you need to read single entity by Id
Do you need to read a collection with or without filtering/sorting
Do you need to Create an entity?
Use GetByIdHandlerBase as a base class
Use GetEnumerableHandlerBase as a base class
Use the corresponding base handler class from Infrastructure


Do you need to validate the request (command or query)
Can you express validation rules using Data-Annotations attributes only?
Good, use them, they are supported by default
Define IValidator<TContext> or IAsyncValidator<TContext> depending on your workflow
Do you need to reuse the same data across multiple workflow steps
Define TContext: OperationContext<TRequest>: ICommand[<T>] / IQuery<T>
Are you using async workflow
Define TContextFactory<T>: IAsyncContextFactory<T>
Use [FromServices] Func<TRequest, TContext> or [FromServices] Func<TRequest, Task<TContext>>

```
public class User : EntityBase
{
    public Contact Contact { get; protected set; }

    public UserName? UserName { get; protected set; }

    public User(Contact contact, UserName? userName = null)
    {
        Contact = contact;
        UserName = userName;
    }
}
```

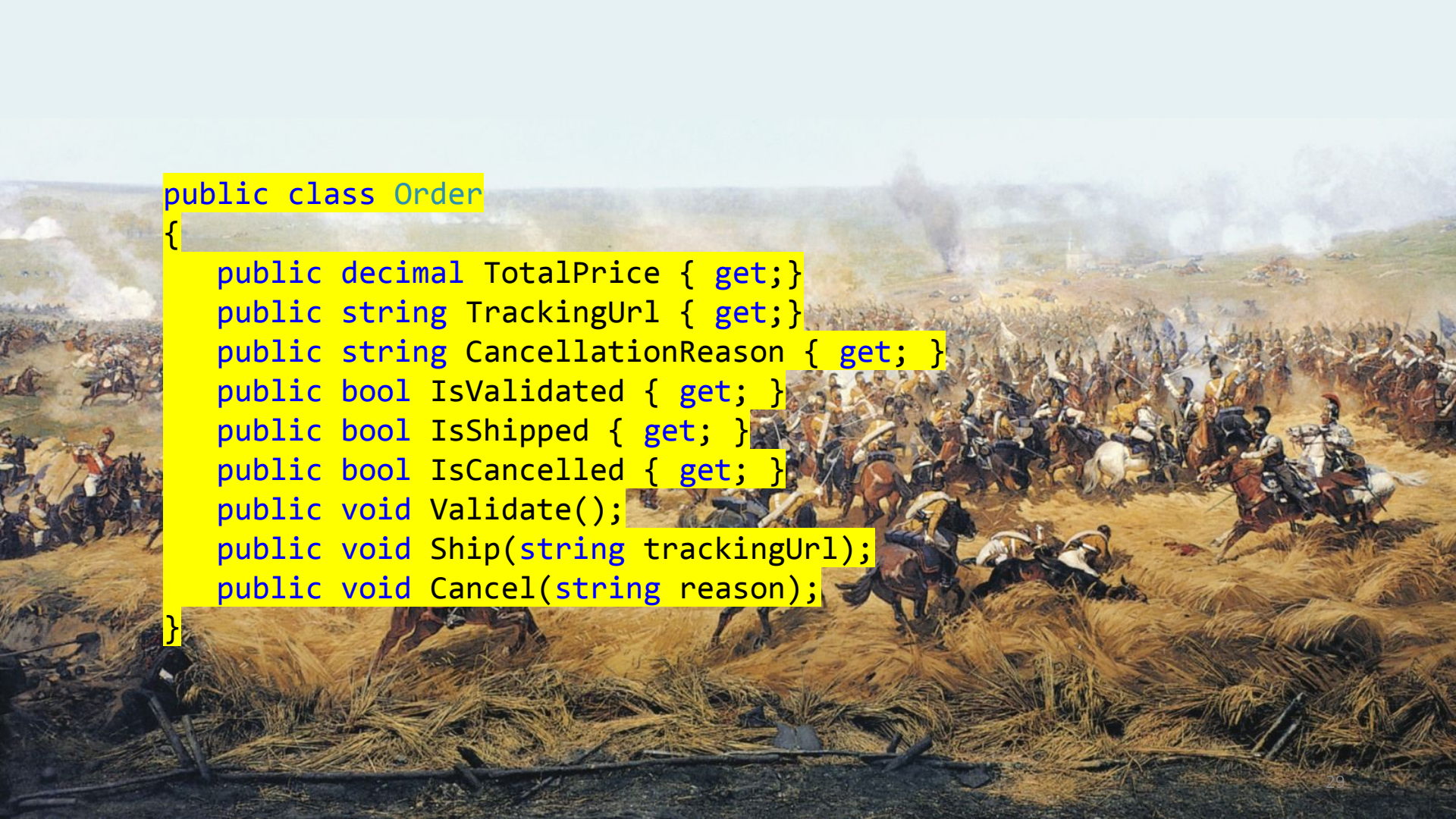
Откуда берутся пользователи?

```
public class User : EntityBase
{
    public Contact Contact { get; protected set; }

    public UserName? UserName { get; protected set; }

    public User (SignUp command)
    {
        Contact = command.Contact;
        UserName = command.UserName;
    }
}
```

- Пользователь может зарегистрироваться сам
- …

```
public class User : EntityBase
{
    public User? Invitee { get; protected set;}

    public Contact Contact { get; protected set; }

    public UserName? UserName { get; protected set; }

    public User (InviteUser command)
    {
        Contact = command.Contact;
        UserName = command.UserName;
        Invitee = command.Invitee;
    }
}
```

- Пользователь может зарегистрироваться сам
- Или его может пригласить другой пользователь



```
public class Order
{
    public decimal TotalPrice { get;}
    public string TrackingUrl { get;}
    public string CancellationReason { get; }
    public bool IsValidated { get; }
    public bool IsShipped { get; }
    public bool IsCancelled { get; }
    public void Validate();
    public void Ship(string trackingUrl);
    public void Cancel(string reason);
}
```

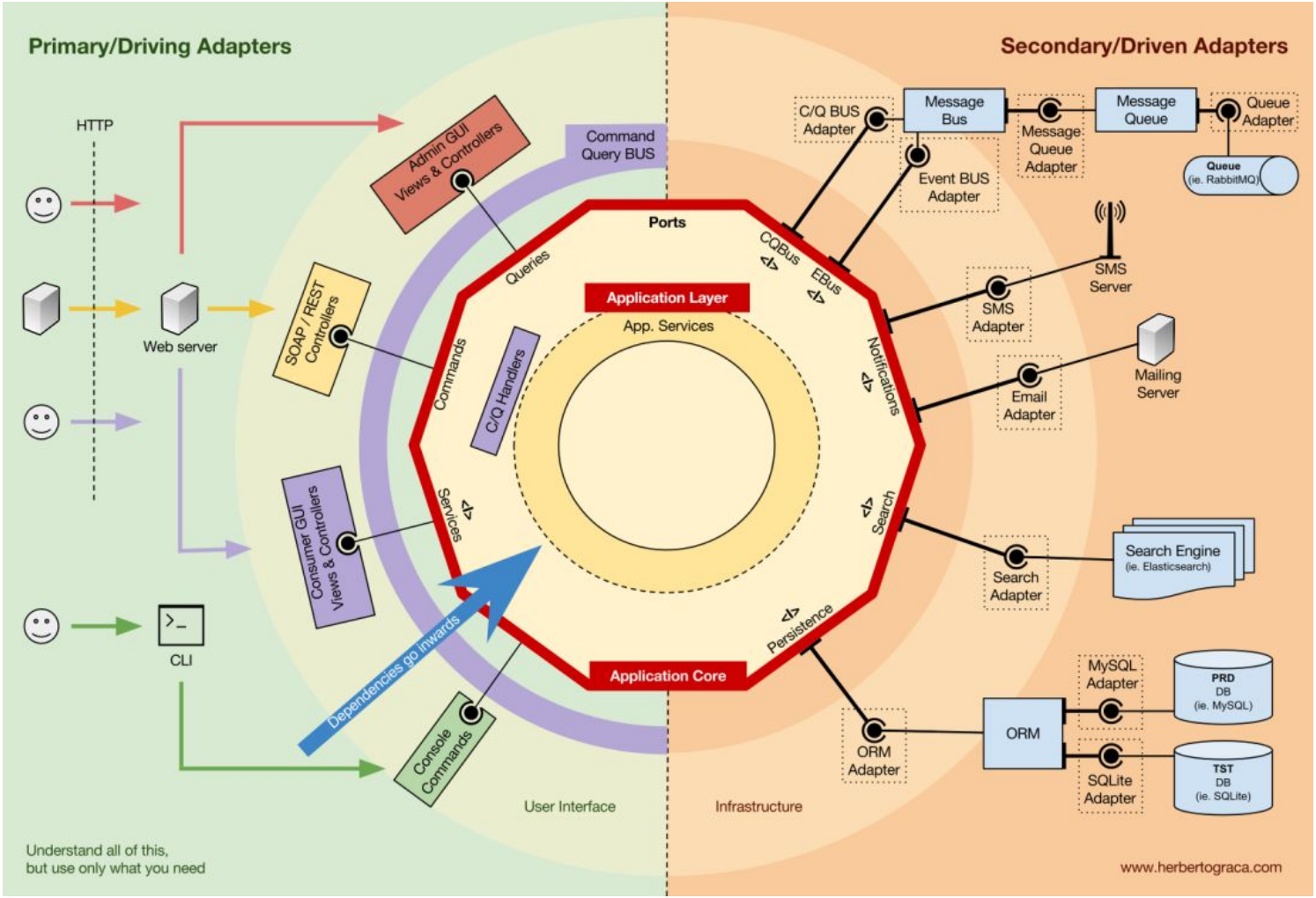
```
public abstract class Document<T> : Document
{ try {
    public Document(UploadedFile file, T meta);

    public void Update(UploadedFile file, T meta);
} }
  catch(FileStorageException)
  catch(SqlException)
```

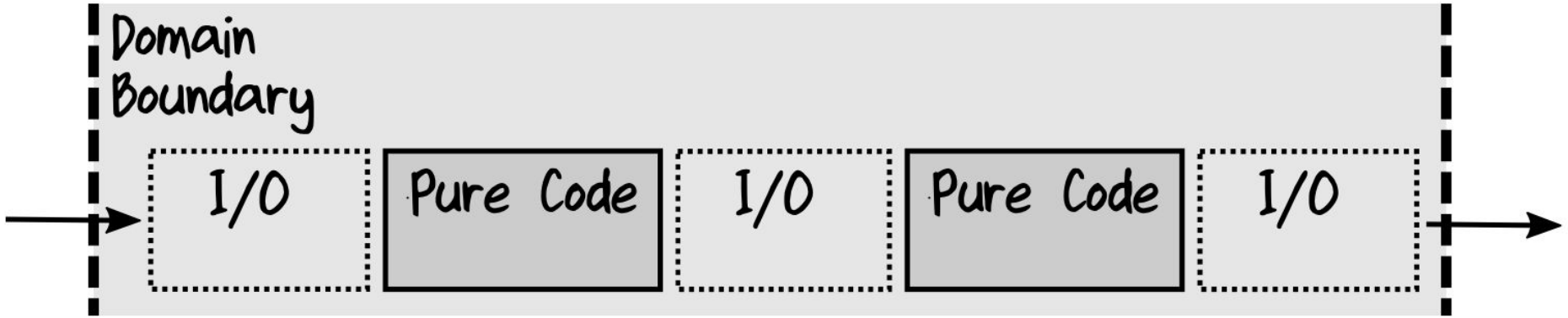
Где мои гарантии?

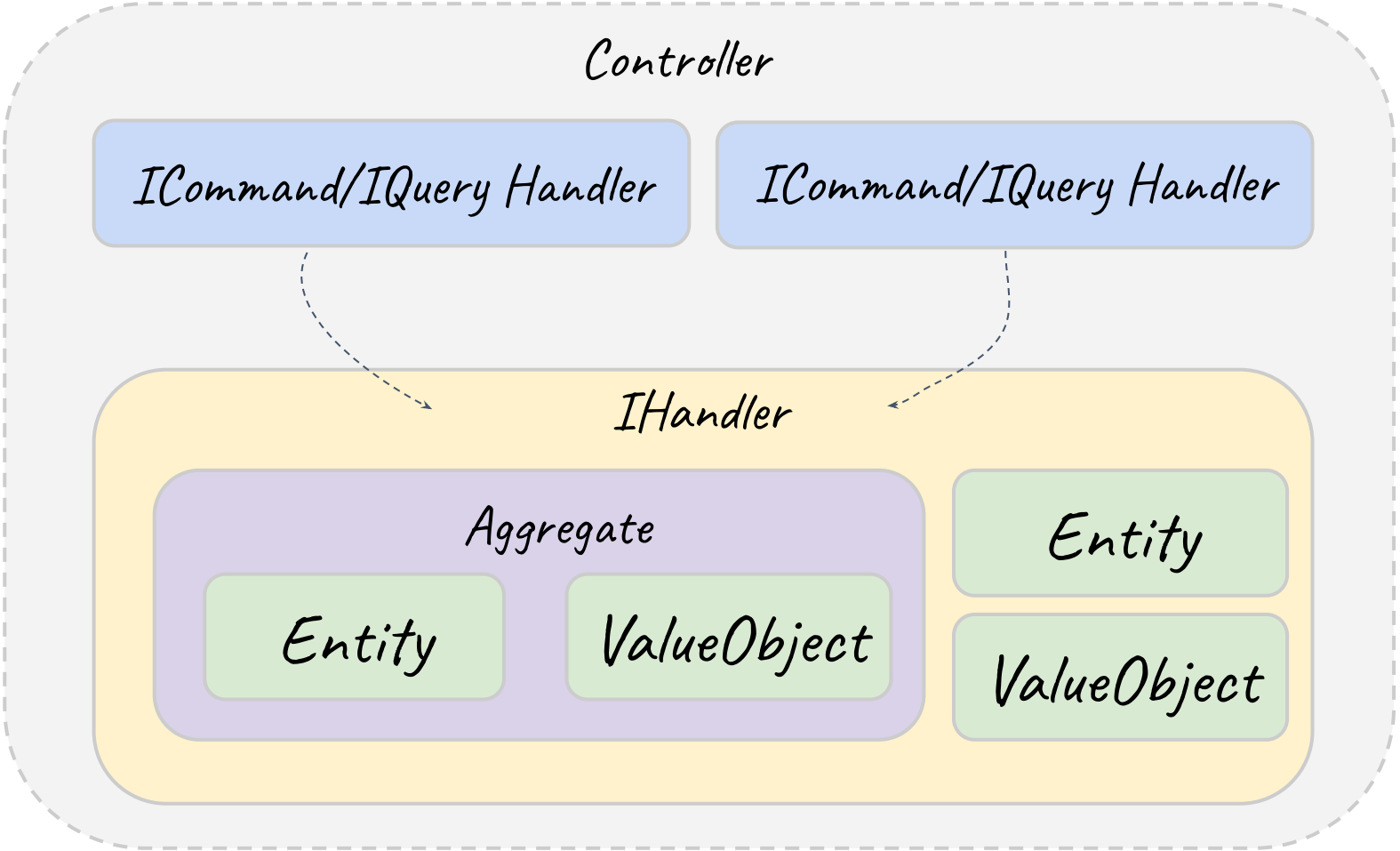
```
public abstract class Document<T> : Document
{ try {
    internal Document(UploadedFile file, T meta

    internal void Update(UploadedFile file, T meta);
} }
  catch(FileStorageException)
  catch(SqlException)
```

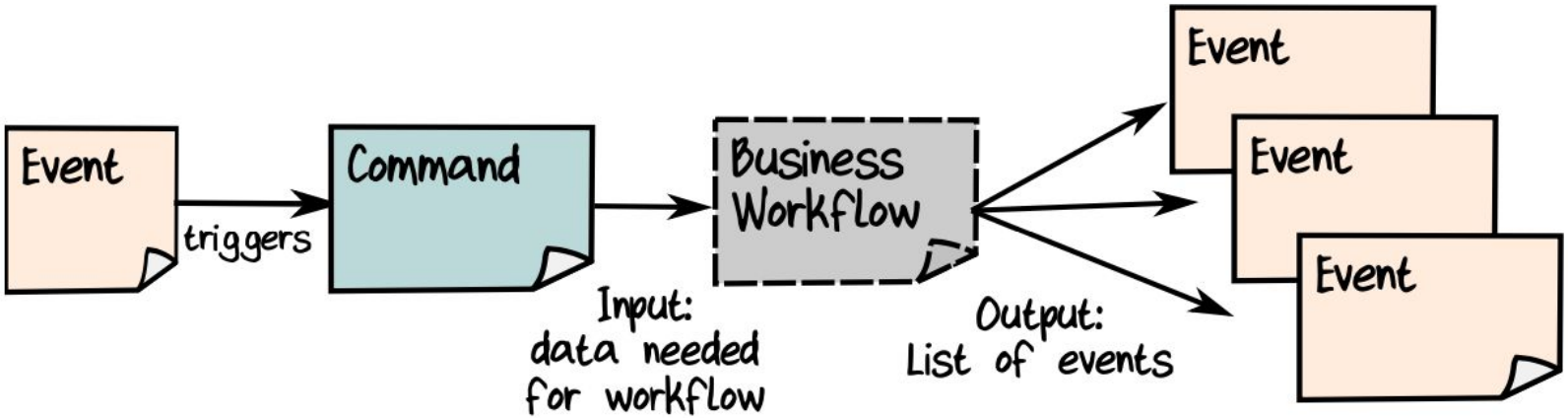
Где мои гарантии?

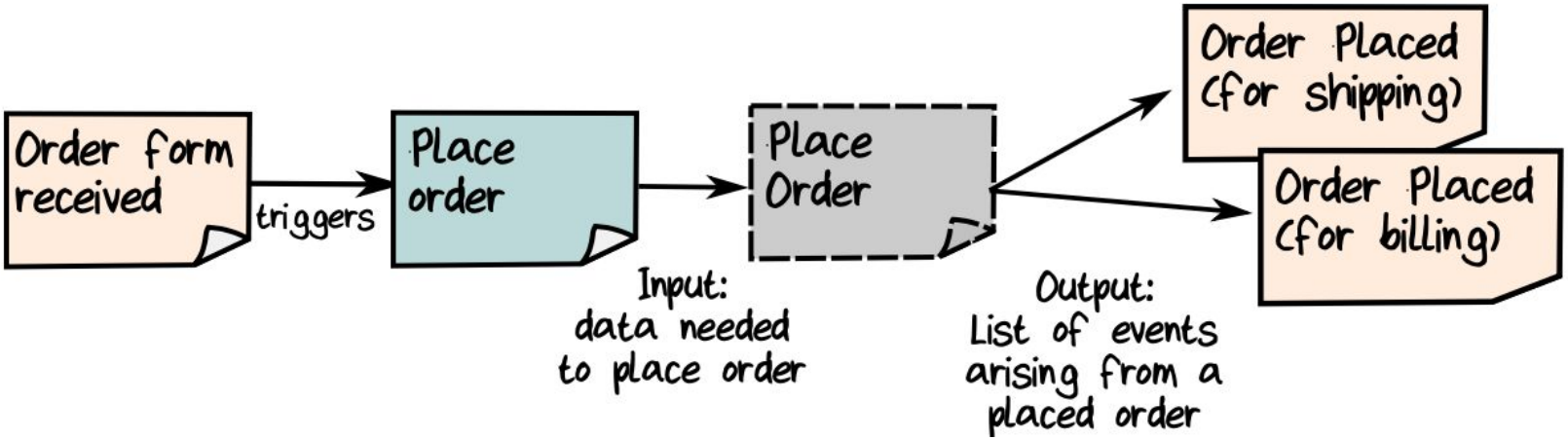
```
public void Update<T>(Document<T> entity, IFormFile file, T data) {
    UploadedFile uploaded = fileStorage.Upload(file);
    try { entity.Update(uploaded, data);}
    catch (SqlException se)
    {
        try { fileStorage.Delete(uploaded); }
        catch (FileStorageException fe)
        {
            logger.Fatal(fe.Message);
            throw;
        }
        throw;
    }
}
```


Primary/Driving Adapters
Secondary/Driven Adapters
HTTP
Web server
CLI
Admin GUI Views & Controllers
SOAP / REST Controllers
Consumer GUI Views & Controllers
Console Commands
Command Query BUS
Queries
Commands
Services
C/Q Handlers
Ports
Application Layer
App. Services
Application Core
CQBus
EBus
Notifications
Search
Persistence
C/Q BUS Adapter
Message Bus
Message Queue Adapter
Message Queue
Queue Adapter
Queue (ie. RabbitMQ)
Event BUS Adapter
SMS Server
SMS Adapter
Mailing Server
Email Adapter
Search Engine (ie. Elasticsearch)
Search Adapter
ORM Adapter
ORM
MySQL Adapter
SQLite Adapter
PRD DB (ie. MySQL)
TST DB (ie. SQLite)
Dependencies go inwards
User Interface
Infrastructure
Understand all of this, but use only what you need
www.herbertograca.com


Domain
Boundary
I/O
Pure Code
I/O
Pure Code
I/O


Controller
ICommand/IQuery Handler
ICommand/IQuery Handler
IHandler
Aggregate
Entity
ValueObject
Entity
ValueObject


Event
triggers
Command
Input:
data needed
for workflow
Business
Workflow
Output:
List of events
Event
Event
Event


Order form received
triggers
Place order
Place Order
Input: data needed to place order
Output: List of events arising from a placed order
Order Placed (for shipping)
Order Placed (for billing)

cruD

Удалить нельзя оставить


заявление
Прошу уволить меня 15 мая 2019 года с занимаемой должности по собственному желанию, так как имею предложение другой работы, которая меня более устраивает.
25 апреля 2019 года

cruD

Удалить нельзя оставить

cruD

Удалить нельзя, оставить

cruD

Soft Delete

cruD

~~Soft Delete~~

cruD

Запрос на возврат денежных средств

cruD

Увольнение