

Переходим  
от Feature-based разработки  
к Domain Driven Design

---



# SkyEng

---

- Skyeng — это онлайн-школа английского языка нового поколения.
- В школе работают профессионалы, помогающие жителям современных мегаполисов выучить английский язык в условиях недостатка времени.



Групповые уроки в формате квеста для детей 10-17 лет. 300 рублей за урок 🧑🏫💻🧑🏫

skyeng

ДЕТЯМ

КОМПАНИЯМ

УЧИТЕЛЯМ

8 (800) 555-45-22

Бесплатный урок

Войти



# Персональные уроки с учителем в онлайн-школе Skyeng

Взрослым  
английский для  
путешествий,  
карьеры и  
развлечений



Детям  
английский и  
математика в  
возрасте от 4 до  
17 лет



Задайте вопрос





# Маркетинг

---

- **Маркетинг** (от англ. marketing «рыночная деятельность») — организационная функция и совокупность процессов **создания, продвижения и предоставления** продукта или услуги покупателям и управление взаимоотношениями с ними с выгодой для организации.



# Маркетинг

---

- Тратить меньше (на продвижение и предоставление)
- Получать больше (увеличение аудитории, создание новых продуктов)
- Деньги!



**ГДЕ ДЕНЬГИ, ЛЕБОВСКИ?!**





# Что требуется от разработки

---

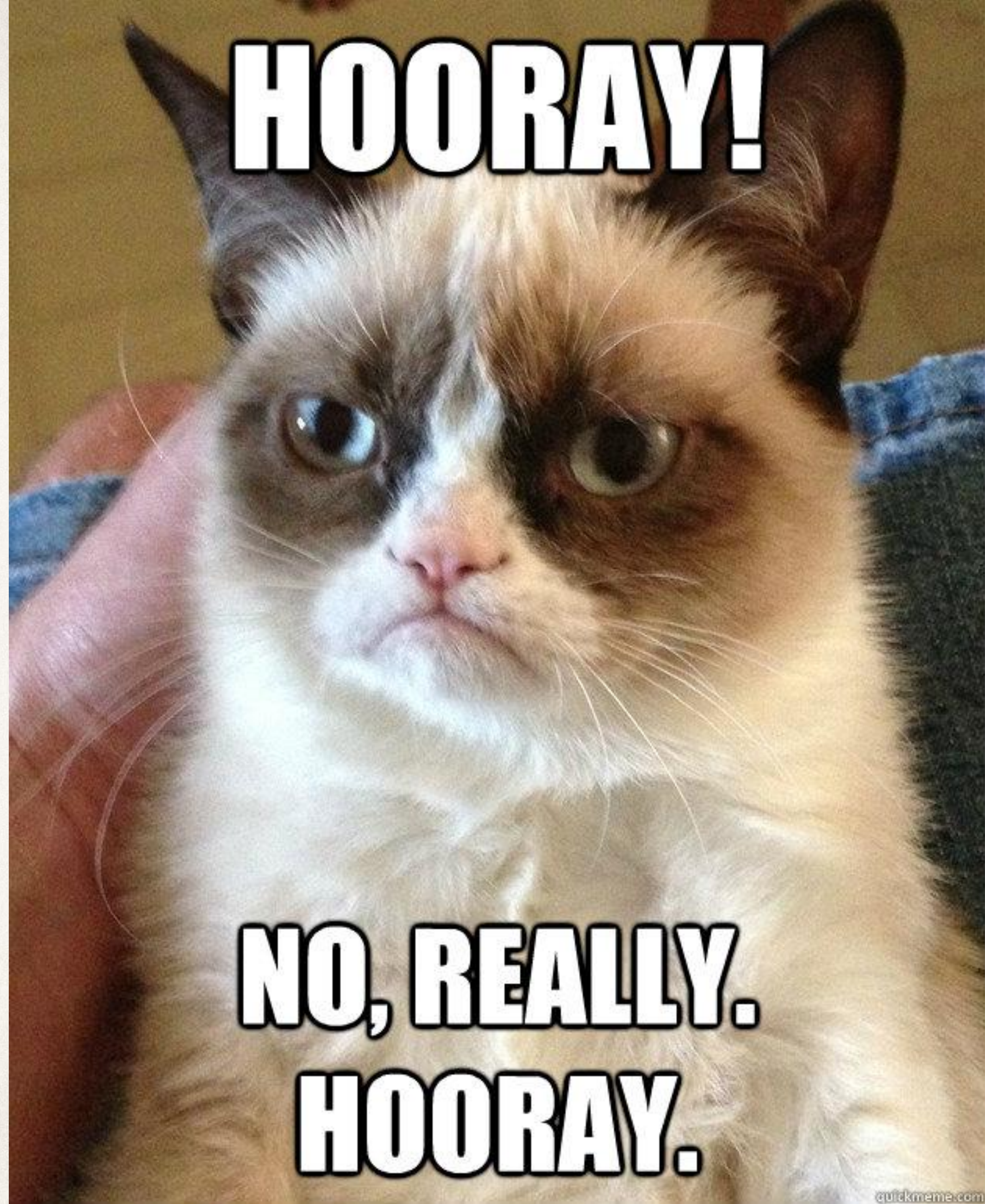
- Качественно
- Быстро
- Дешево





Пример

---





# Задача

---

- Надо сделать виджет
- Дизайн есть!
- Пользователь оставляет заявку
- В календаре выбирает дату и время вводного урока







# Задача

---

- Надо сделать виджет
- Дизайн есть!
- Пользователь оставляет заявку
- В календаре выбирает дату и время вводного урока





# выберите время урока

Выберите время бесплатного вводного урока. Он длится до 50 минут  
в зависимости от вашего уровня владения английским

МОСКОВСКОЕ ВРЕМЯ < ПН, 3 ВТ, 4 СР, 5 ЧТ, 6 ПТ, 7 СБ, 8 ВС, 9 >

09:00							
10:00							
11:00							
12:00							
13:00			ВЫБРАТЬ				
14:00							
15:00							
16:00							
17:00							
18:00							
19:00							
20:00							
21:00							
22:00							



# Как будем делать тех. ревью?

---

- От базы/интеграций
- От арі фронтенда
- От проблемы бизнеса



# И от базы/интеграций и от арі

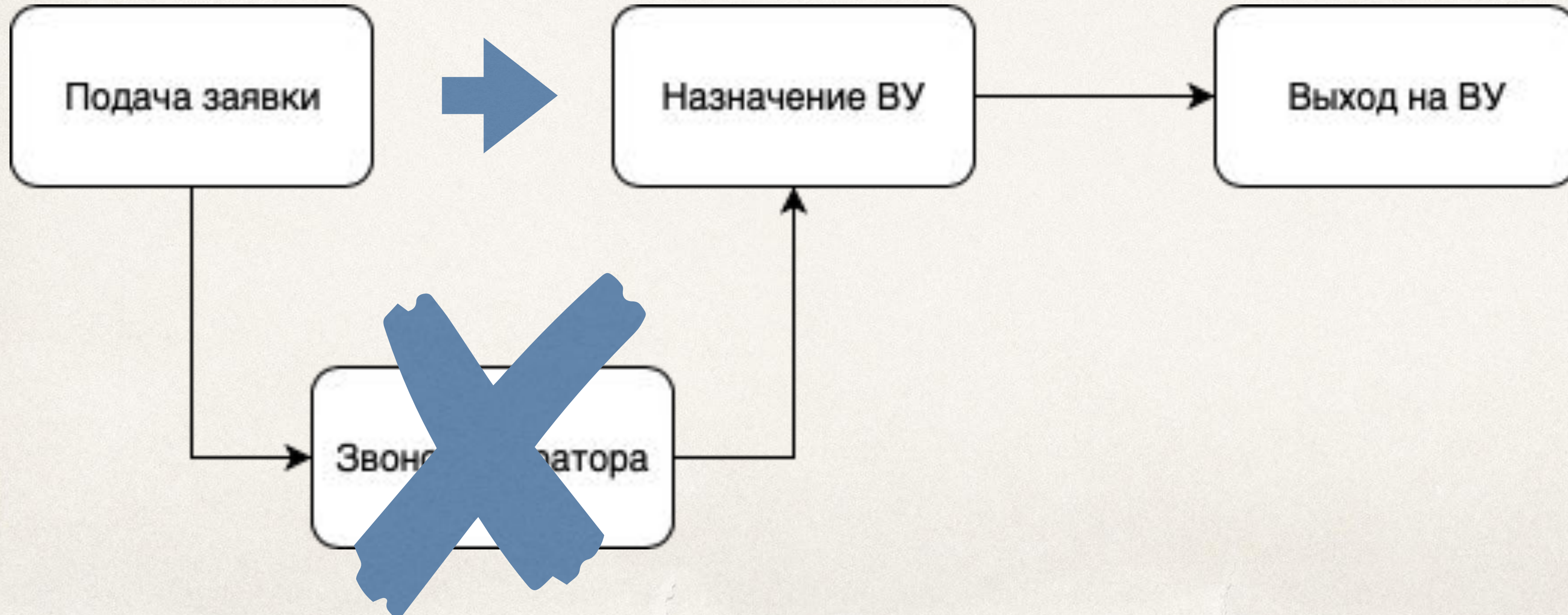
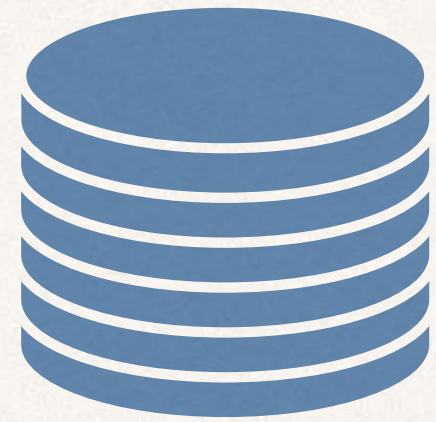
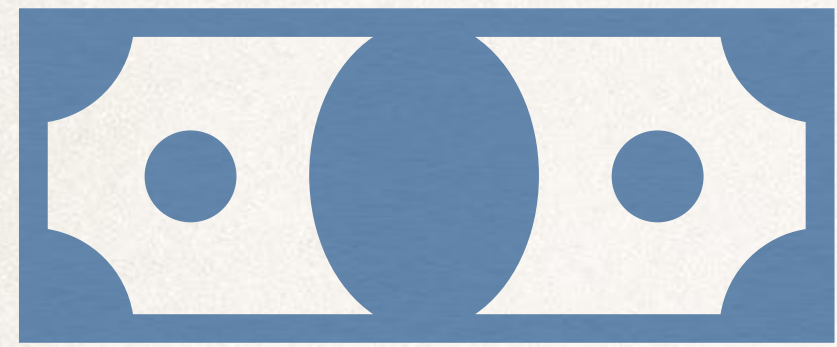
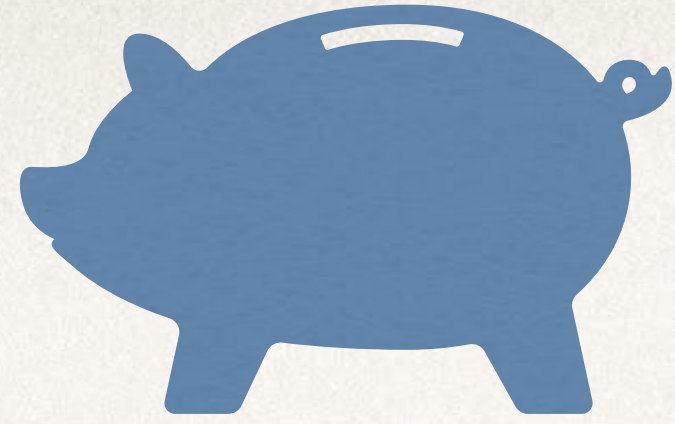
---

- Сервис букинга — получить, выбрать, отменить
- Фронтное арі — получить, выбрать, отменить
- База — таблица свзяка education\_service\_id, booking\_slot\_id
- Что там дальше, подумаем потом ;)
- Что думает по этому поводу бизнес?











# Чего хочет бизнес от разработки.

---

- Качественно — не терять сценарии и лучше понимать бизнес проблемы
- Быстро — быстро описывать сценарии в коде в отрыве от конкретных технологий (технологии не важны)
- Дешево — возможность проверять сценарии как можно раньше (оно вообще работает? может и разрабатывать не надо?)



# Что дальше?

---

- Опишем сервис
- Пробуем удовлетворить все 3 потребности
- Посмотрим со стороны гексагональной архитектуры



«Allow an application to equally be driven by **users, programs, automated test or batch scripts**, and to be developed and tested in isolation from its **eventual run-time devices and databases.**».

Позволяет взаимодействовать с приложением как **пользователю, так и программам, автоматическим тестам, скриптам пакетной обработки**. Также позволяет разрабатывать и тестировать приложение без каких-либо **дополнительных устройств или баз данных**.

— *Alistair Cockburn*



# Hexagonal architecture / Ports and adapters

**Пользователь**  
**Программы**  
**Тесты**  
**Скрипты**

**Приложение**

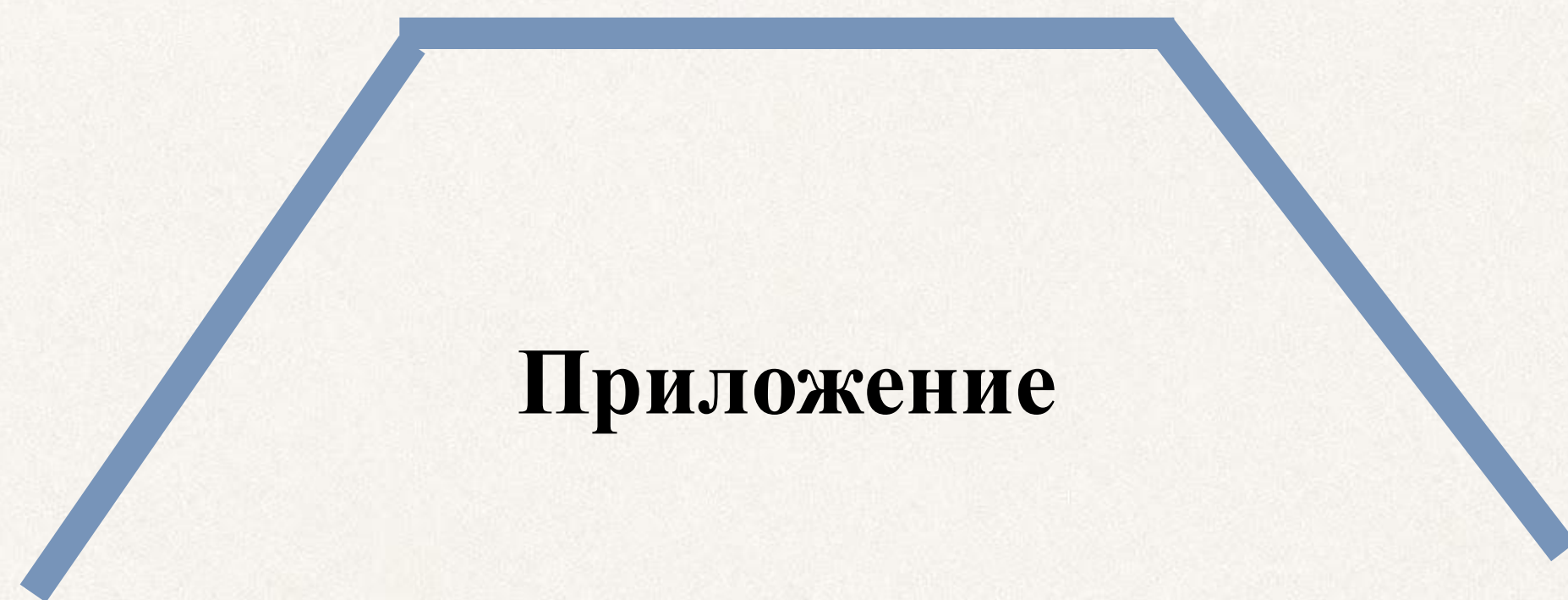
**Базы**  
**Внешние Апи**  
**Доп. устройства**





# Попробуем описать сервис — Application

---





```
class SelfTrialBookingService
{
    ...

    public function __construct(
        OperatorsServiceInterface $operatorsService,
        BookingServiceInterface $bookingService,
        SelfTrialRepositoryInterface $selfTrialRepository
    )
    {
        $this->operatorsService = $operatorsService;
        $this->bookingService = $bookingService;
        $this->selfTrialRepository = $selfTrialRepository;
    }
}
```

- Какие сервисы нужны?  
(что если их нет? или есть?)
- Что нам понадобится от них?
- Что нужно от репозитория?



```
interface OperatorsServiceInterface
```

```
{
```

```
    public function holdCall(int $educationServiceId, TimeInterval $interval, string $reason): void;
```

```
    public function disableCall(int $educationServiceId, string $reason): void;
```

```
    public function enableCall(int $educationServiceId, string $reason): void;
```

```
}
```

- Отложить звонок, сразу после заявки
- Отменить звонок вообще, если пользователь выбрал дату / время
- Назначить звонок, если пользователь передумал



```
interface BookingServiceInterface
```

```
{
```

```
    public function bookSlot(string $slotId, string $reason): void;
```

```
    public function cancelSlot(string $slotId, string $reason): void;
```

```
    public function isSlotAvailableForEducationService(string $slotId, int $educationServiceId):  
bool;
```

```
    public function getAvailableSlots();
```

```
}
```

- Выбрать дату / время
- Отменить дату / время
- Проверить подходит ли дата / время
- Получить список дат / времени



```
interface SelfTrialRepositoryInterface
{
    public function save(SelfTrial $selfTrial);

    public function getSelfTrialByEducationServiceId(int $educationServiceId): ?SelfTrial;
}
```

- Сохранить
- Получить



```
public function startSelfTrialProcess(int $educationServiceId): void
{
    $this->operatorsService->holdCall(
        $educationServiceId,
        new DateInterval(self::HOLD_CALL_INTERVAL),
        'self_trial'
    );

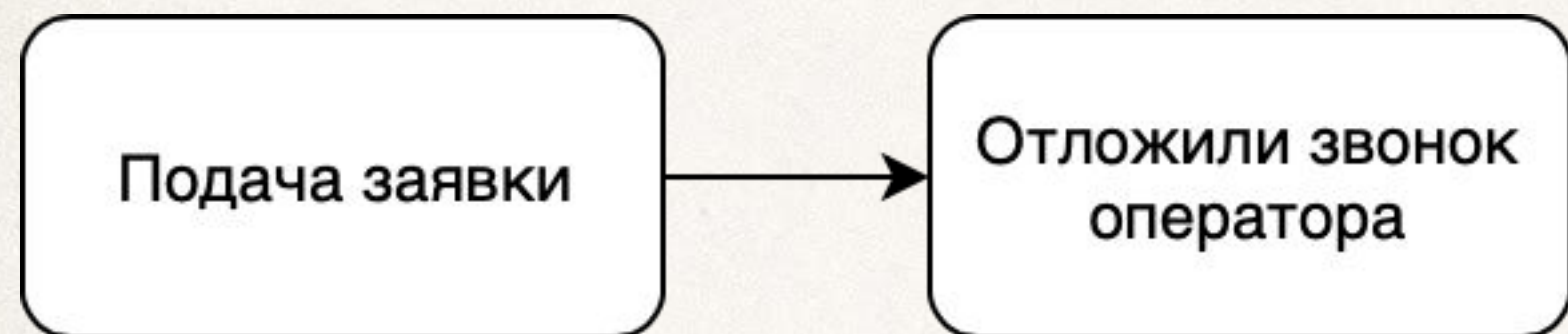
    $selfTrial = SelfTrial::start($educationServiceId);

    $this->selfTrialRepository->save($selfTrial);
}
```



```
public static function start(int $educationServiceId) : SelfTrial
{
    $instance = new self($educationServiceId);
    $instance->status = SelfTrialStatus::STARTED();
    return $instance;
}
```







```
public function bookSlot(int $educationServiceId, string $slotId, string $reason):
void
{
    $selfTrial = $this
        ->selfTrialRepository
        ->getSelfTrialByEducationServiceId($educationServiceId);

    //Валидация и DomainException

    $this->bookingService->bookSlot($slotId, $reason);

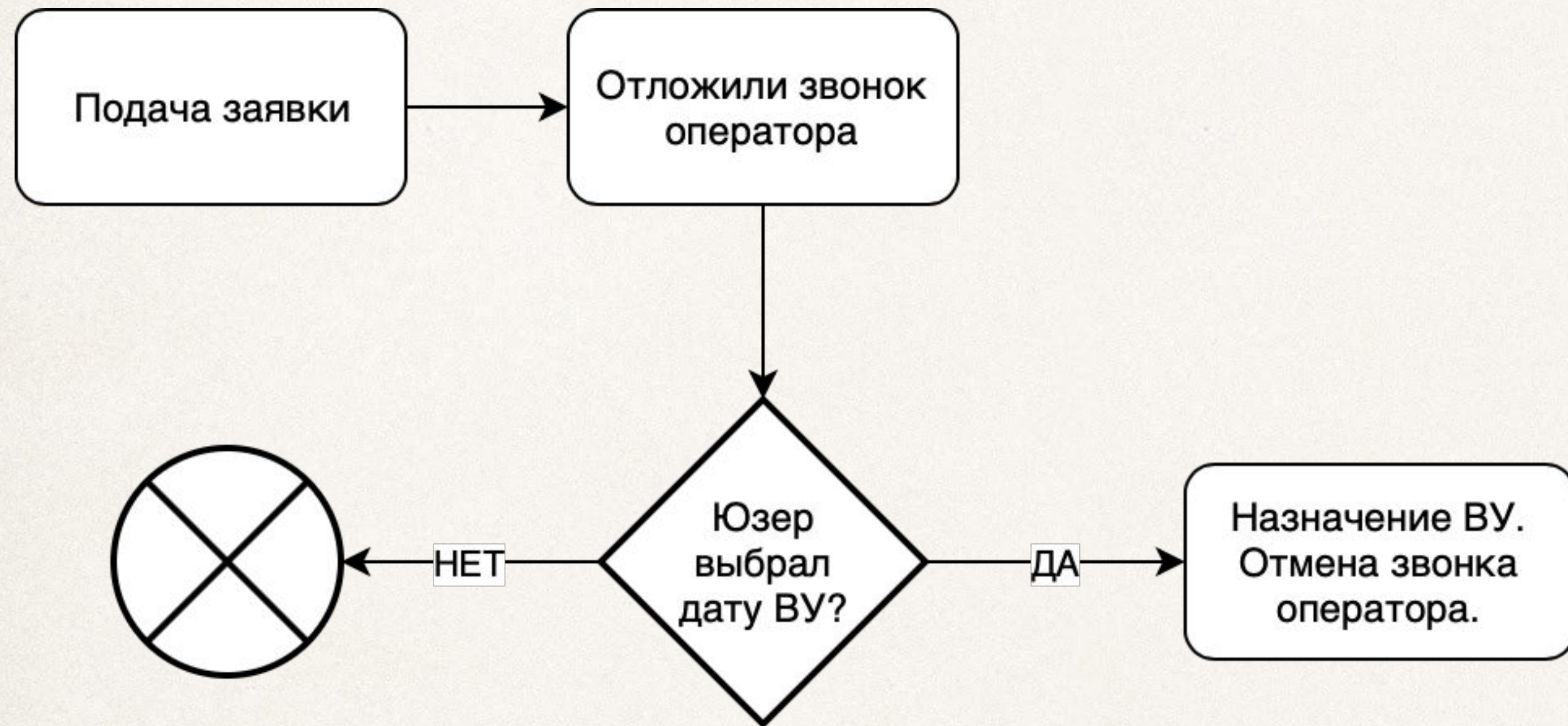
    $this->operatorsService->disableCall($educationServiceId, $reason);

    $selfTrial->bookSlot($slotId);
    $this->selfTrialRepository->save($selfTrial);
}
```



```
public function bookSlot(string $slotId)
{
    $this->slotId = $slotId;
    $this->status = SelfTrialStatus::BOOKED();
}
```







```
public function cancelSelfTrial(int $educationServiceId, string $reason): void
{
    $selfTrial = $this
        ->selfTrialRepository
        ->getSelfTrialByEducationServiceId($educationServiceId);

    //Валидация и DomainException

    $this->bookingService->cancelSlot($selfTrial->getSlotId(), $reason);

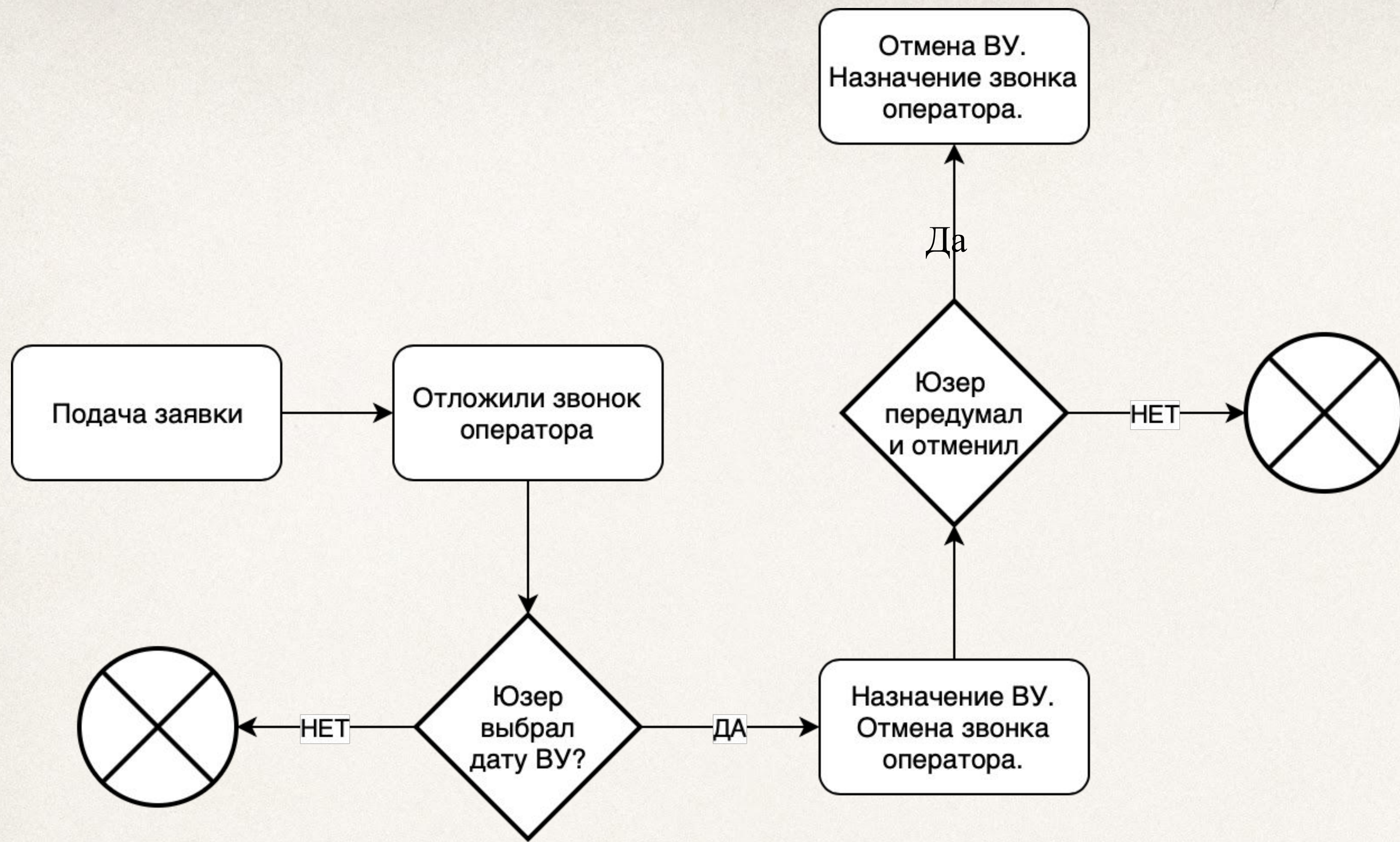
    $this->operatorsService->enableCall($educationServiceId, $reason);

    $selfTrial->cancel();
    $this->selfTrialRepository->save($selfTrial);
}
```



```
public function cancel()
{
    $this->status = SelfTrialStatus::CANCELLED();
}
```







# Application

---

- Качественно — не терять сценарии и лучше понимать бизнес проблемы
- Быстро — быстро описывать сценарии в коде в отрыве от конкретных технологий (технологии не важны)
- Дешево — возможность проверять сценарии как можно раньше (оно вообще работает? может и разрабатывать не надо?)



# Насколько это гибко?



## выберите время урока

Выберите время бесплатного вводного урока. Он длится до 50 минут  
в зависимости от вашего уровня владения английским

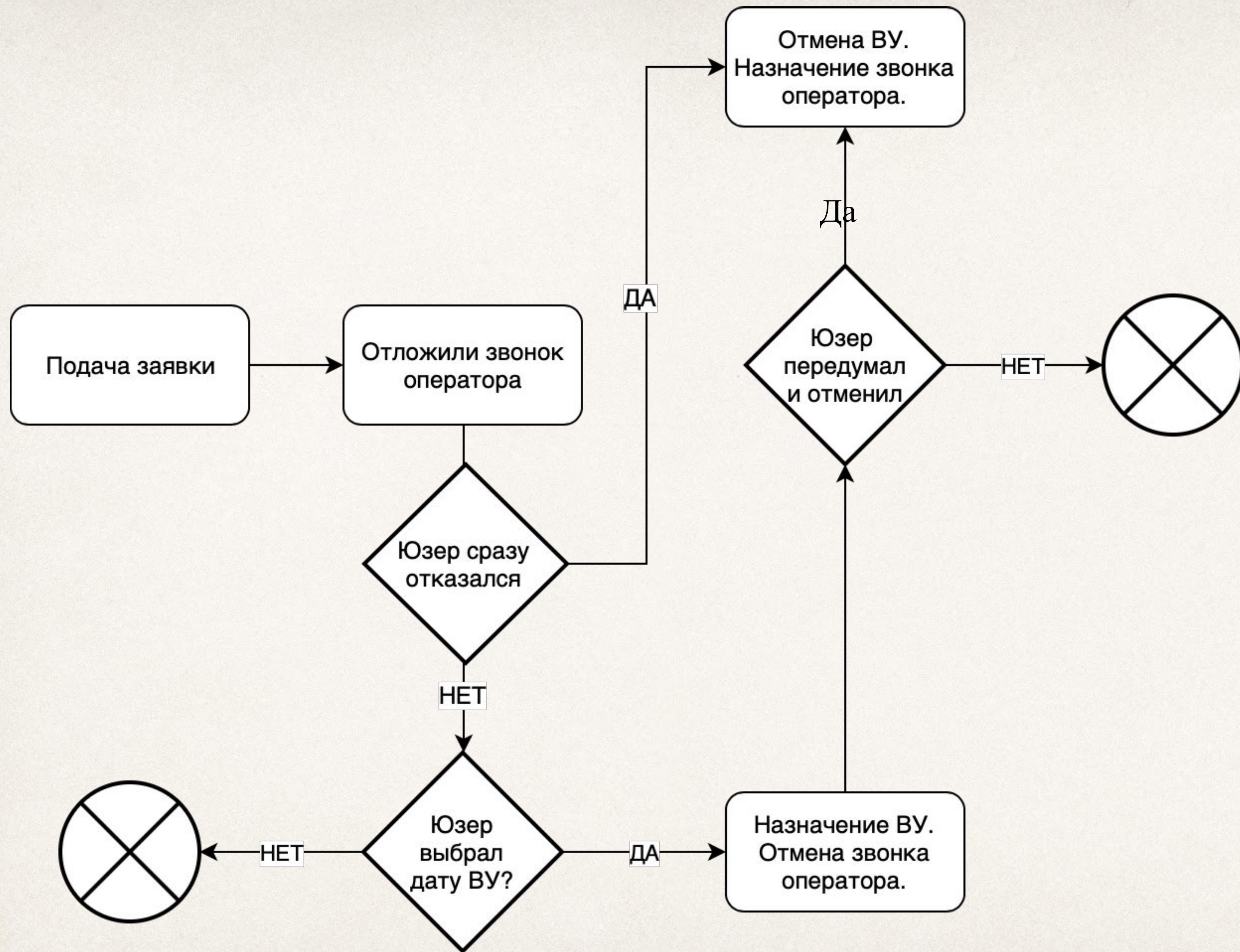
МОСКОВСКОЕ ВРЕМЯ < ПН, 3 ВТ, 4 СР, 5 ЧТ, 6 ПТ, 7 СБ, 8 ВС, 9 >

09:00							
10:00							
11:00							
12:00							
13:00			ВЫБРАТЬ				
14:00							
15:00							
16:00							
17:00							
18:00							
19:00							
20:00							
21:00							
22:00							



Вот тут будет кнопка отмены!!!







# Насколько это гибко?

---



## **МЫ ВАМ ПОЗВОНИМ**

Вы выбрали урок 22 сентября в 14:00. Наш менеджер свяжется с вами в течение двух часов, чтобы уточнить детали урока.

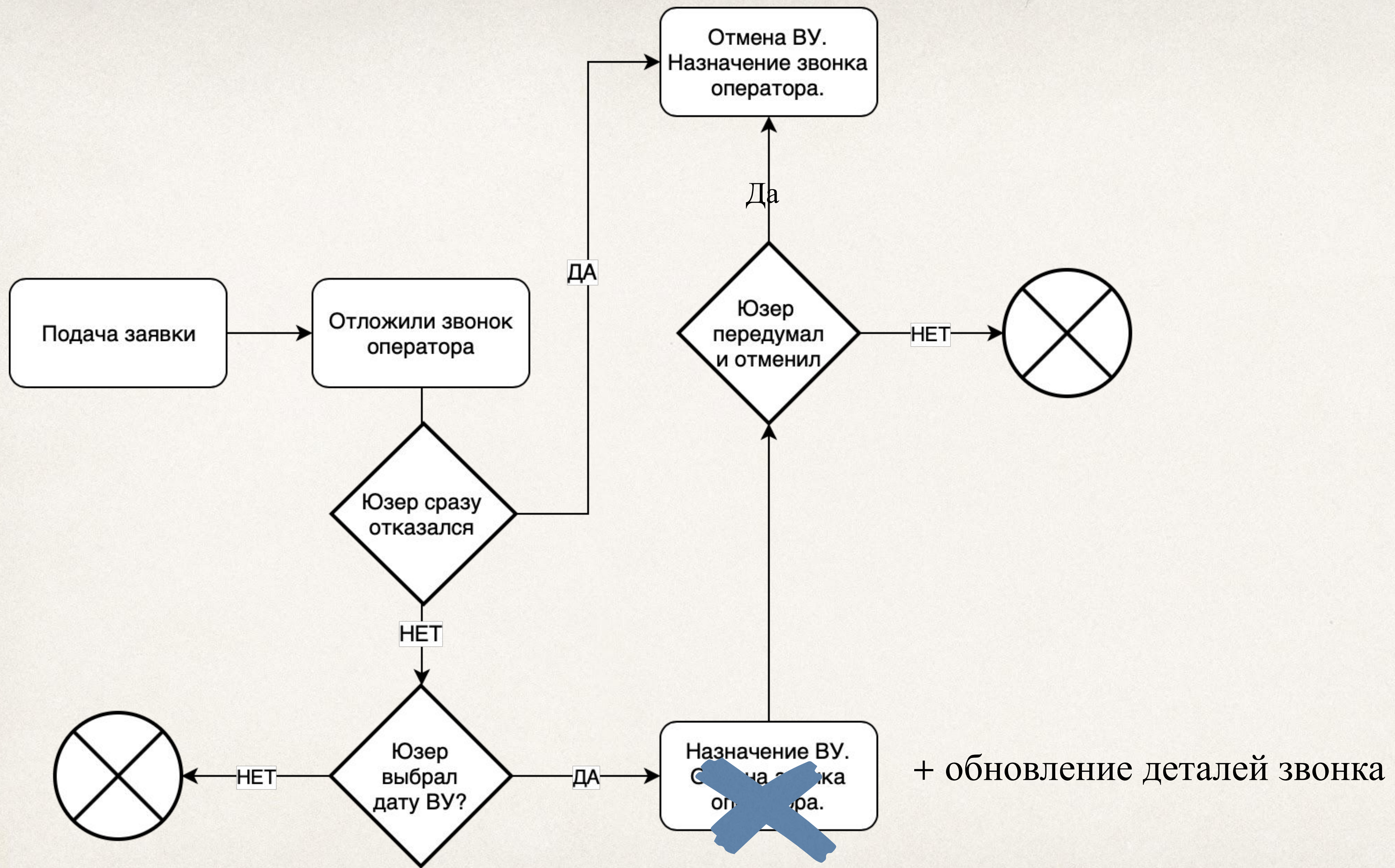
Добавить в календарь

✕ Отменить запись на урок



```
public function bookSlot(int $educationServiceId, string $slotId, string $reason):  
void  
{  
    $selfTrial = $this  
        ->selfTrialRepository  
        ->getSelfTrialByEducationServiceId($educationServiceId);  
  
    //Валидация и DomainException  
  
    $this->bookingService->bookSlot($slotId, $reason);  
  
$this->operatorsService->disableCall($educationServiceId, $reason);  
    $this->operatorsService->enableCall($educationServiceId, $reason);  
  
    $selfTrial->bookSlot($slotId);  
    $this->selfTrialRepository->save($selfTrial);  
}
```







# При чем тут DDD?

---

- Ubiquitous Language (единый язык)
  - Можно показать заказчику
  - Если не поймет, покрыть BDD тестом
- Bounded Context (контекст предметной области)



# Hexagonal architecture / Ports and adapters

**Пользователь**  
**Программы**  
**Тесты**  
**Скрипты**

**Приложение**

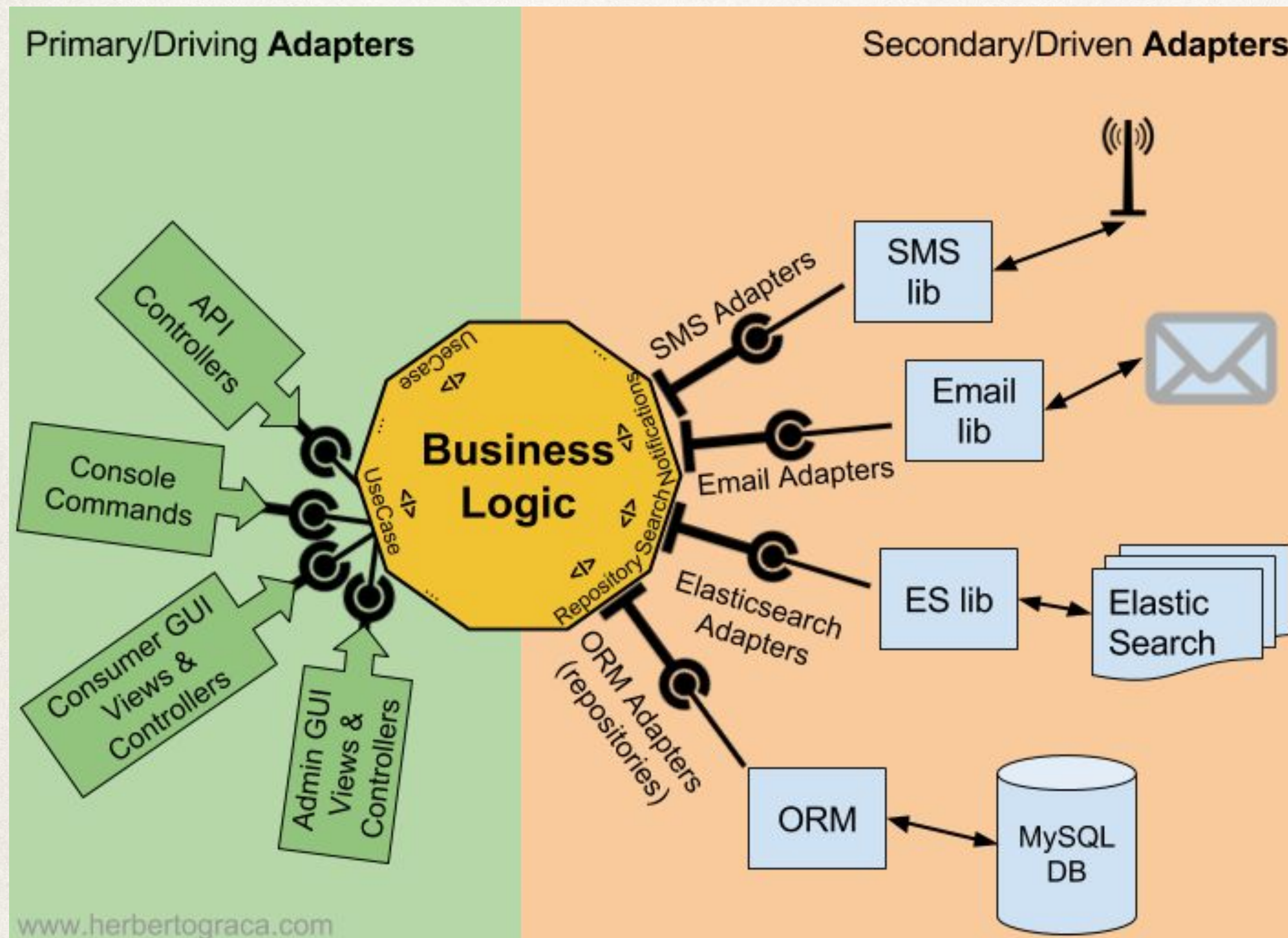
**Базы**  
**Внешние Апи**  
**Доп. устройства**

```
▼ AwesomeApplication
  ▼ src
    ▶ Application
    ▶ Controller
    ▶ Infrastructure
```



# Более красивая картинка ;)

- <https://herbertograca.com/2017/09/14/ports-adapters-architecture/>





# Зачем

---

- Позволяет концентрироваться на домене
- Выделение бизнес логики
- Тесты
- Заменяемые элементы
- Дисциплина (именование папок, куда что положить)



# Заметки на полях

---

- Что с атомарностью (`@synchronized`)?
- Почему не ивенты из модели?
- Где эксепшены?



# Спасибо.

---

- <https://www.infoq.com/minibooks/domain-driven-design-quickly/>
- <http://www.ntcoding.co.uk/workshops/strategic-ddd-practices>
- <http://www.ouarzy.com/2016/07/25/micro-service-and-bounded-context-clarification/>
- [http://www.dossier-andreas.net/software\\_architecture/ports\\_and\\_adapters.html](http://www.dossier-andreas.net/software_architecture/ports_and_adapters.html)
- Ссылка на Github с примером ;)