ozon fintech

400 релизов в неделю — реально.

Просто внедряй эти инженерные практики

Егор Стручин
тимлид группы тестирования микросервисов

Инженерные практики

Автоматизация не ограничивается автотестами

Поговорим про:

ozon fintech

Практики и решения

- Ускорение тестов
- Стабилизация тестов
- Борьба с избыточностью запусков
- Борьба с дублируемостью тестов
- Наглядная оценка покрытия
- Тестовая аналитика
- Сервис генерации данных
- Система моков внешних сервисов
- Балансировщик тестов
- Система трассировок
- Тест-дизайн юнит-тестов
- Система хранения покрытия

Поговорим про:

ozon fintech

Практики и решения

- Ускорение тестов
- Стабилизация тестов
- Борьба с избыточностью запусков
- Борьба с дублируемостью тестов
- Наглядная оценка покрытия
- Тестовая аналитика
- Сервис генерации данных
- Система моков внешних сервисов
- Балансировщик тестов
- Система трассировок
- Тест-дизайн юнит-тестов
- Система хранения покрытия

Поговорим про:

ozon fintech

Практики и решения

- Ускорение тестов
- Стабилизация тестов
- Борьба с избыточностью запусков
- Борьба с дублируемостью тестов
- Наглядная оценка покрытия
- Тестовая аналитика
- Сервис генерации данных
- Система моков внешних сервисов
- Балансировщик тестов
- Система трассировок
- Тест-дизайн юнит-тестов
- Система хранения покрытия

Поговорим про:

ozon fintech

Практики и решения

- Ускорение тестов
- Стабилизация тестов
- **Борьба с избыточностью запусков**
- Борьба с дублируемостью тестов
- Наглядная оценка покрытия
- Тестовая аналитика
- Сервис генерации данных
- Система моков внешних сервисов
- **Балансировщик тестов**
- **Система трассировок**
- Тест-дизайн юнит-тестов
- Система хранения покрытия

Поговорим про:

ozon fintech

Практики и решения

- Ускорение тестов
- Стабилизация тестов
- Борьба с избыточностью запусков
- **Борьба с дублируемостью тестов**
- Наглядная оценка покрытия
- Тестовая аналитика
- Сервис генерации данных
- Система моков внешних сервисов
- Балансировщик тестов
- Система трассировок
- **Тест-дизайн юнит-тестов**
- **Система хранения покрытия**

Поговорим про:

ozon fintech

Практики и решения

- Ускорение тестов
- Стабилизация тестов
- Борьба с избыточностью запусков
- Борьба с дублируемостью тестов
- Наглядная оценка покрытия
- Тестовая аналитика
- Сервис генерации данных
- Система моков внешних сервисов
- Балансировщик тестов
- Система трассировок
- Тест-дизайн юнит-тестов
- Система хранения покрытия

Поговорим про:

ozon fintech

Практики и решения

- Ускорение тестов
- Стабилизация тестов
- Борьба с избыточностью запусков
- Борьба с дублируемостью тестов
- Наглядная оценка покрытия
- Тестовая аналитика
- Сервис генерации данных
- Система моков внешних сервисов
- Балансировщик тестов
- Система трассировок
- Тест-дизайн юнит-тестов
- Система хранения покрытия

Поговорим про:

ozon fintech

Практики и решения

- Ускорение тестов
- Стабилизация тестов
- Борьба с избыточностью запусков
- Борьба с дублируемостью тестов
- Наглядная оценка покрытия
- Тестовая аналитика
- Сервис генерации данных
- Система моков внешних сервисов
- Балансировщик тестов
- Система трассировок
- Тест-дизайн юнит-тестов
- Система хранения покрытия

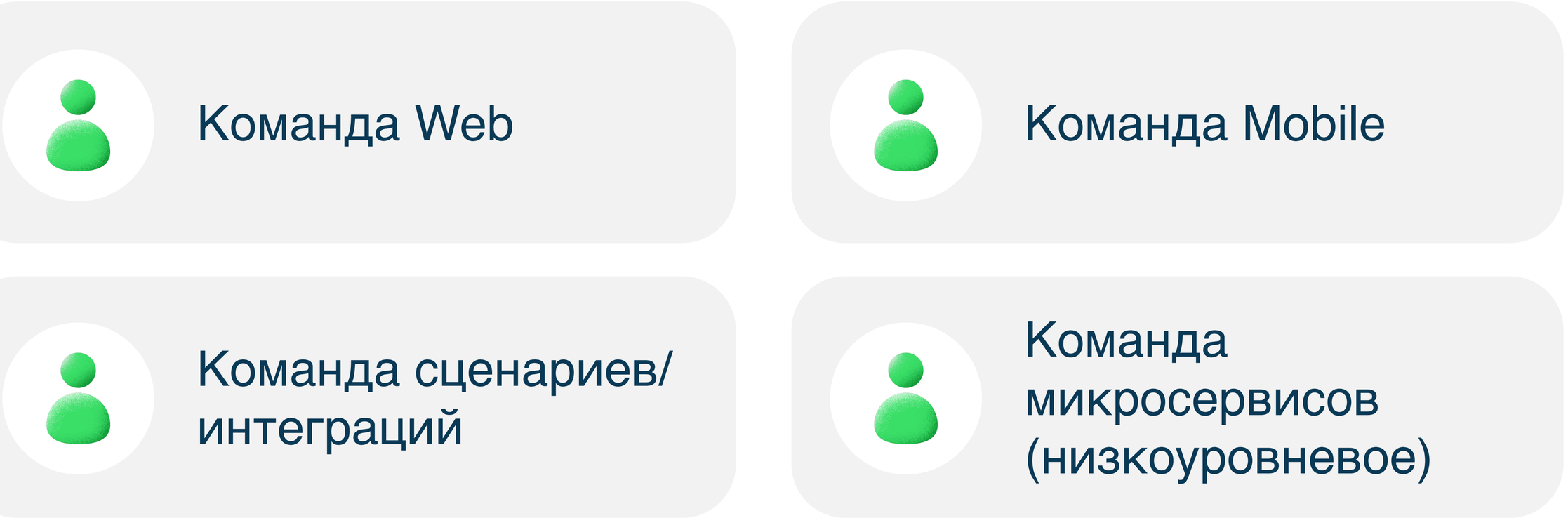
Применяем это — улучшаем процесс разработки

Структура команды

ozon fintech

Продукт: Fintech (много backend, API, web/mobile)

Тестирование:

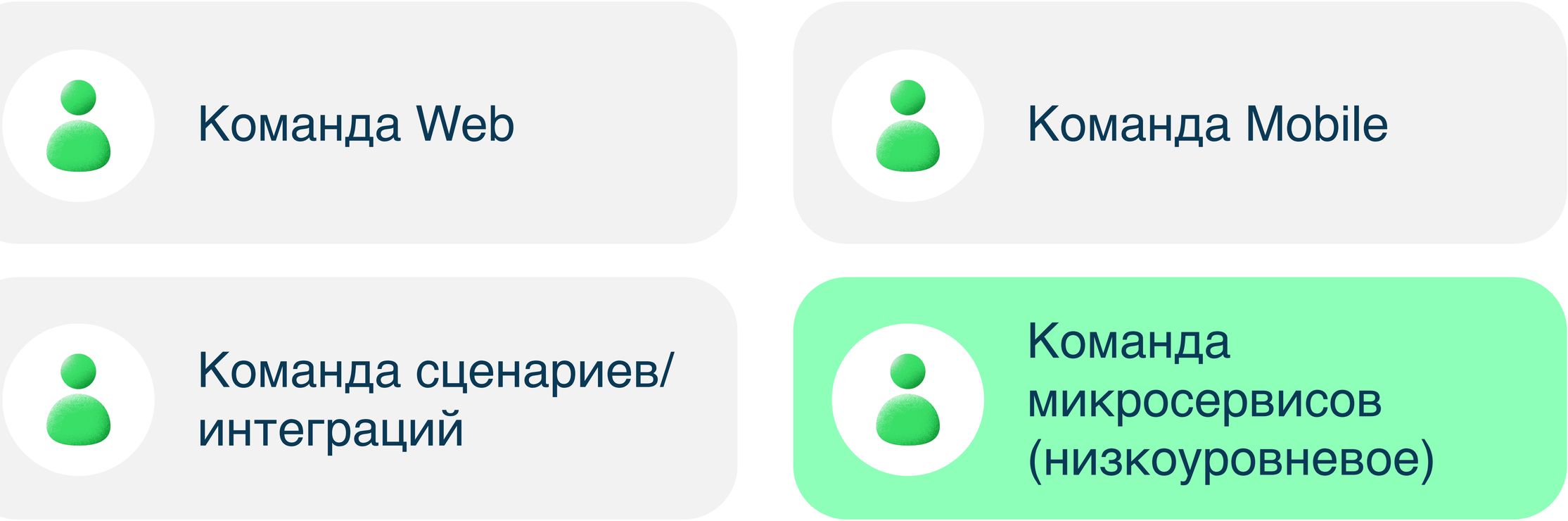


Структура команды

ozon fintech

Продукт: Fintech (много backend, API, web/mobile)

Тестирование:

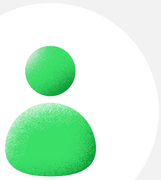


Структура команды

ozon fintech

Продукт: Fintech (много backend, API, web/mobile)

Тестирование:



Команда микросервисов (низкоуровневое)

Задачи:

- Юнит-тесты с разработчиками

Структура команды

ozon fintech

Продукт: Fintech (много backend, API, web/mobile)

Тестирование:

Команда микросервисов (низкоуровневое)

Задачи:

- Юнит-тесты с разработчиками

Экспертиза:

- Устройство сервисов и деливери (ci/cd)
- Разработка, создание своей инфраструктуры и утилит

Структура команды

ozon fintech

Продукт: Fintech (много backend, API, web/mobile)

Тестирование:

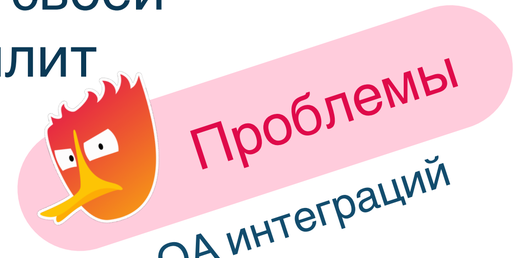
Команда микросервисов (низкоуровневое)

Задачи:

- Юнит-тесты с разработчиками

Экспертиза:

- Устройство сервисов и деливери (ci/cd)
- Разработка, создание своей инфраструктуры и утилит



Команда, задачи и проблематика…

ozon fintech

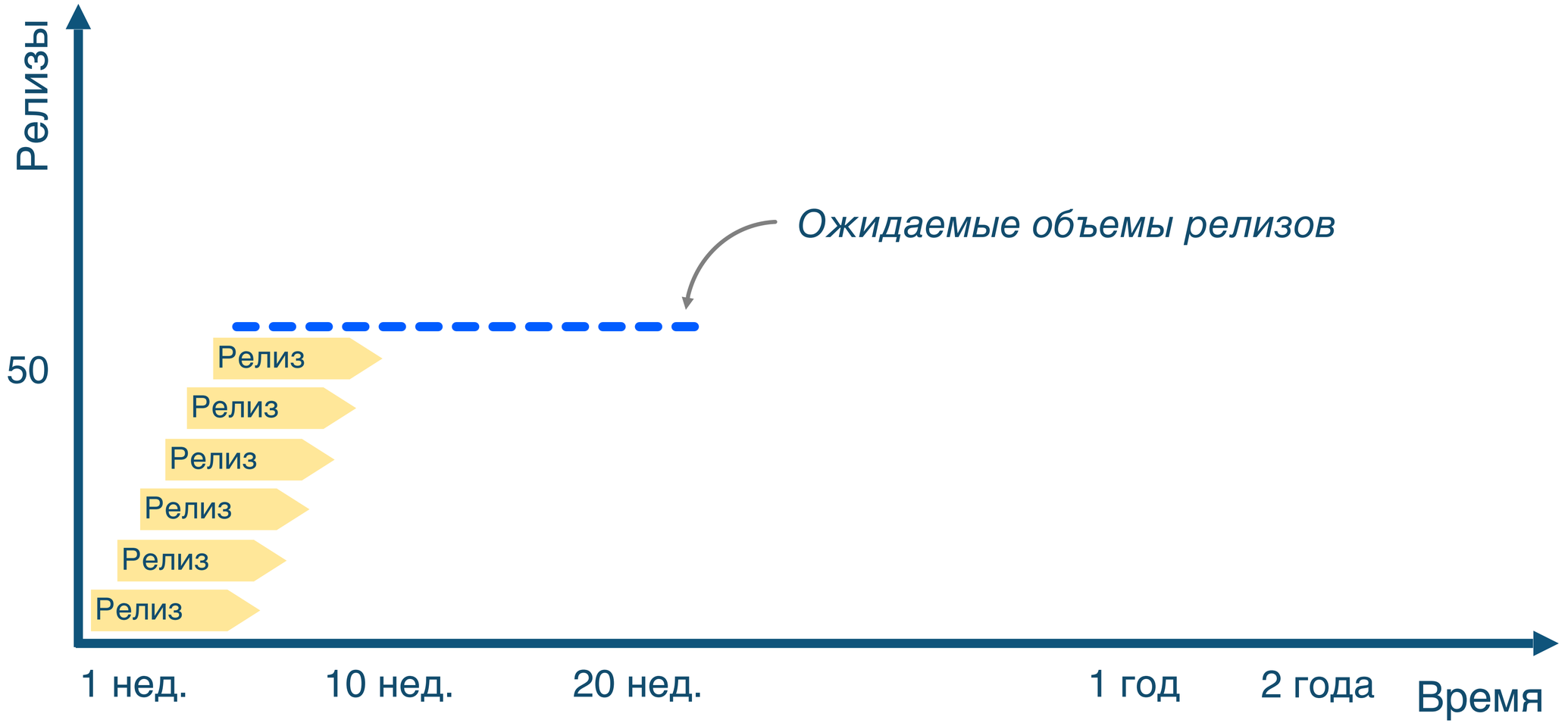
Backend, API

Проблемы, решения
и устоявшиеся практики
на конкретных примерах

Процесс тестирования релизов

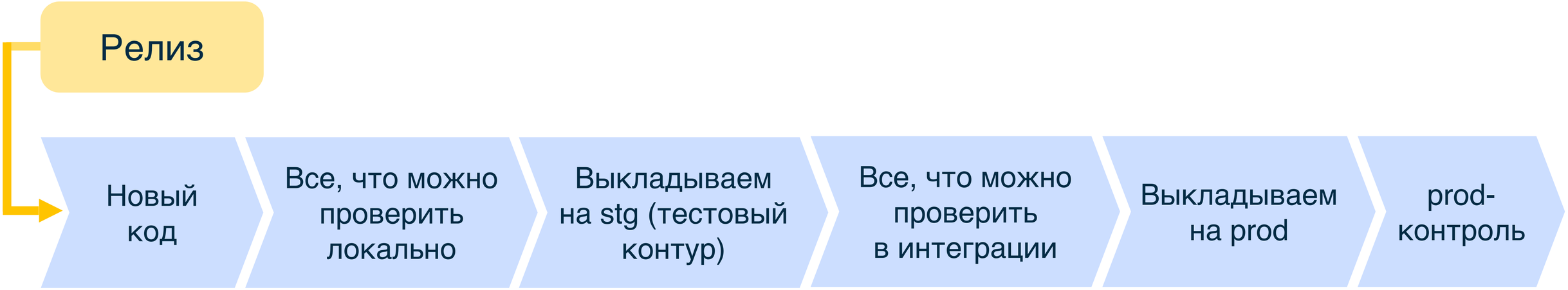
ozon fintech

Вводные: 40 сервисов, каждый обновляется 2 раза в неделю — примерно 50 релизов



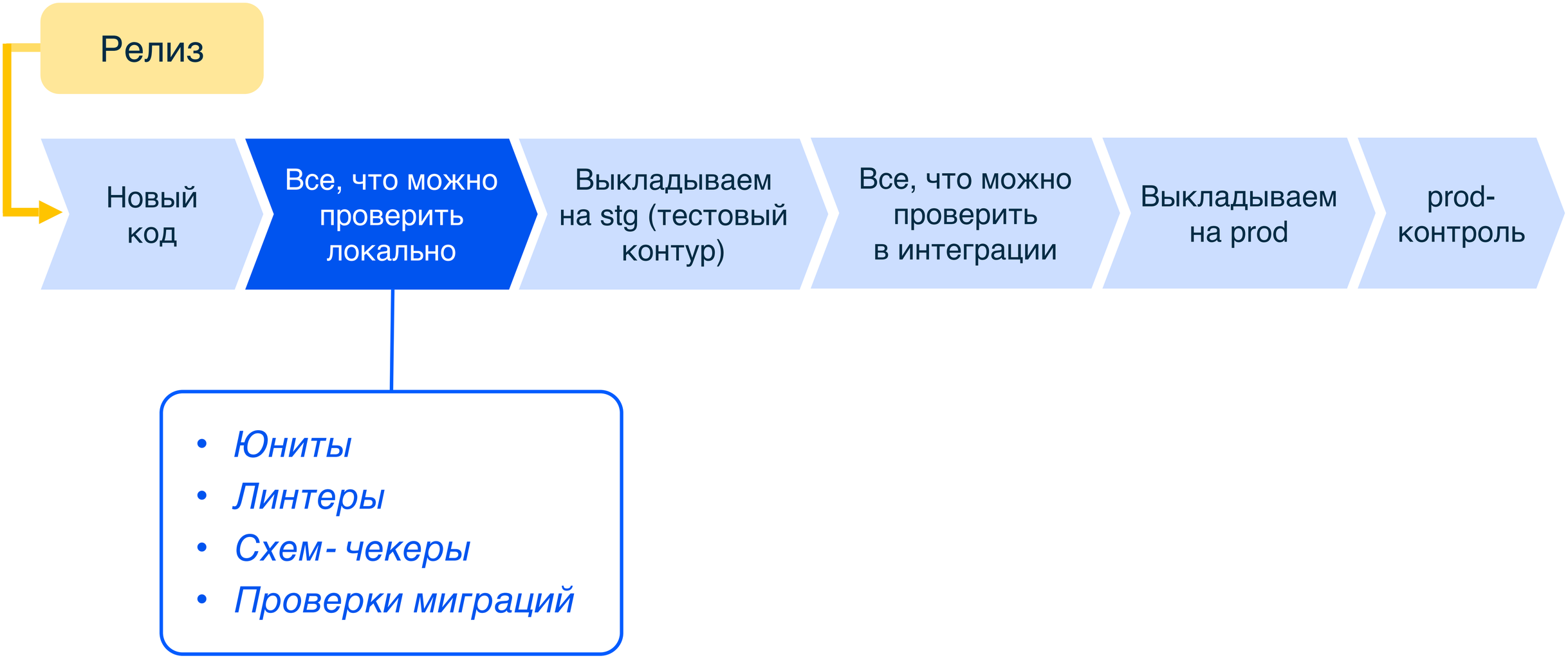
Процесс тестирования релизов

ozon fintech



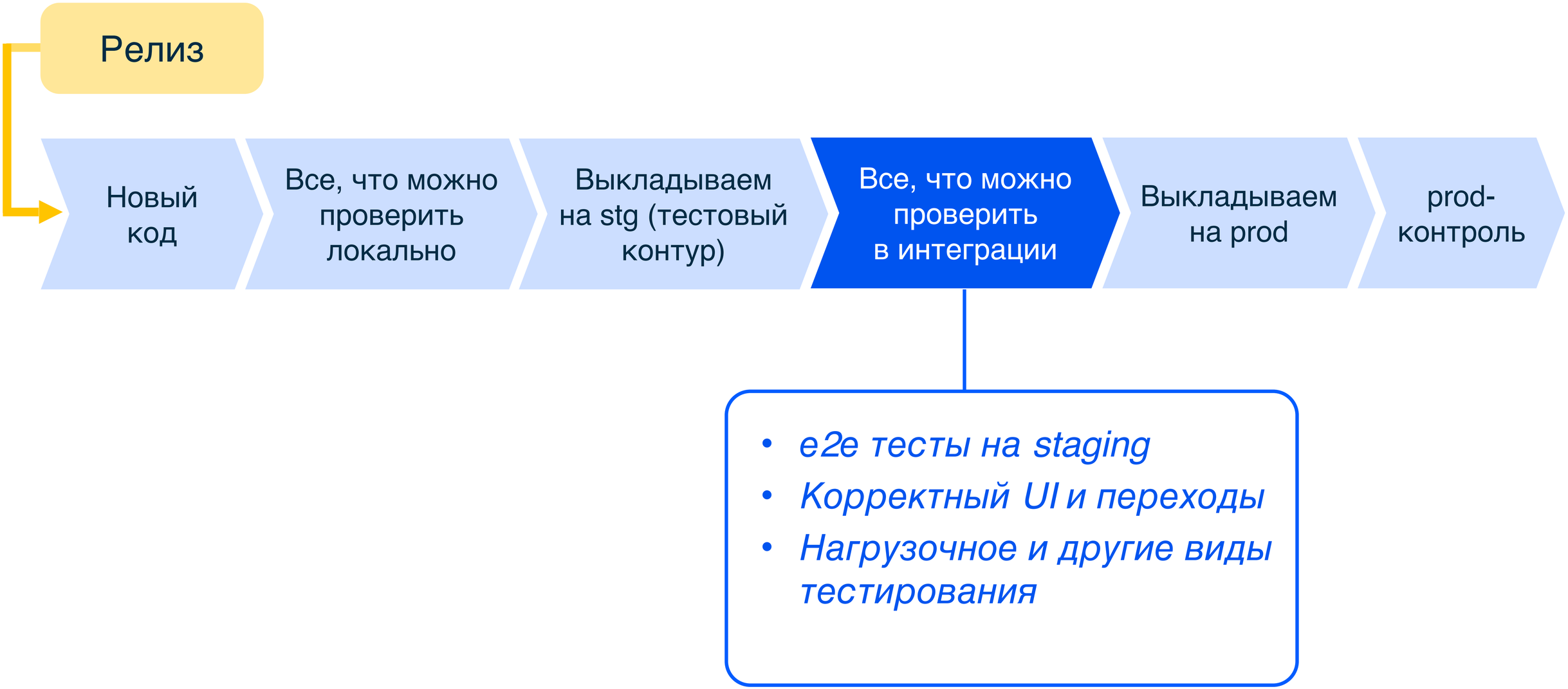
Процесс тестирования релизов

ozon fintech



Процесс тестирования релизов

ozon fintech



Процесс тестирования релизов

ozon fintech

Релиз

Новый код → Все, что можно проверить локально → Выкладываем на stg (тестовый контур) → Все, что можно проверить в интеграции → Выкладываем на prod → prod-контроль

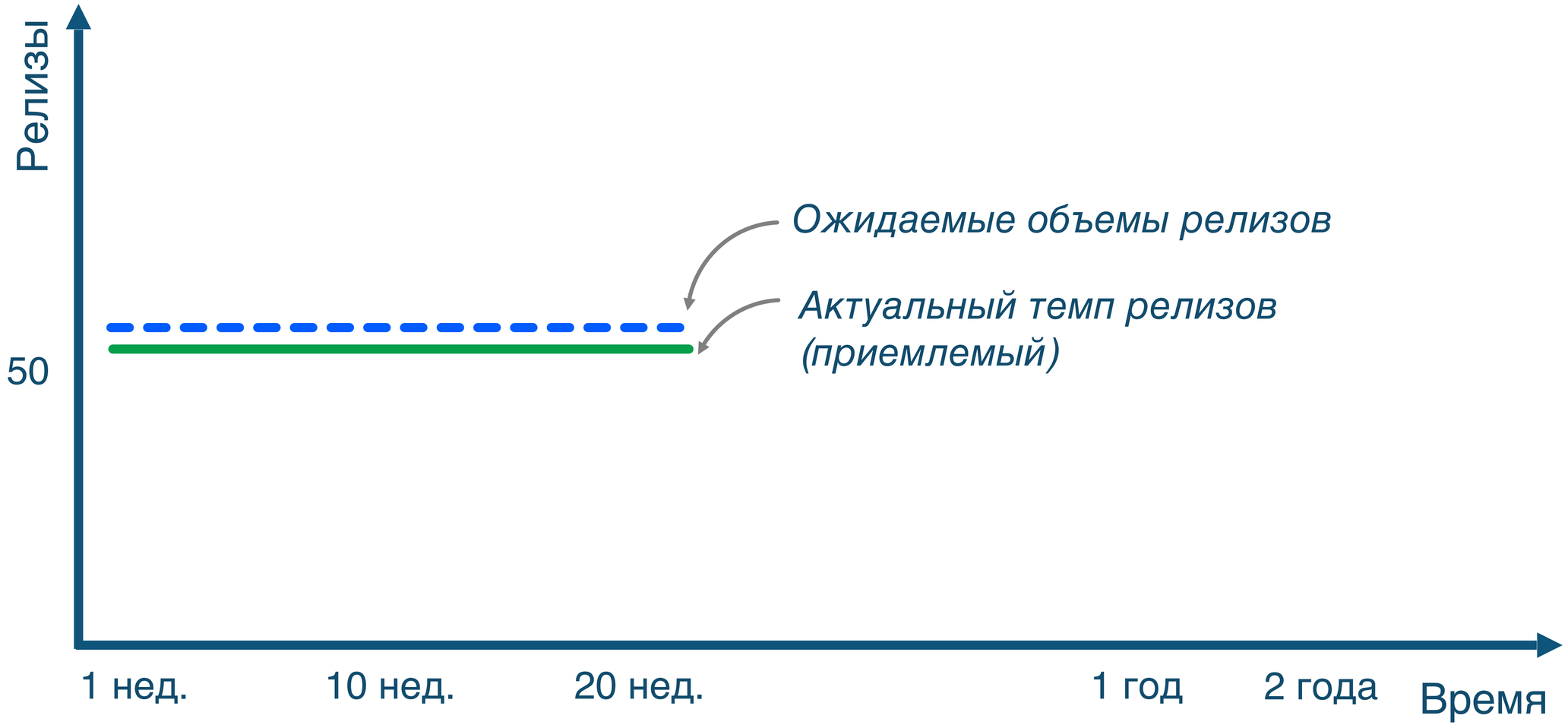
ПЕТЬКА, ПРИБОРЫ!

- *Канарейка*
- *Мониторинг*
- *Система логов*
- *Железная леди*

Процесс тестирования релизов

ozon fintech

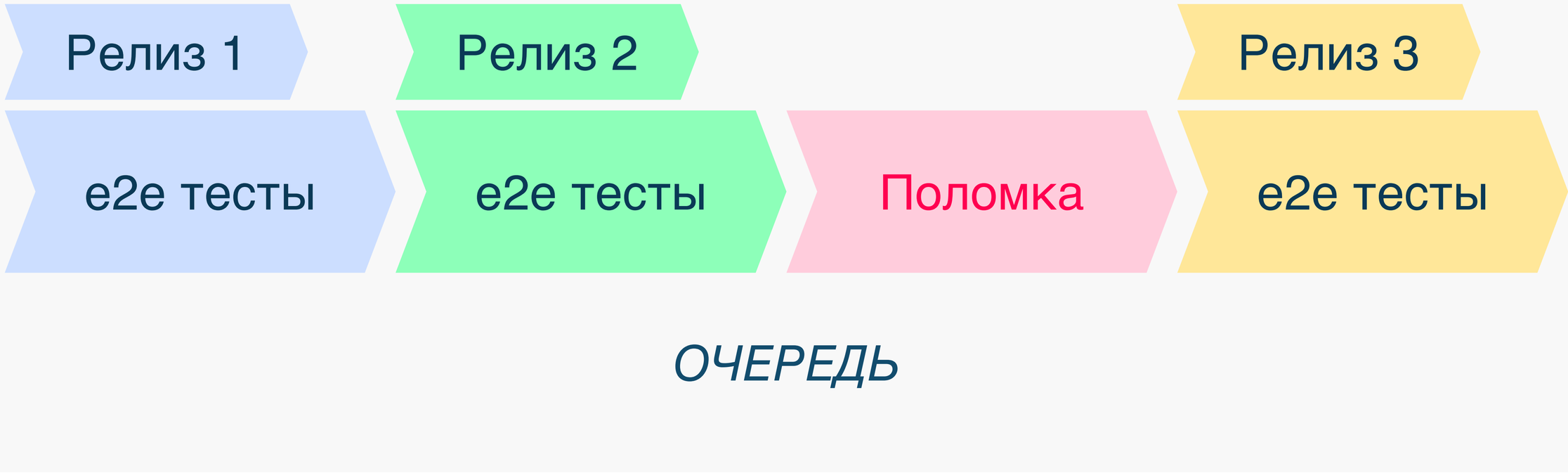
- За полгода — 40 сервисов
- 400 сценариев автотестов на staging
- 2 раза в неделю обновить каждый сервис — примерно 50 релизов



Первые проблемы

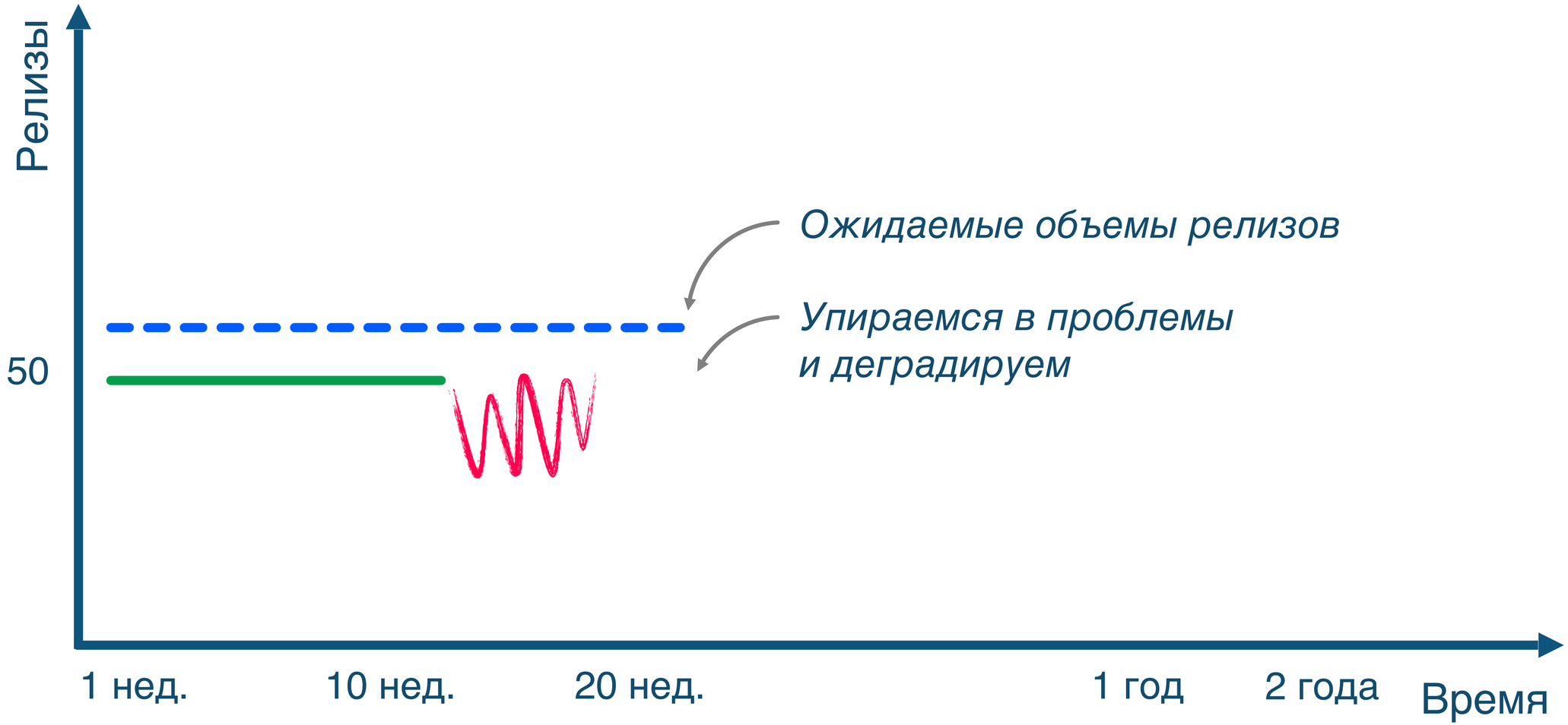
ozon fintech

Регулярные запуски, 400 автотестов — нестабильный STAGING, поломки в очереди



Первые проблемы

ozon fintech



Варианты решения

ozon fintech

Накинуть ресурсов – **отсрочка проблемы**

Попробуем **ускорять** тесты и **снижать нагрузку** от них

Анализ тестов

ozon fintech

Почему тесты долгие?

В Fintech основная масса тестов выглядит так:

A. Подготовка

B. Активность

C. Проверить результат

Анализ тестов

ozon fintech

Почему тесты долгие?

В Fintech основная масса тестов выглядит так:

A. Подготовка

Регистрация пользователя *(10 сек)*, обработать перс. данные *(10 сек)*, завести ему карту *(10 сек)*, накинуть денег *(5 сек)*

B. Активность

Потратить деньги, сделать перевод или настроить лимиты трат *(5-10 сек)*

C. Проверить результат

Где деньги? *(5 сек)*

Анализ тестов

ozon fintech

Почему тесты долгие?

В Fintech основная масса тестов выглядит так:

A. Подготовка

Регистрация пользователя *(10 сек)*, обработать перс. данные *(10 сек)*, завести ему карту *(10 сек)*, накинуть денег *(5 сек)*

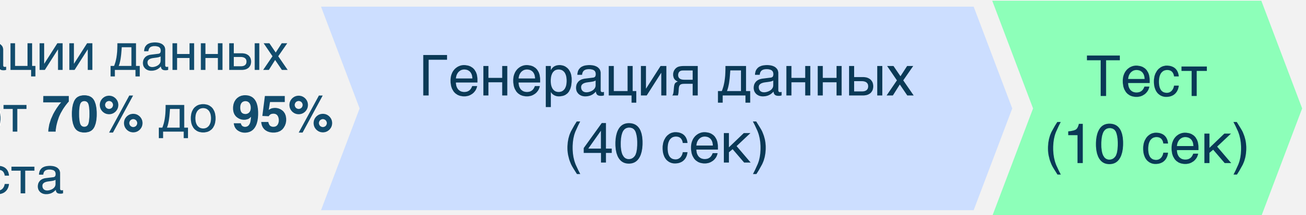
B. Активность

Потратить деньги, сделать перевод или настроить лимиты трат *(5-10 сек)*

C. Проверить результат

Где деньги? *(5 сек)*

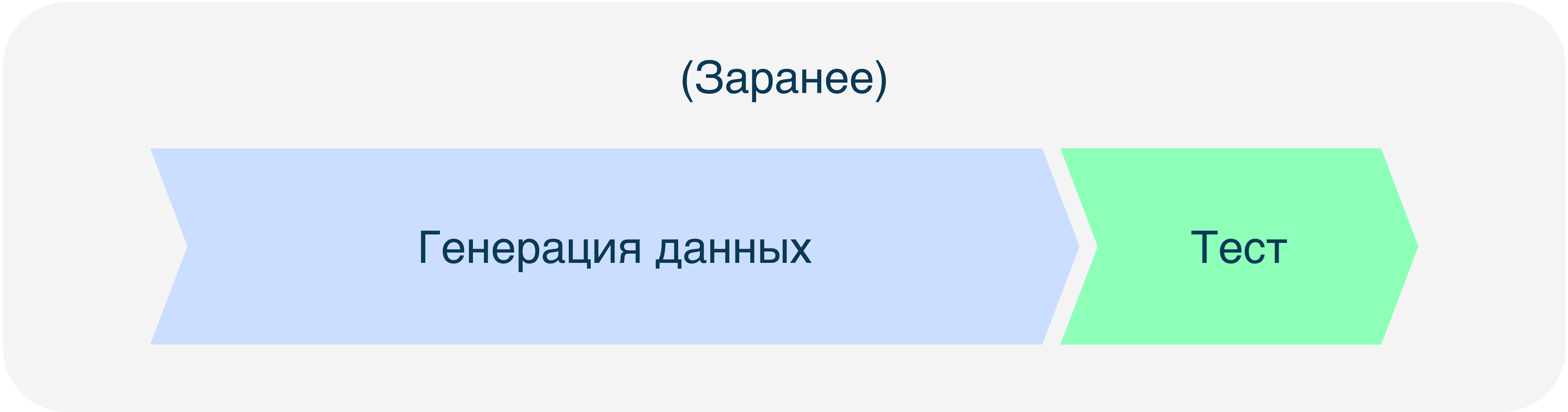
Этап генерации данных составлял от **70%** до **95%** времени теста



И самое популярное решение, которое напрашивалось…

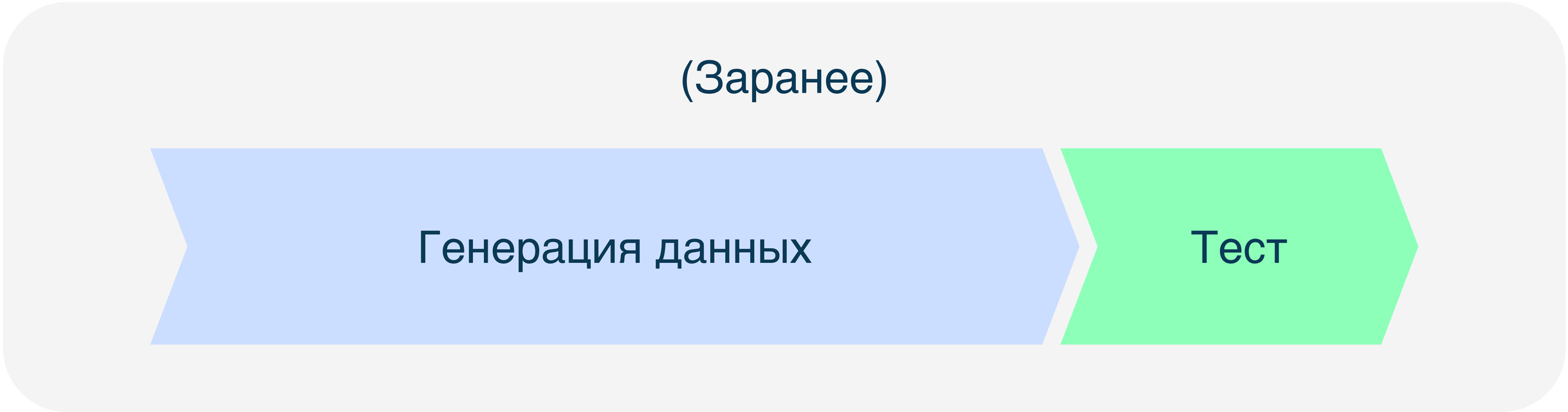
Подготовить тестовые данные

ozon fintech



Подготовить тестовые данные

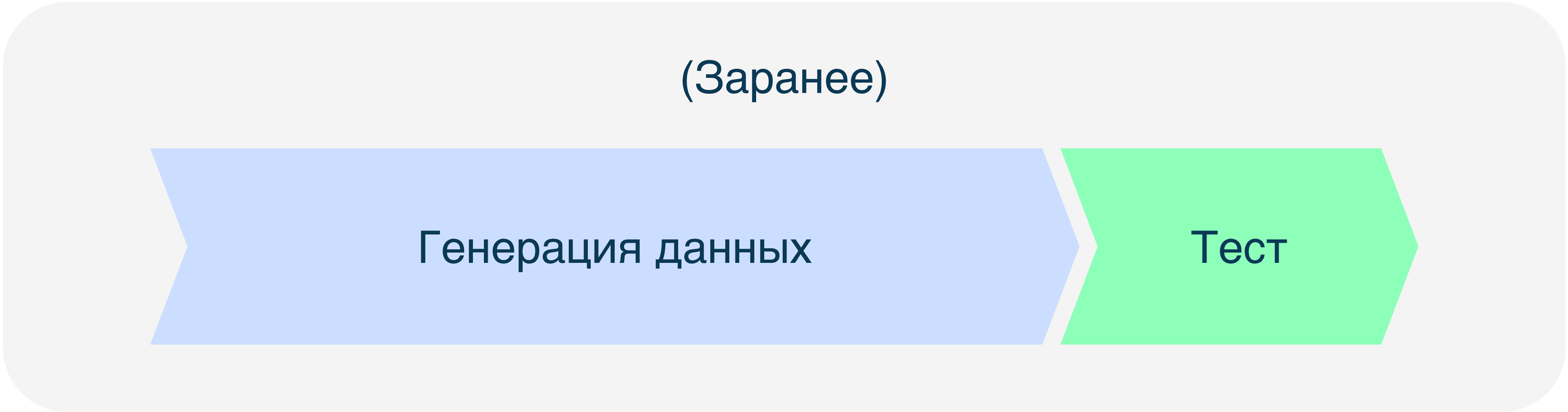
ozon fintech



Сколько?

Подготовить тестовые данные

ozon fintech

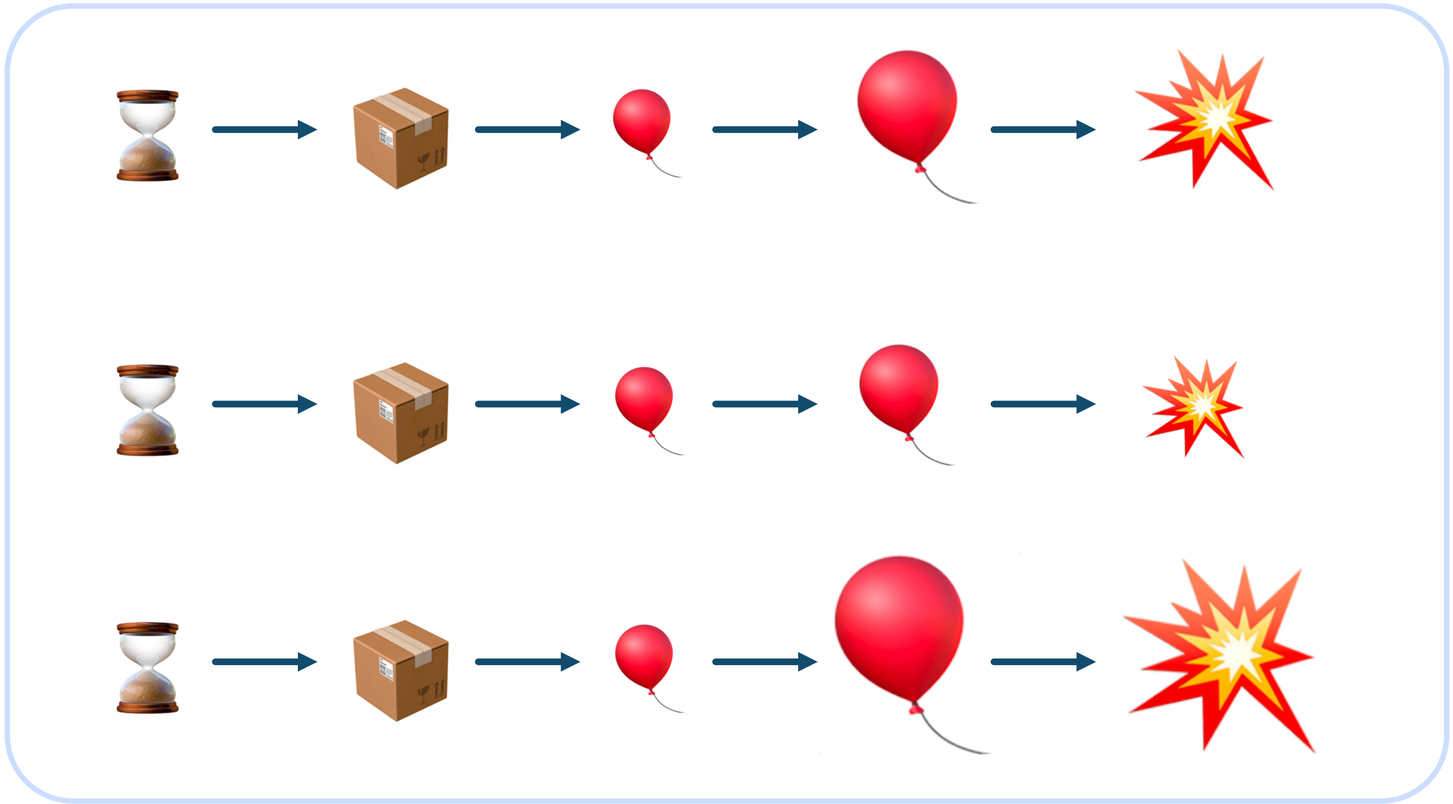


Сколько и как?

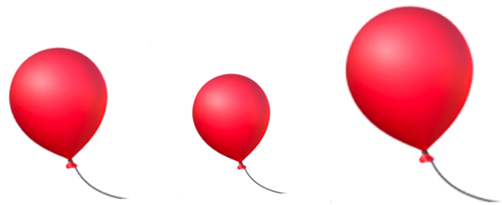
Сколько?

ozon fintech

Запуск



Количество
на 1 запуск:



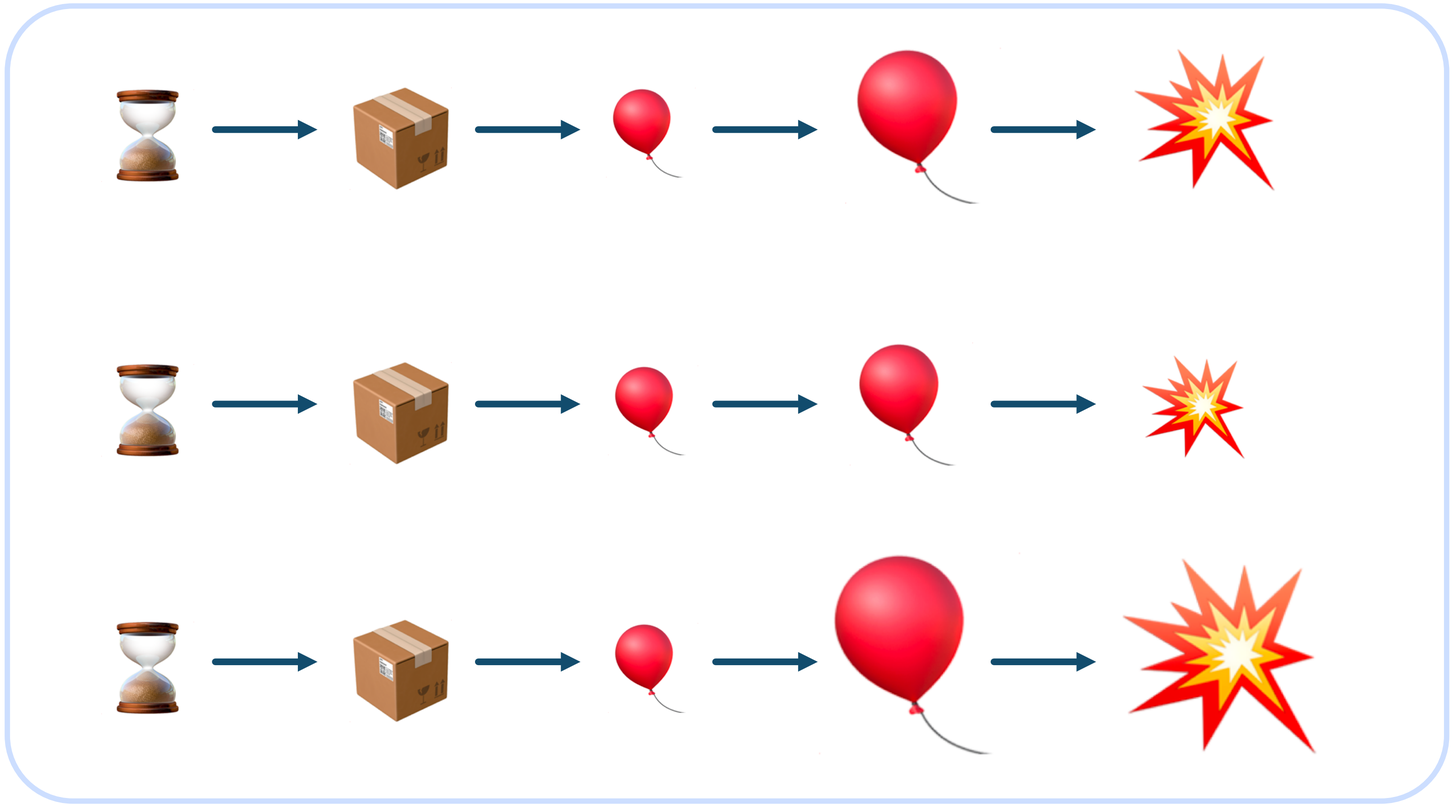
Запусков
за сутки: 10

x10

Сколько?

ozon fintech

Запуск



Количество
на 1 запуск:

Запусков
за сутки: 10

Подготовить: как?

ozon fintech

Минусы:

Вручную	В рамках рабочего дня, нужно следить на предмет неудачи при генерации
По расписанию (ночью)	Нужно следить на предмет неудачи при генерации. Неудача при генерации сильно повлияет на тестирование
Постоянно	Сложный проект. Нужно поддерживать инфраструктуру, нужны дежурства по проблемам генерации

Подготовить: как?

ozon fintech

Вручную

Минусы:

В рамках рабочего дня, нужно следить на предмет неудачи при генерации

Подготовить: как?

ozon fintech

По расписанию (ночью)

Минусы:

Нужно следить на предмет неудачи при генерации.
Неудача при генерации сильно повлияет на тестирование

Подготовить: как?

ozon fintech

Постоянно

Минусы:

Сложный проект. Нужно поддерживать инфраструктуру, нужны дежурства по проблемам генерации

Подготовить: как?

ozon fintech

Минусы:

Вручную	В рамках рабочего дня, нужно следить на предмет неудачи при генерации
По расписанию (ночью)	Нужно следить на предмет неудачи при генерации. Неудача при генерации сильно повлияет на тестирование
Постоянно	Сложный проект. Нужно поддерживать инфраструктуру, нужны дежурства по проблемам генерации

Постоянно генерировать 

ozon fintech

Сервис

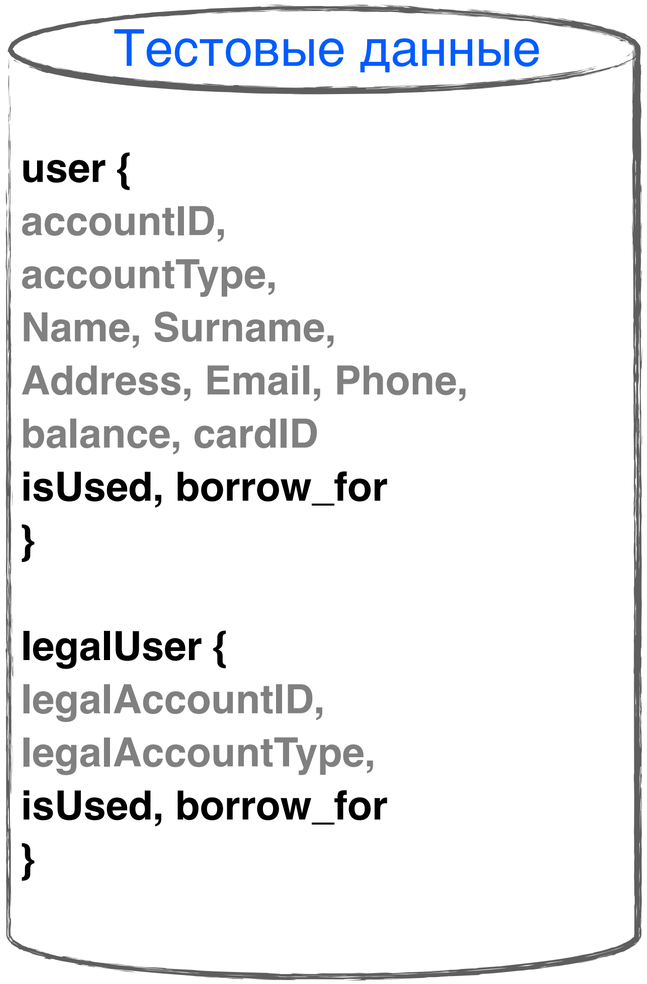
Типовой для разработки стек — шаблон для микросервисов:

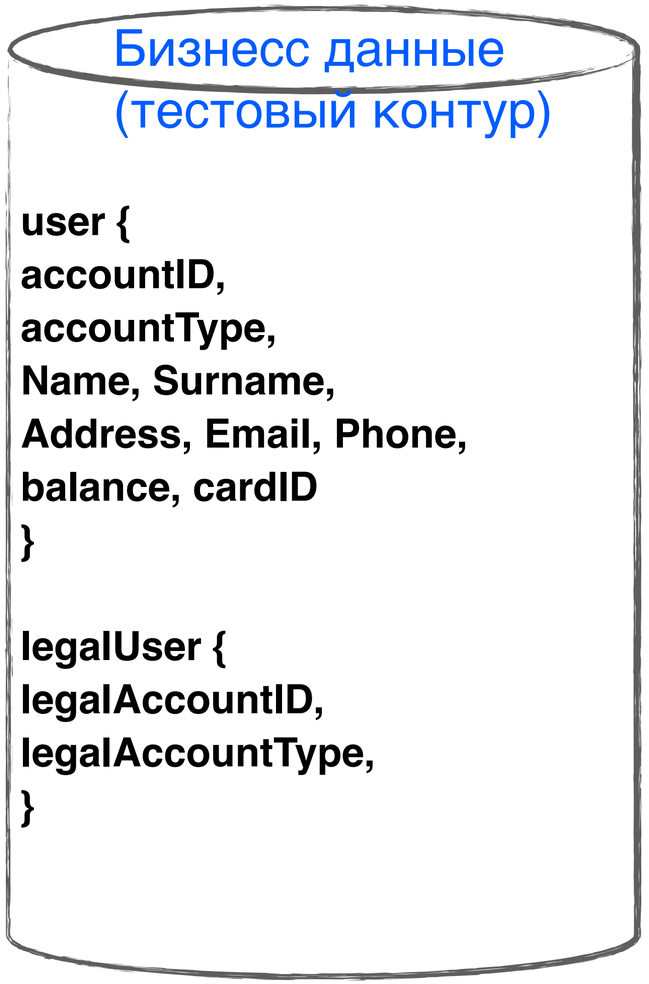
- **Go** + **Framework (http/grpc)** + **Swagger**
- Хранилище – **Postgres**
- Метрики в **Grafana** + **алерты в чаты**
- Горутины с воркерами, которые включают при необходимости в данных

```
go watcher() {
недостаточно( ) -> запустить генерацию( )
собрать использованных( ) -> переподготовить( )
}
```

Данные: организация и жизненный цикл

ozon fintech





Данные: организация и жизненный цикл

ozon fintech

Тестовые данные

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
isUsed, borrow_for
}

legalUser {
legalAccountID,
legalAccountType,
isUsed, borrow_for
}

Создание:

User = CreateUser()
User.RegisterAccount().Upgrade().AttachCard().AddCash()

User.SetFree()

Отметка в тестовых данных:
Пользователь: доступен

Использование:

User.SetBusy().BorrowFor(“2*days“)

* Тесты пройдены *

* Прошло два дня *

go **Watcher**() {
User = GetBusy(now > borrow_for)
User.DetachCard().EraseOperations().StripCash()
User.SetFree()

Бизнесс данные
(тестовый контур)

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
}

legalUser {
legalAccountID,
legalAccountType,
}

Данные: организация и жизненный цикл

ozon fintech

Тестовые данные

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
isUsed, borrow_for
}

legalUser {
legalAccountID,
legalAccountType,
isUsed, borrow_for
}

Создание:

User = CreateUser()

User.RegisterAccount().Upgrade().AttachCard().AddCash()

User.SetFree()

Отметка в тестовых данных:
Пользователь: доступен

Использование:

User.SetBusy().BorrowFor(“2*days“)

* Тесты пройдены *

* Прошло два дня *

go **Watcher**() {
User = GetBusy(now > borrow_for)
User.DetachCard().EraseOperations().StripCash()
User.SetFree()

Бизнесс данные
(тестовый контур)

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
}

legalUser {
legalAccountID,
legalAccountType,
}

Данные: организация и жизненный цикл

ozon fintech

Тестовые данные

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
isUsed, borrow_for
}

legalUser {
legalAccountID,
legalAccountType,
isUsed, borrow_for
}

Создание:

User = CreateUser()

User.RegisterAccount().Upgrade().AttachCard().AddCash()

User.SetFree()

Отметка в тестовых данных:
Пользователь: доступен

Использование:

User.SetBusy().BorrowFor(“2*days“)

* Тесты пройдены *

* Прошло два дня *

go **Watcher**() {

User = GetBusy(now > borrow_for)

User.DetachCard().EraseOperations().StripCash()

User.SetFree()

Бизнесс данные
(тестовый контур)

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
}

legalUser {
legalAccountID,
legalAccountType,
}

Данные: организация и жизненный цикл

ozon fintech

Тестовые данные

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
isUsed, borrow_for
}

legalUser {
legalAccountID,
legalAccountType,
isUsed, borrow_for
}

Создание:

User = CreateUser()

User.RegisterAccount().Upgrade().AttachCard().AddCash()

User.SetFree()

Отметка в тестовых данных:
Пользователь: доступен

Использование:

User.SetBusy().BorrowFor(“2*days“)

* Тесты пройдены *

* Прошло два дня *

go **Watcher**() {

User = GetBusy(now > borrow_for)

User.DetachCard().EraseOperations().StripCash()

User.SetFree()

Бизнесс данные
(тестовый контур)

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
}

legalUser {
legalAccountID,
legalAccountType,
}

Данные: организация и жизненный цикл

ozon fintech

Тестовые данные

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
isUsed, borrow_for
}

legalUser {
legalAccountID,
legalAccountType,
isUsed, borrow_for
}

Создание:

User = CreateUser()

User.RegisterAccount().Upgrade().AttachCard().AddCash()

User.SetFree()

Отметка в тестовых данных:
Пользователь: доступен

Использование:

User.SetBusy().BorrowFor(“2*days“)

* Тесты пройдены *

* Прошло два дня *

go **Watcher**() {

User = GetBusy(now > borrow_for)

User.DetachCard().EraseOperations().StripCash()

User.SetFree()

Бизнесс данные
(тестовый контур)

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
}

legalUser {
legalAccountID,
legalAccountType,
}

Данные: организация и жизненный цикл

ozon fintech

Тестовые данные

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
isUsed, borrow_for
}

legalUser {
legalAccountID,
legalAccountType,
isUsed, borrow_for
}

Создание:

User = CreateUser()

User.RegisterAccount().Upgrade().AttachCard().AddCash()

User.SetFree()

Отметка в тестовых данных:
Пользователь: доступен

Использование:

User.SetBusy().BorrowFor(“2*days“)

* Тесты пройдены *

* Прошло два дня *

go **Watcher**() {
User = GetBusy(now > borrow_for)
User.DetachCard().EraseOperations().StripCash()
User.SetFree()

Бизнесс данные
(тестовый контур)

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
}

legalUser {
legalAccountID,
legalAccountType,
}

Данные: организация и жизненный цикл

ozon fintech

Тестовые данные

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
isUsed, borrow_for
}

legalUser {
legalAccountID,
legalAccountType,
isUsed, borrow_for
}

Создание:

User = CreateUser()

User.RegisterAccount().Upgrade().AttachCard().AddCash()

User.SetFree()

Отметка в тестовых данных:
Пользователь: доступен

Использование:

User.SetBusy().BorrowFor(“2*days“)

* Тесты пройдены *

* Прошло два дня *

go **Watcher**() {

User = GetBusy(now > borrow_for)

User.DetachCard().EraseOperations().StripCash()

User.SetFree()

Бизнесс данные
(тестовый контур)

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
}

legalUser {
legalAccountID,
legalAccountType,
}

Данные: организация и жизненный цикл

ozon fintech

ГЕНЕРАЦИЯ

Создание:

User = CreateUser()

User.RegisterAccount().Upgrade().AttachCard().AddCash()

User.SetFree()

Отметка в тестовых данных:
Пользователь: доступен

Использование:

User.SetBusy().BorrowFor(“2*days“)

ПЕРЕИСПОЛЬЗОВАНИЕ

* Тесты пройдены *

* Прошло два дня *

go **Watcher**() {

User = GetBusy(now > borrow_for)

User.DetachCard().EraseOperations().StripCash()

User.SetFree()

Тестовые данные

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
isUsed, borrow_for
}

legalUser {
legalAccountID,
legalAccountType,
isUsed, borrow_for
}

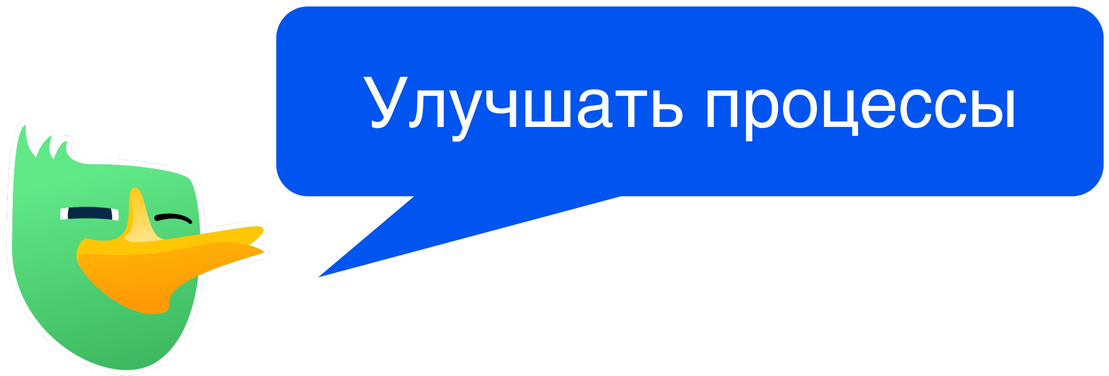
Бизнесс данные
(тестовый контур)

user {
accountID,
accountType,
Name, Surname,
Address, Email, Phone,
balance, cardID
}

legalUser {
legalAccountID,
legalAccountType,
}

Постоянно генерировать

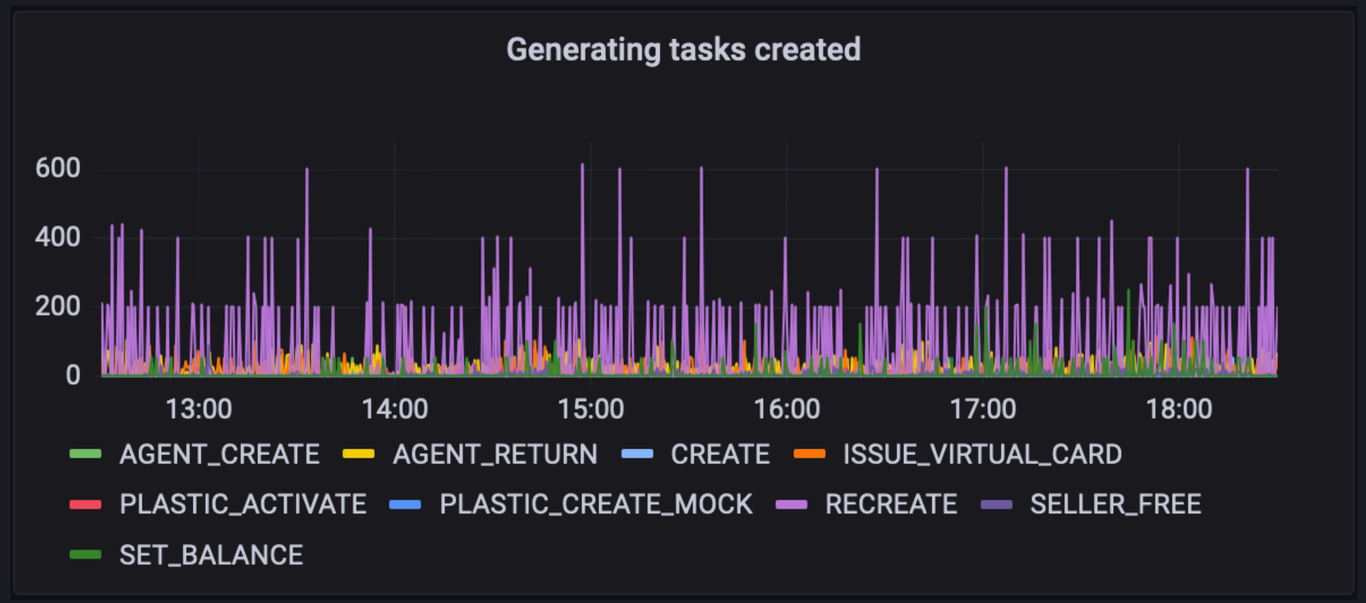
ozon fintech

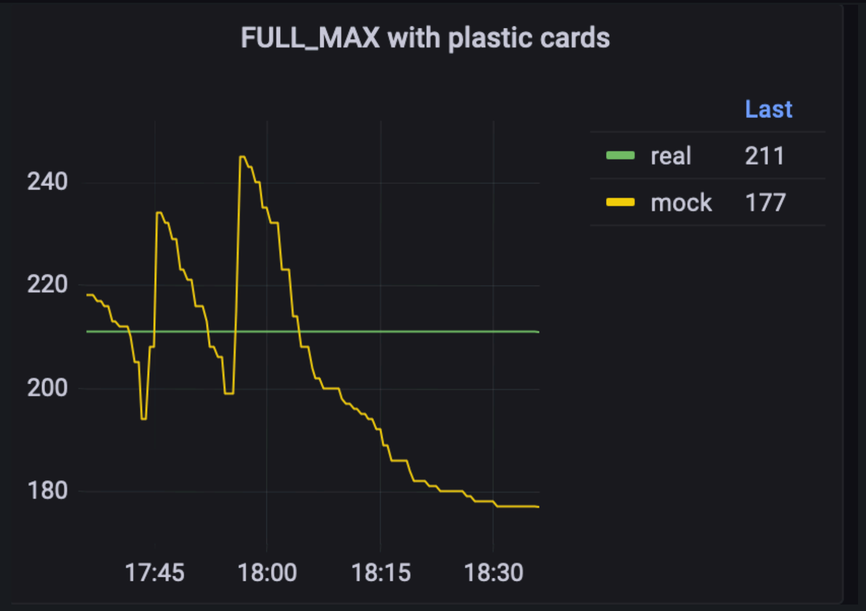


Не нашли с достаточным балансом	• Берем бедного и начисляем денег
Много стандартных, мало премиальных	• Брать их и повышать

Постоянно генерировать

ozon fintech





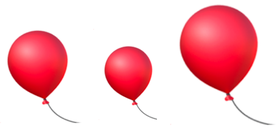
Постоянно генерировать

ozon fintech

Улучшать алерты и дежурства

Ошибки повышения	• customers_duty
Ошибки баланса	• accounts_duty
Ошибки карт	• processing_duty

Данные: сколько?



ozon fintech

Физические счета

- Без карты, с виртуальной картой с пластиковой
- Без денег/с суммой от 10 до нескольких миллионов
- С признаком «нехорошего человека» четырех типов
- С кредитом/с товаром в кредит

Юридические счета

- Шесть типов юр.лиц
- Агенты выдачи карт

Пластиковые карты

- Через тестовый процессинг
- в обход процессинга

Различные события

Зачем так сложно?

ozon fintech

- Строили планы на использование
- Надо выделять специальных людей (это не бесплатно, но на дистанции гораздо выгодней; многие не выделяют и закрывают вопрос числом)
- Много ответственности

Инструмент для многих

ozon fintech

Viktor

Что мы умеем?

Пополнить счет

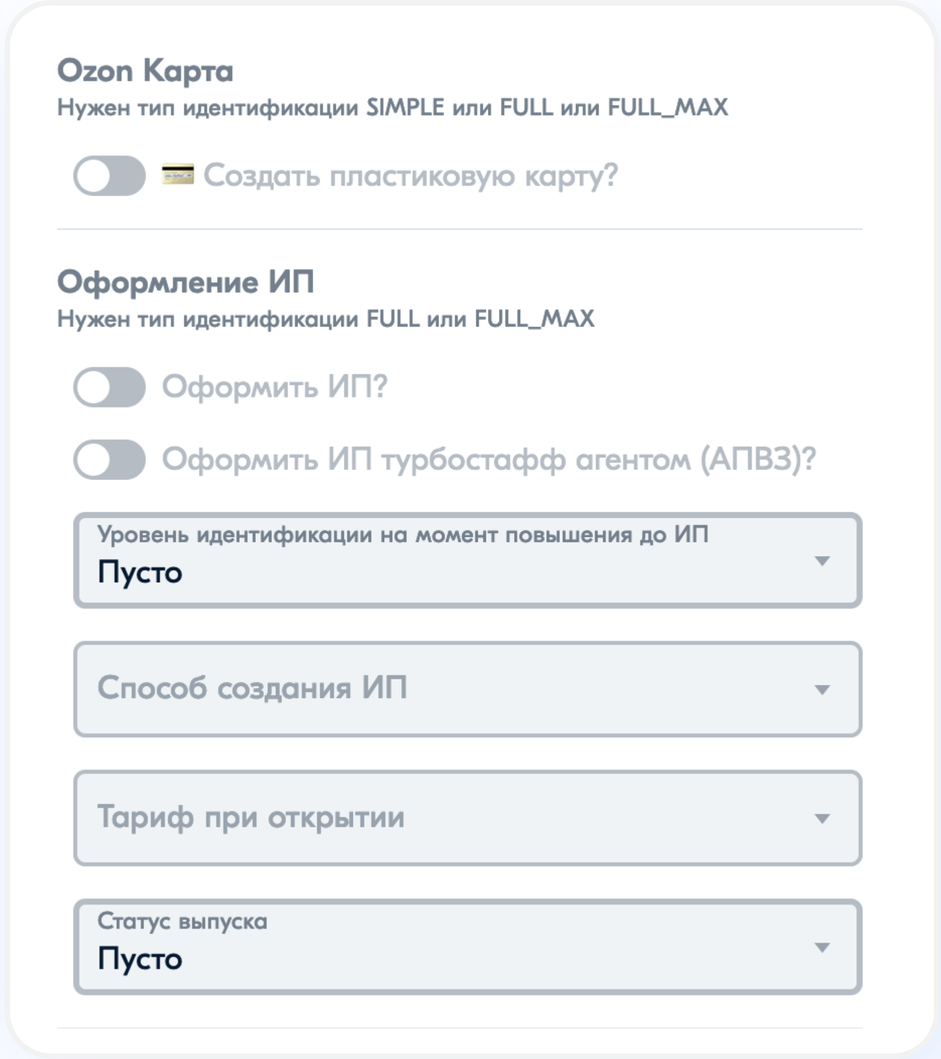
Создать нового клиента

Создать новую пластиковую карту

Собрать ·meshversion

Сгенерировать QR код для оплаты/подписки C2B

Создать агента выдачи карт

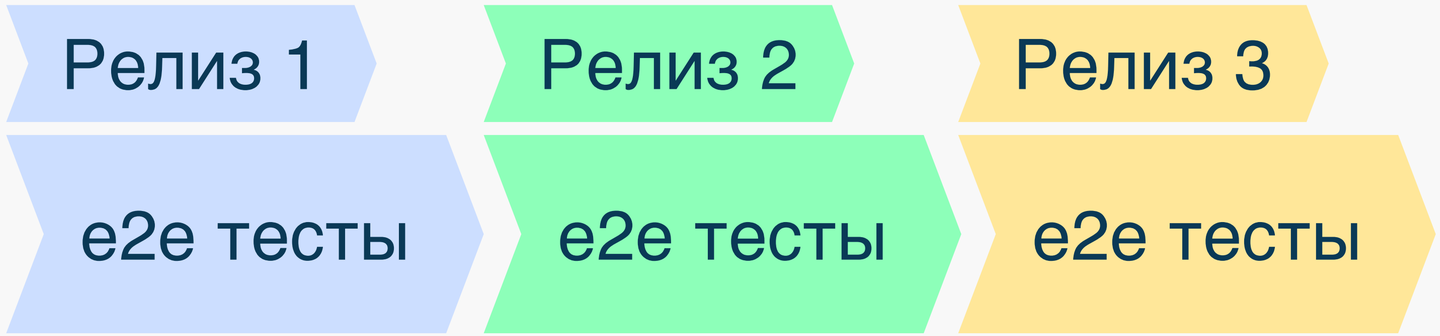


Итог

ozon fintech

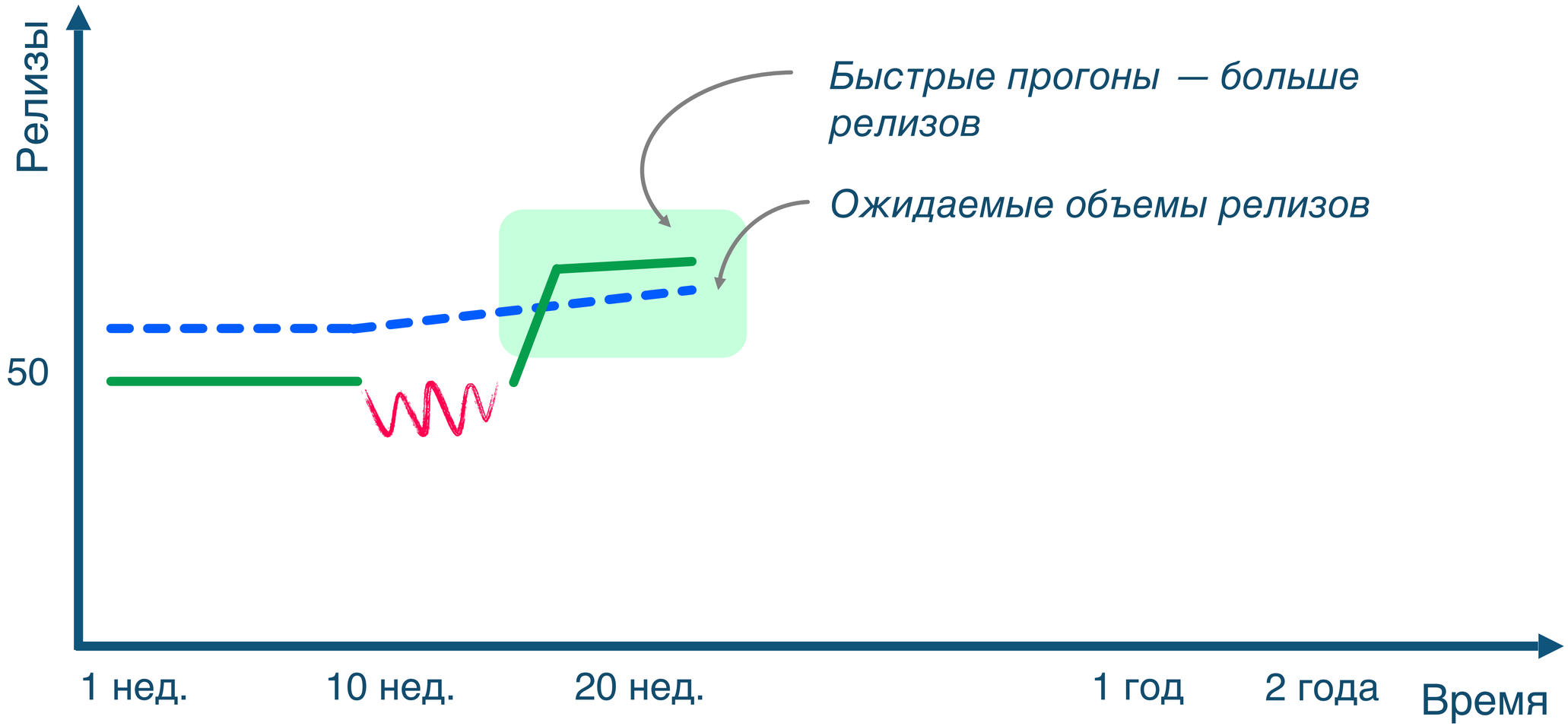
Скорость тестов 15мин -> 3-4мин + Распределение нагрузки

todo красивая стрелка



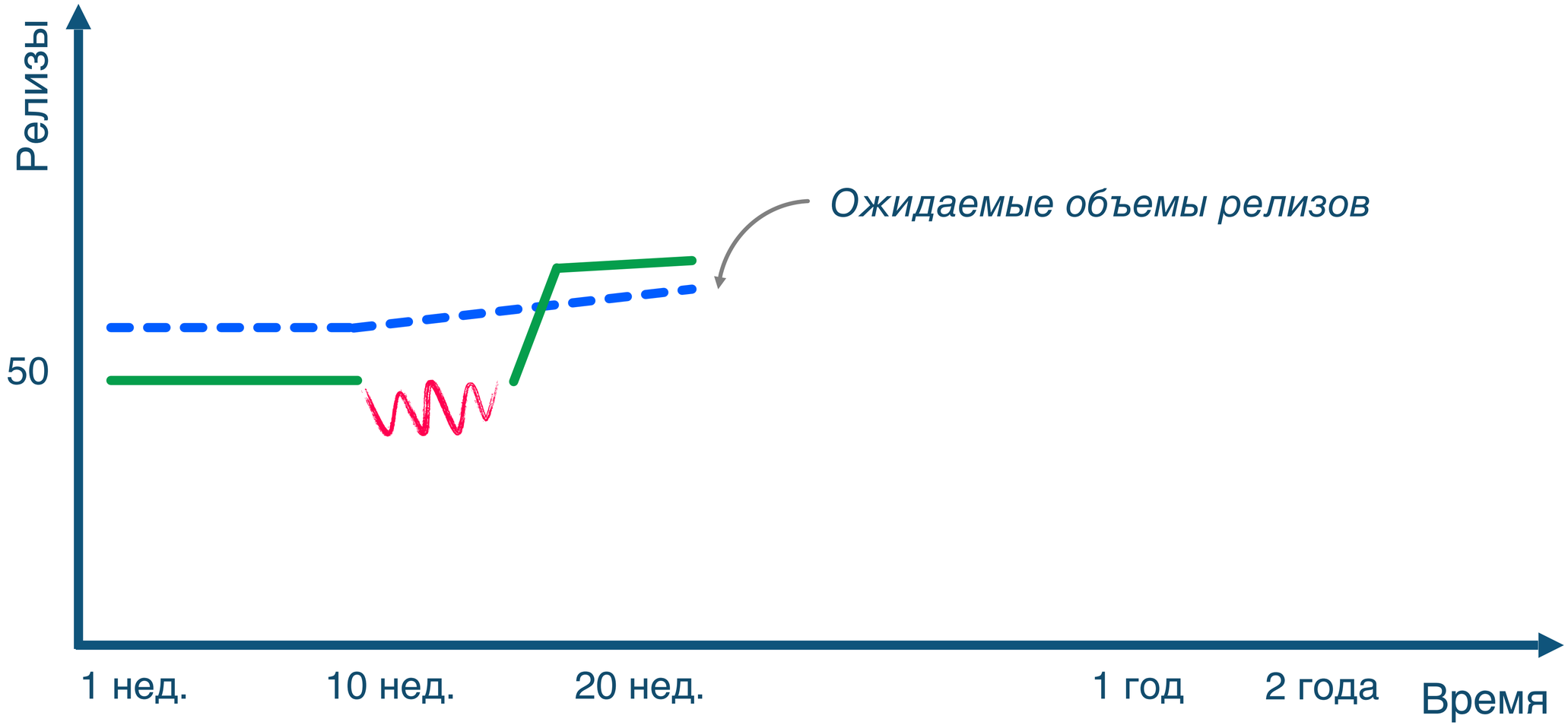
Итог

ozon fintech



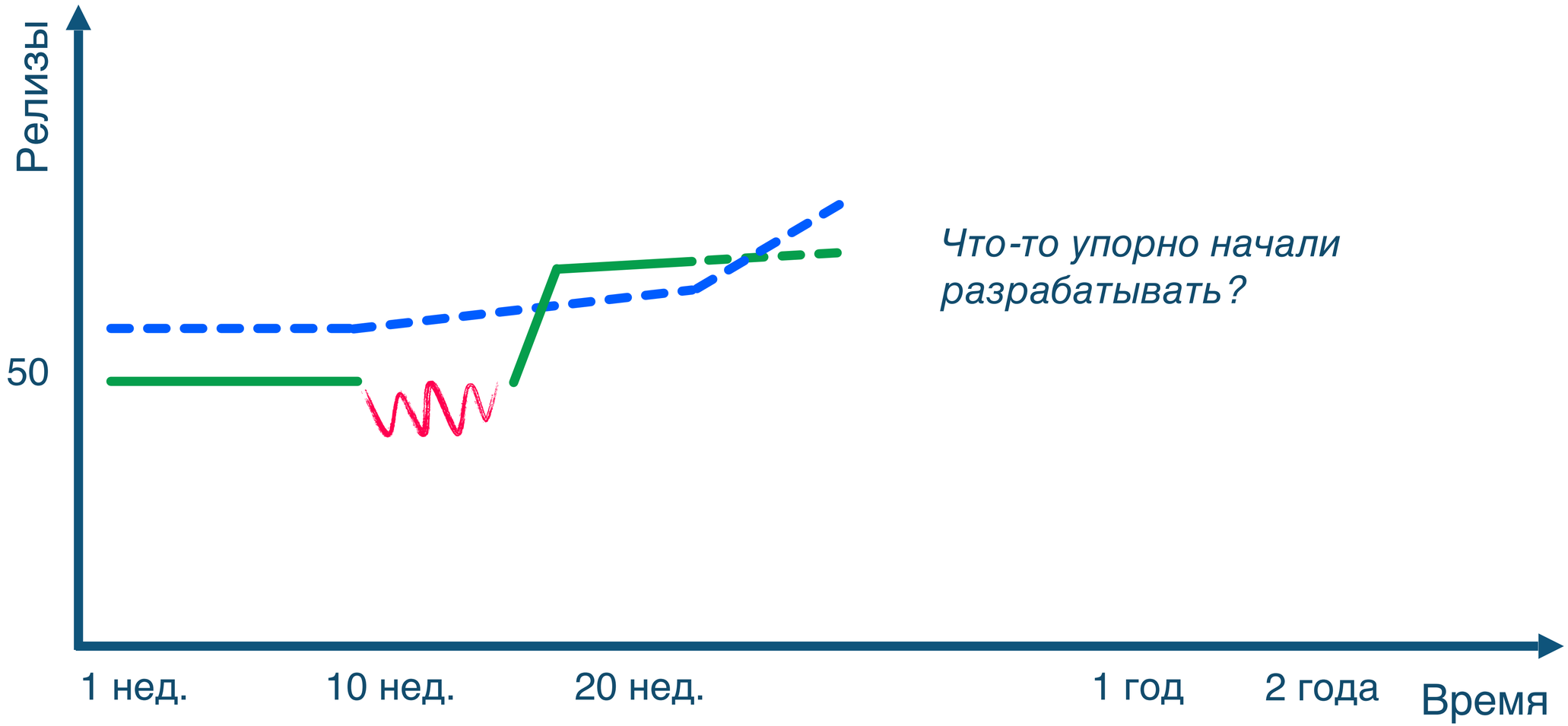
Новые ~~проблемы~~ вызовы

ozon fintech



Новые ~~проблемы~~ вызовы

ozon fintech



Что-то упорно начали разрабатывать?

Появление внешней системы

ozon fintech

- Разработка сервисов для интеграции
- +100 сценариев тестирования

Следующие проблемы

ozon fintech

Группа сервисов интеграции с внешним партнером

Проблема 1

Внешний партнер не дает тестовый контур
(или дает и вы его перегружаете одним прогоном)

Проблема 2

Входящие сценарии не написать никак
(оплата картой, входящий перевод, входящий штраф, пуш, etc.)

Итог: мало сценариев (и тем более автотестов) — нет релизов

ozon fintech

Требуется

- Реализовать сценарии
- Исключить влияние на наш тестовый контур

Сценарии

ozon fintech

- Простые и сложные
- Сложнее взаимодействие — сложнее реализовать тест

Сценарии

ozon fintech

- Простой случай: курс валюты

Запрос: сколько за 1 тугрик

Ответ: 50 рублей

Сценарии

ozon fintech

- Заменяем внешний запрос хардкодом

Запрос: сколько за 1 тугрик

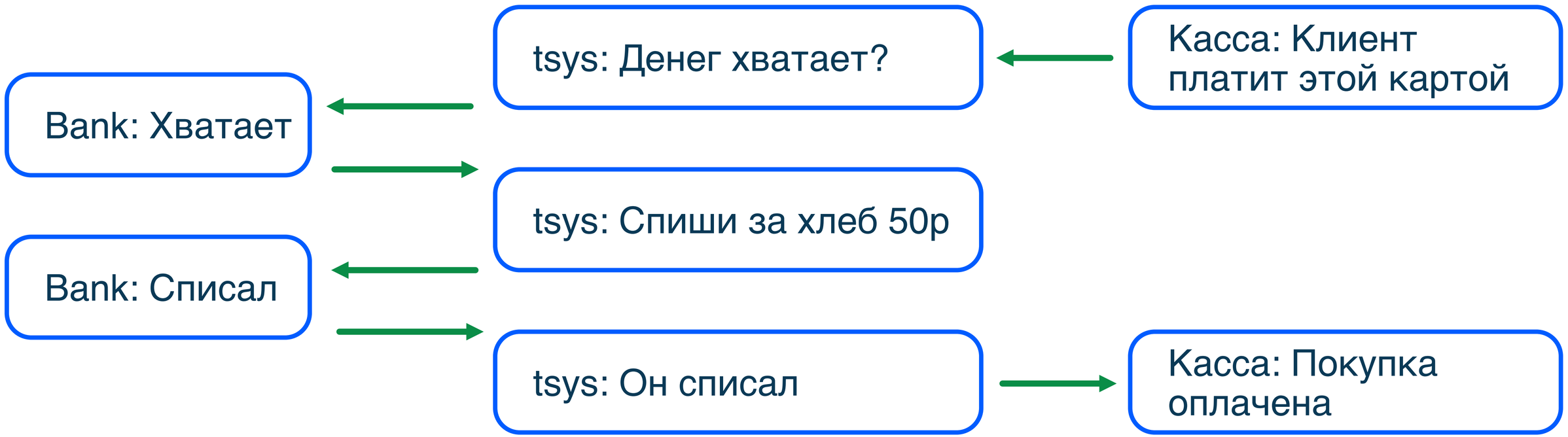
Ответ: 50 рублей

Сценарии

ozon fintech

- Сложный случай: платеж картой в супермаркете

Старт: оплатили хлеб картой на кассе

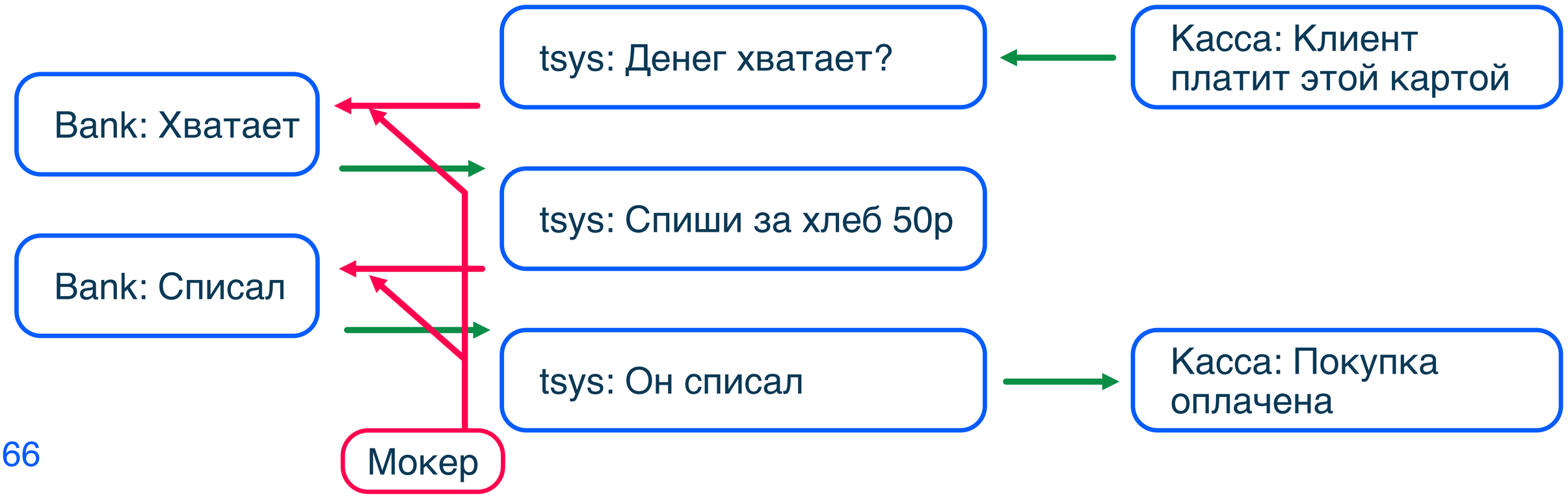


Сценарии

ozon fintech

- Утилита-мокер подменяет ответ

Старт: оплатили хлеб картой на кассе

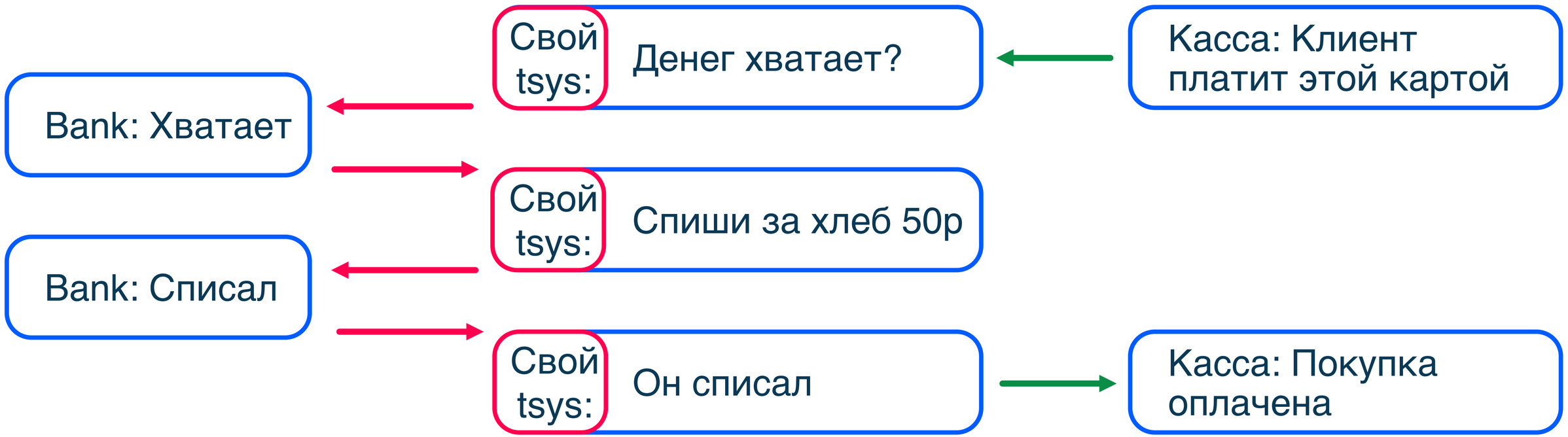


Сценарии

ozon fintech

- Копия внешнего сервиса эмулирует работу

Старт: оплатили хлеб картой на кассе

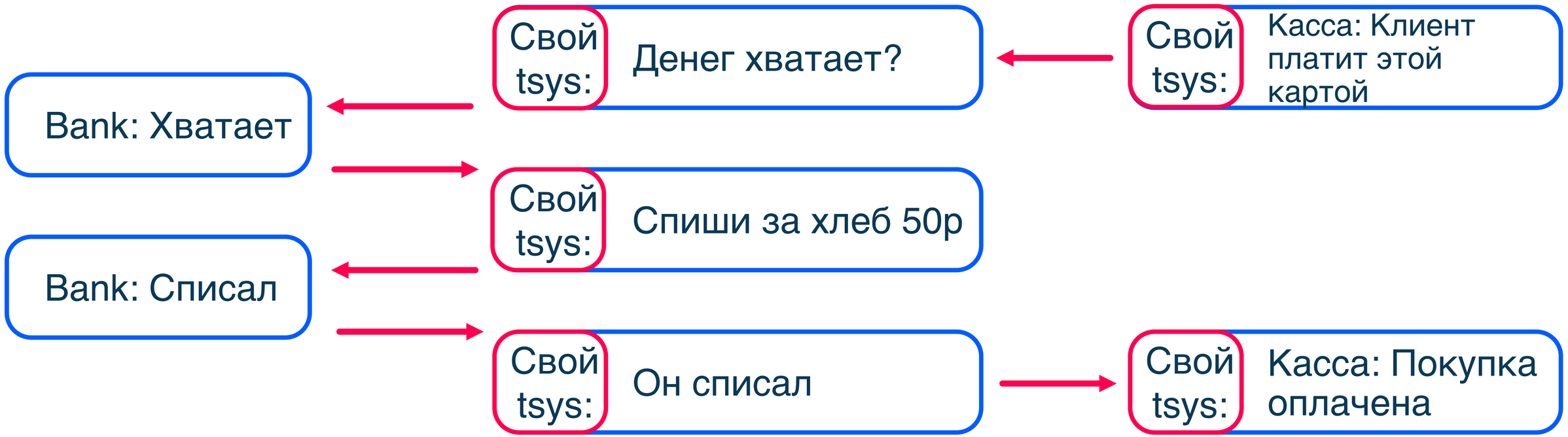


Сценарии

ozon fintech

- Копия внешнего сервиса эмулирует работу и добавляет функционала

Старт: оплатили хлеб картой на кассе



Зачем так сложно?

ozon fintech

- Строили планы на использование
- Надо выделять специальных людей
- Вроде можно воспользоваться мок-инструментами, но:
  - коробочные решения показали проблемы стабильности
  - свой проект можно сильно модифицировать
- Много ответственности

Затем:

ozon fintech

1. Автотесты на сценарии извне
2. Свой мок-сервис — можно оптимизировать

Затем:

ozon fintech

1. Автотесты на сценарии извне
2. Свой мок-сервис — можно оптимизировать

1 + 2 =

3. Получили сценарное нагрузочное тестирование внешними покупками

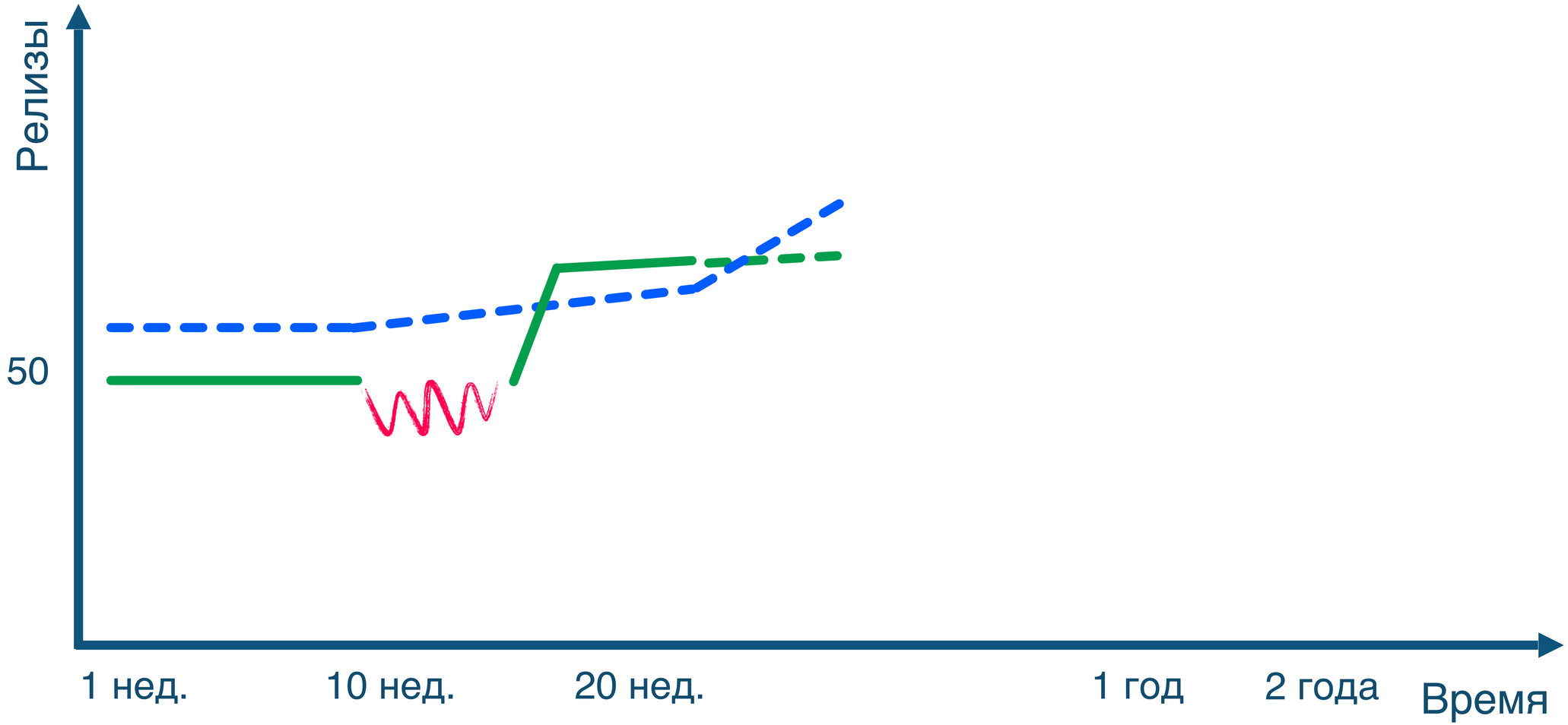
Итог

ozon fintech

- Решили для сложных сценариев с внешним траффиком
- Устранили ненадежные компоненты
- Теперь 500 стабильных e2e автотестов

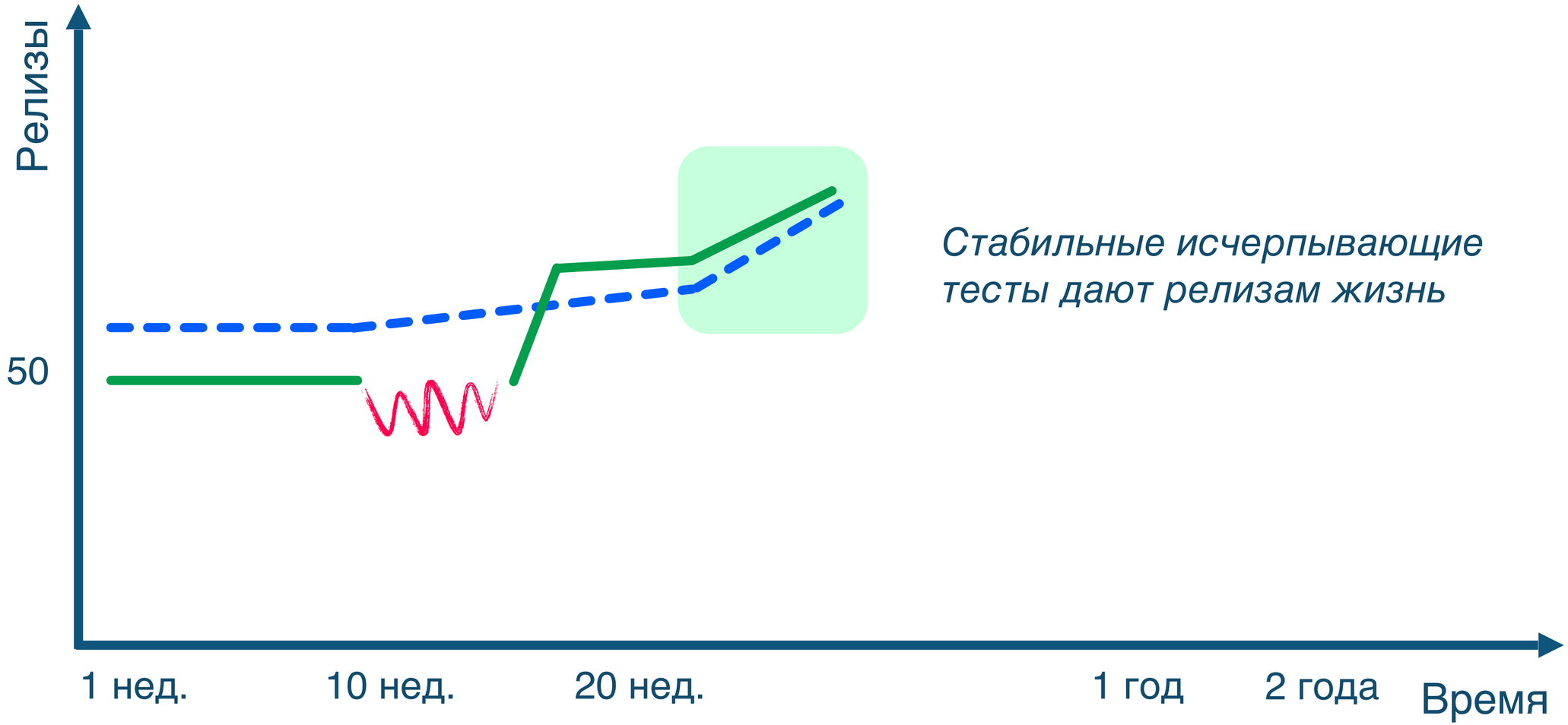
Новые ~~проблемы~~ вызовы: были

ozon fintech



Новые ~~проблемы~~ вызовы: решены

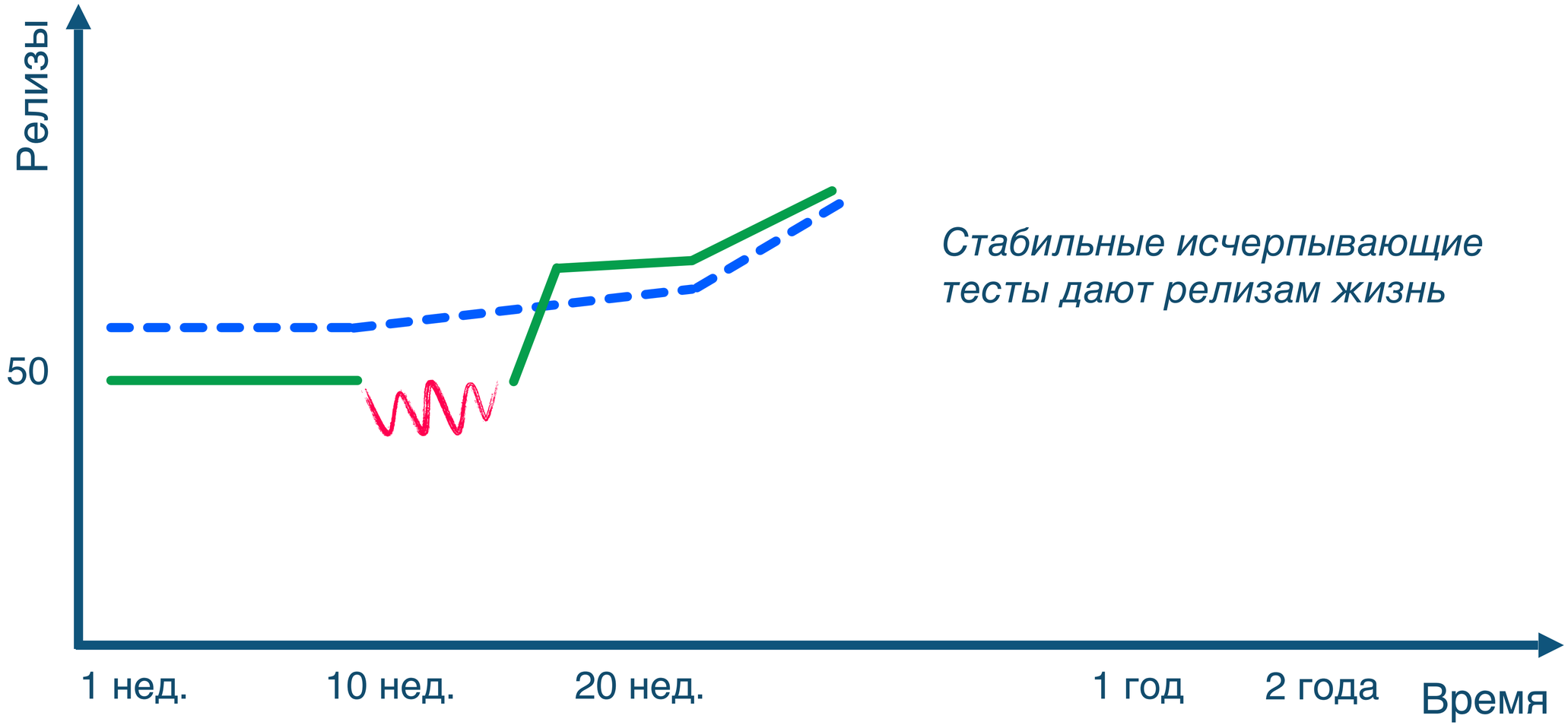
ozon fintech



Стабильные исчерпывающие тесты дают релизам жизнь

Не расслабляемся

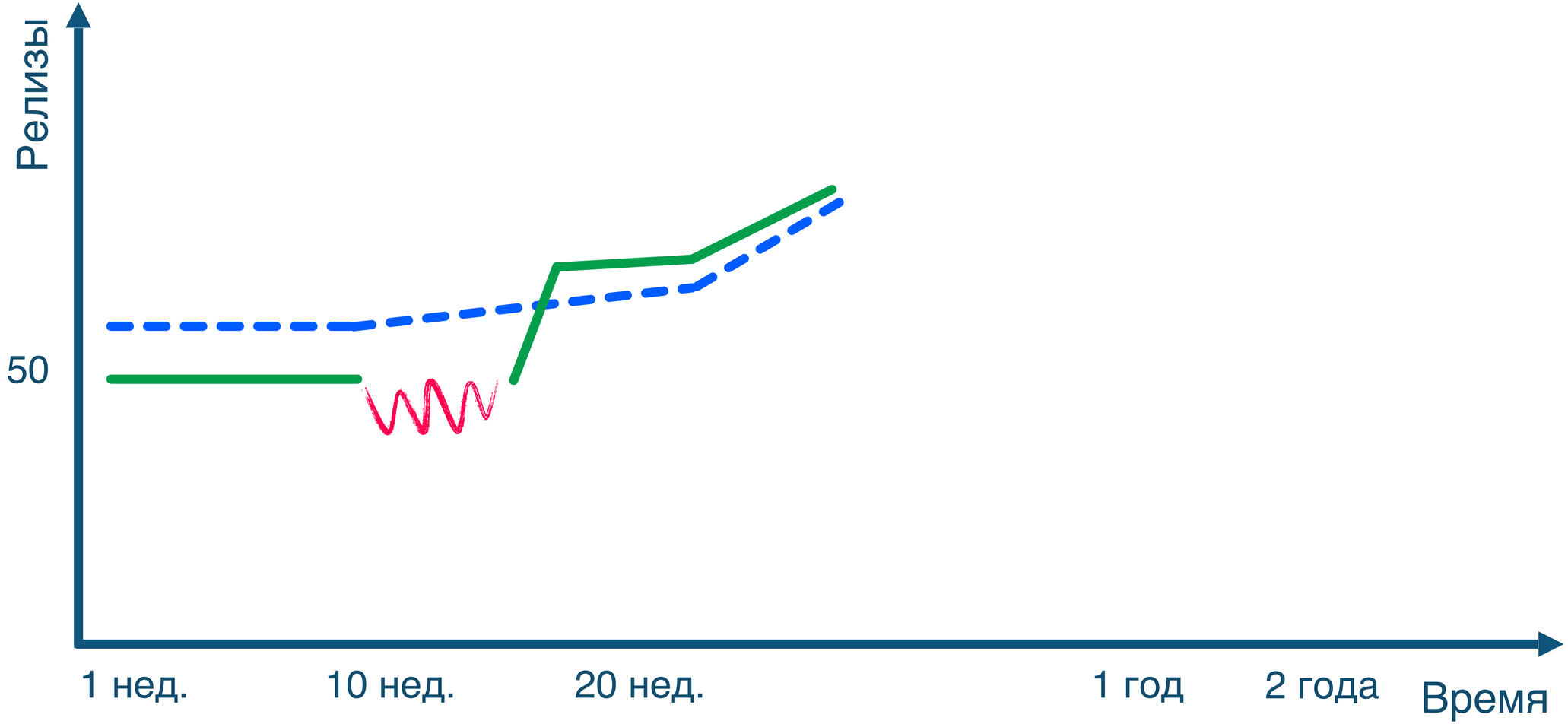
ozon fintech



Стабильные исчерпывающие тесты дают релизам жизнь

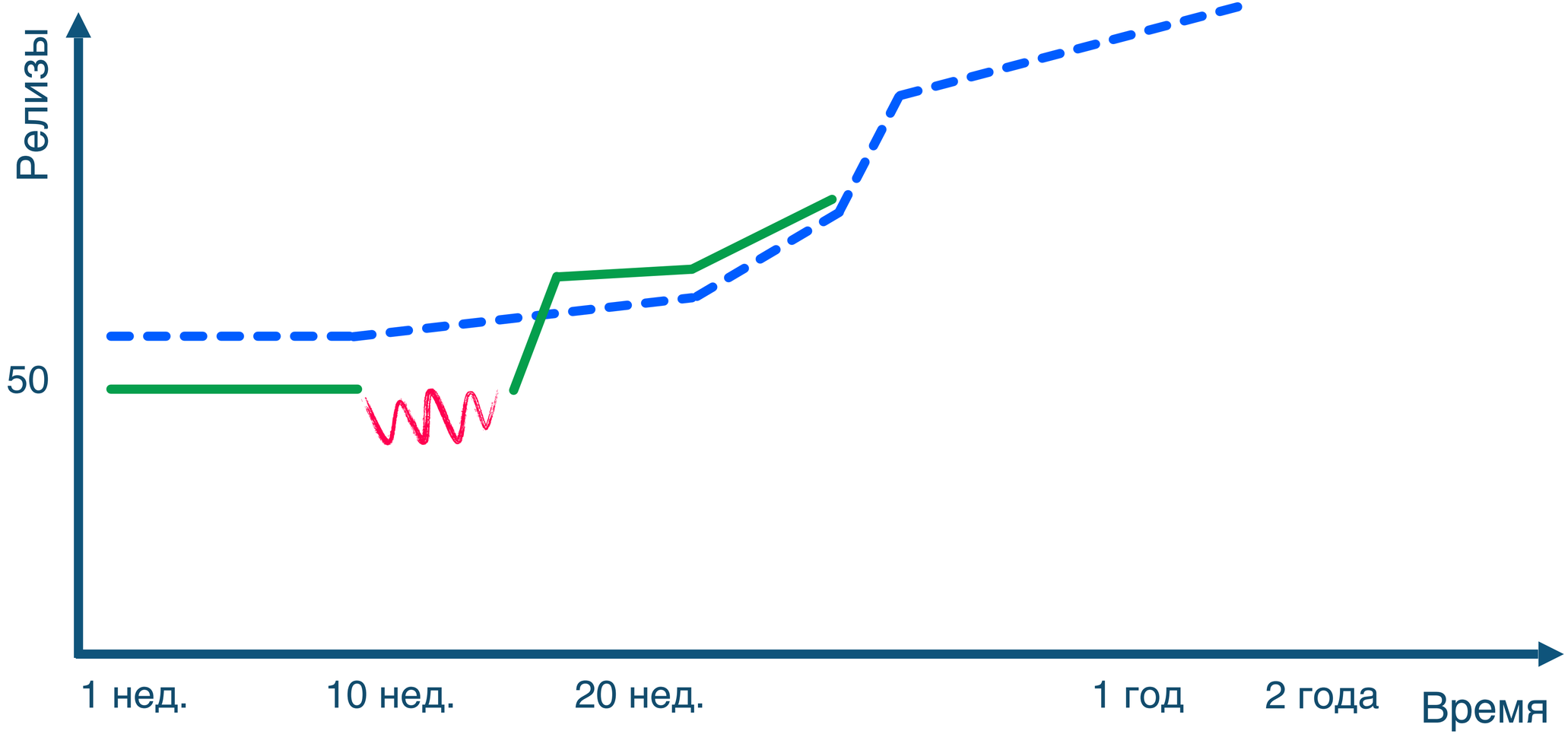
Бизнес: рост x2, найм x2, релизы x2

ozon fintech



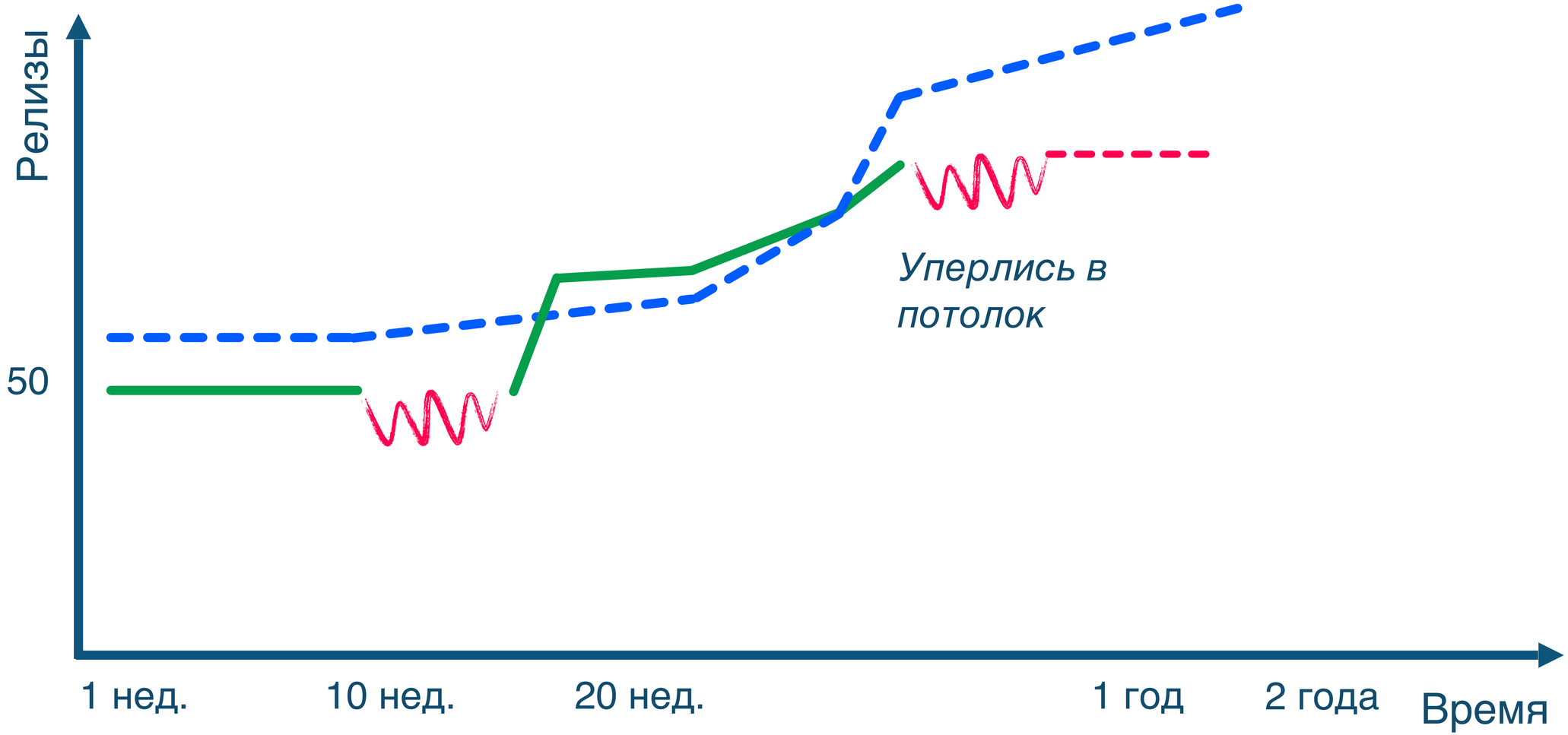
Бизнес: рост x2, найм x2, релизы x2

ozon fintech



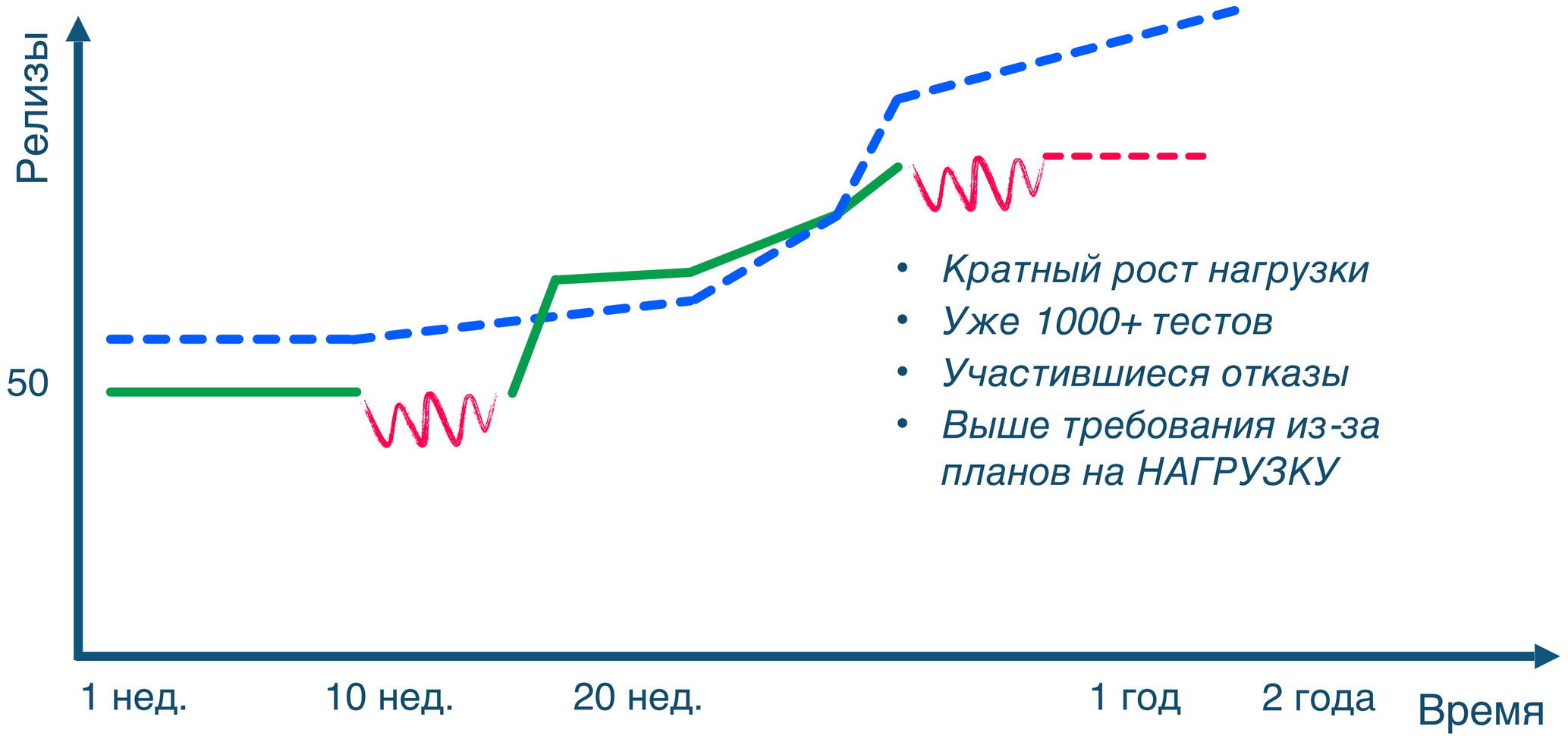
Бизнес: рост x2, найм x2, релизы x2

ozon fintech



Проблемы

ozon fintech



- *Кратный рост нагрузки*
- *Уже 1000+ тестов*
- *Участившиеся отказы*
- *Выше требования из-за планов на НАГРУЗКУ*

Необходимость

ozon fintech

Ускорить тесты (еще больше)

Запускать меньше без потери качества

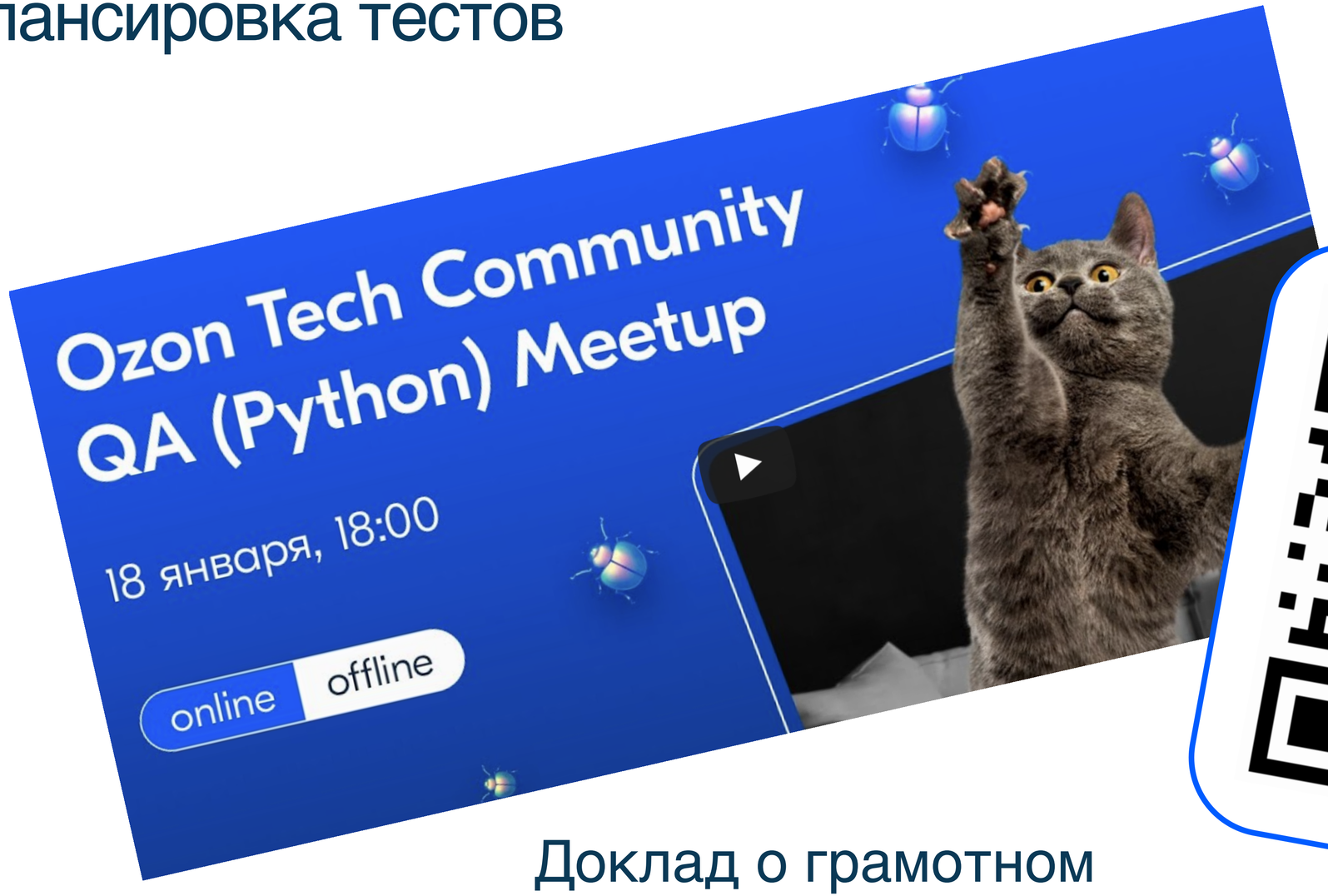
Стабилизировать тестовый контур

Ускорить тесты
(еще больше)

Балансировка

Балансировка тестов

ozon fintech



Доклад о грамотном расспараллеливании

e2e тесты

e2e тесты

ozon fintech

Allure timeline: анализ длительности

- Долгие хвосты, вплоть до двукратного времени прогона

Что делаем:

- Храним длительность тестов в отдельном сервисе
- С помощью pytest хука (collection modifyitems) отбалансировали тесты

e2e тесты

ozon fintech

Allure timeline: анализ длительности

- Долгие хвосты, вплоть до двукратного времени прогона

Что делаем:

- Храним длительность тестов в отдельном сервисе
- С помощью pytest хука (collection modifyitems) отбалансировали тесты

Как работает:

- Сначала раскладываются параллельно самые долгие тесты
- Затем оставшиеся пустоты заполняют равномерно мелкие тесты

e2e тесты

ozon fintech

Allure timeline: анализ длительности

- Долгие хвосты, вплоть до двукратного времени прогона

Что делаем:

- Храним длительность тестов в отдельном сервисе
- С помощью pytest хука (collection modifyitems) отбалансировали тесты

Что получили:

Как работает:

- Сначала раскладываются параллельно самые долгие тесты
- Затем оставшиеся пустоты заполняют равномерно мелкие тесты

ИТОГ: сократили время прохождения с 8 до 4-5 минут

Решения

ozon fintech

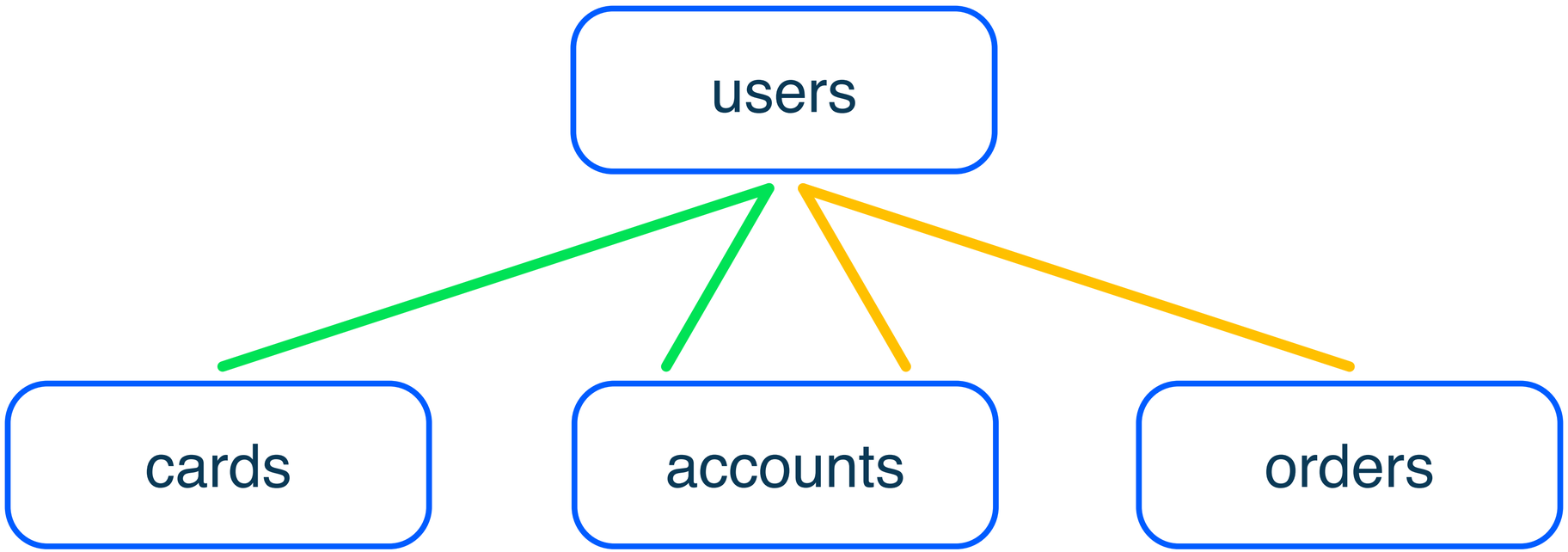
Запуск меньше
без потери качества

Запускать только
то, что нужно…

Трассировка

ozon fintech

Возможность определять путь запроса — трейс (след)



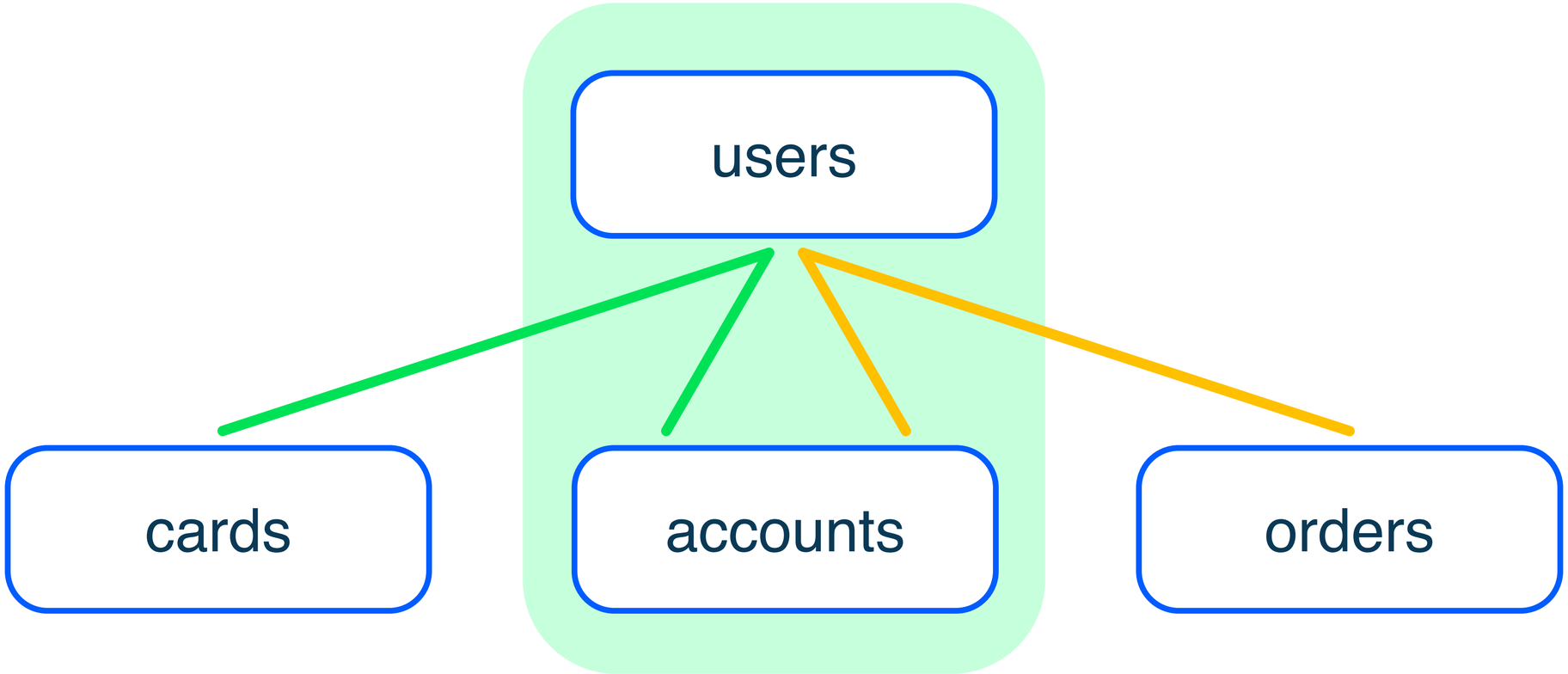
Пополнение из банкомата

Заказ товара

Трассировка

ozon fintech

Для релиза users или accounts — важны все сценарии



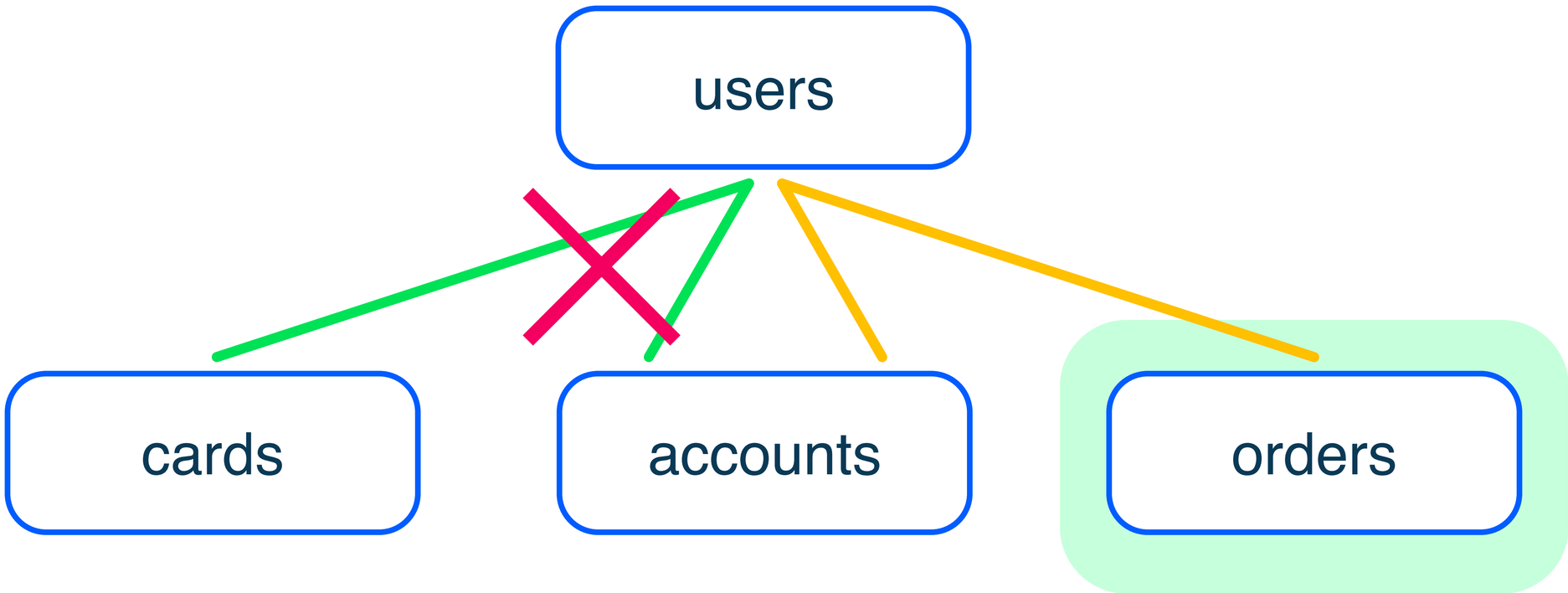
Пополнение из банкомата

Заказ товара

Трассировка

ozon fintech

Для релиза orders сценарий с банкоматом излишен



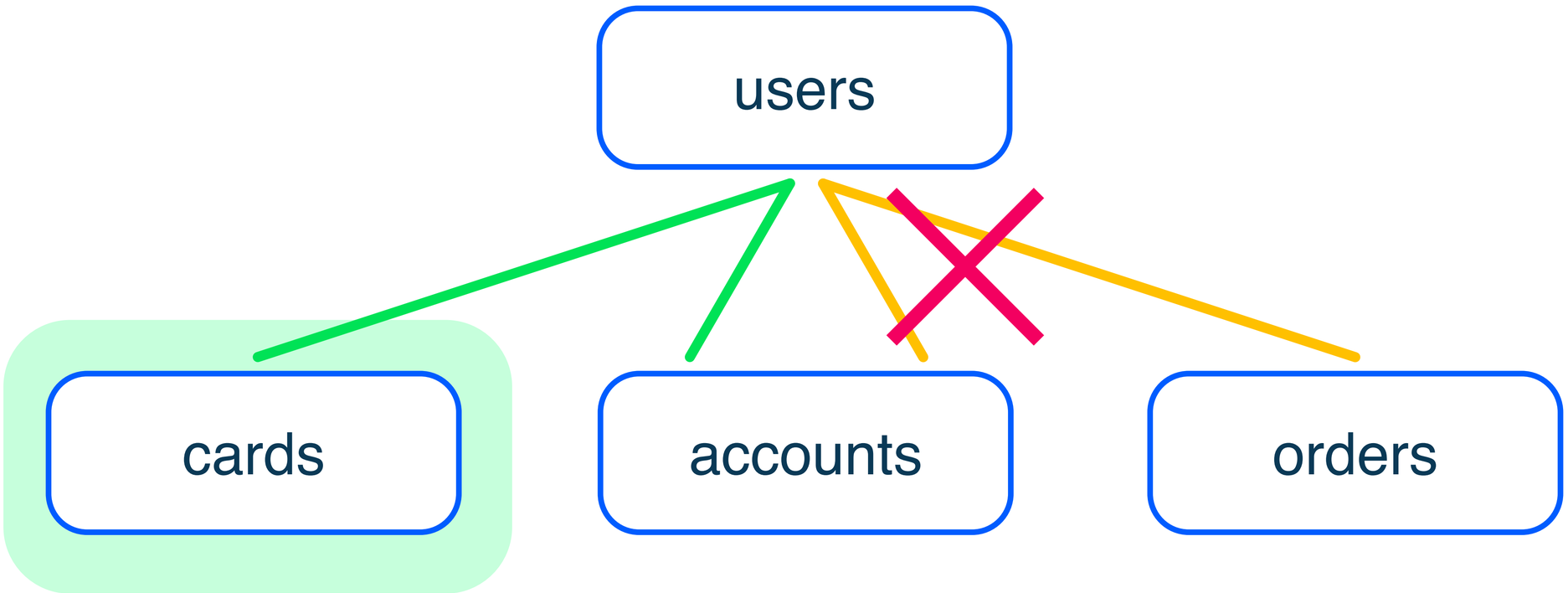
Пополнение из банкомата

Заказ товара

Трассировка

ozon fintech

Для релиза card излишен сценарий с товаром



Пополнение из банкомата

Заказ товара

Трассировка

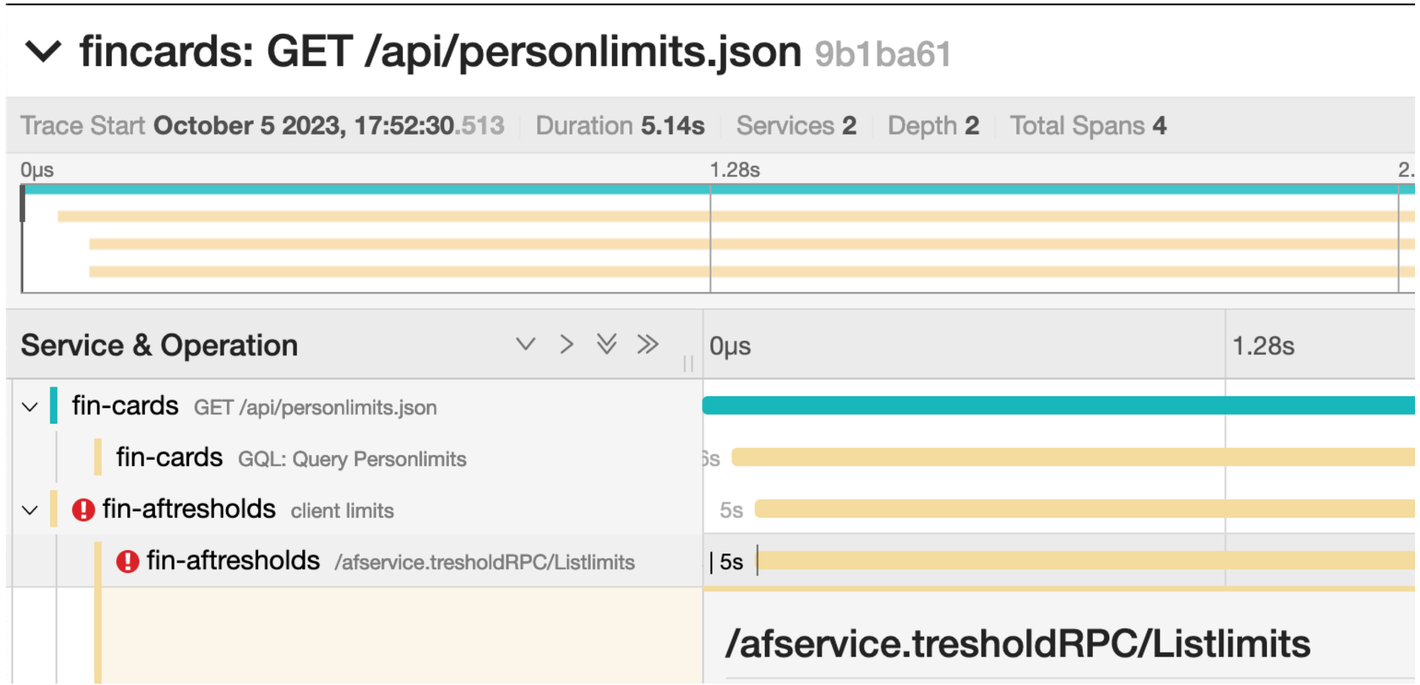
ozon fintech

Нужно всего лишь:

1. Собрать «следы»

2. Соотнести с тестами

Платформенная наработка + jaeger



Цепочки активностей

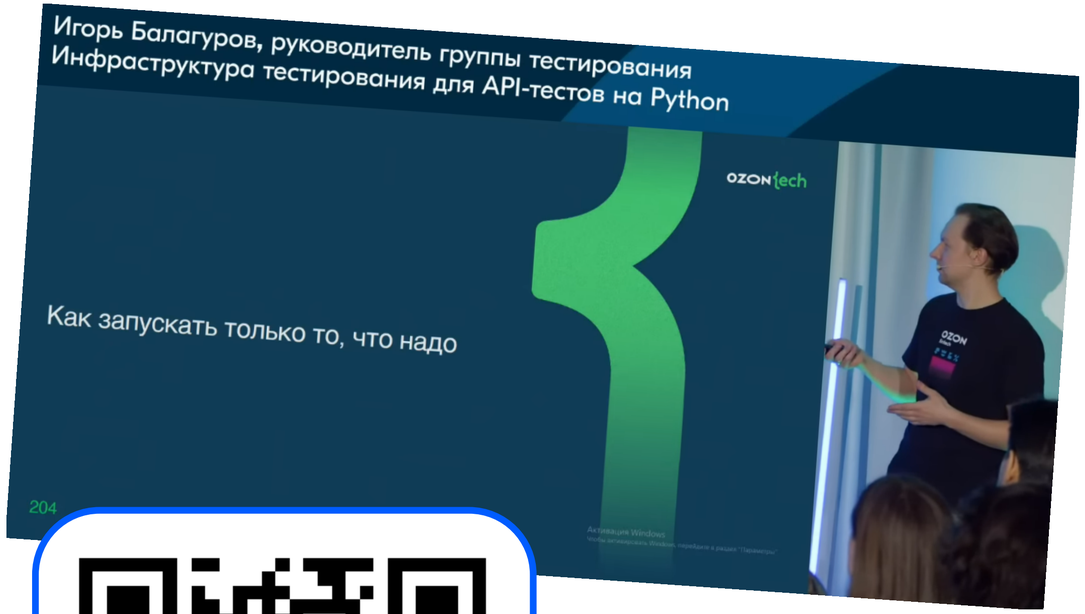
Начало– traceID без данных

Каждый шаг – сохраняем

На выходе id
и его цепочка активностей

Запуск меньше без потери качества: примеры TODO РАСПИСАНИЕ!

ozon fintech



Доклад Игоря Балагурова на тему «Инфраструктура тестирования для API-тестов на Python»

Доклад Алексея Романова на тему «Как мы оптимизировали e2e и дооптимизировались»

15.10 / 17:15 – 18:00

ДОКЛАД

Как мы оптимизировали запуски e2e и дооптимизировались

Алексей Романов
Ozon Fintech

RU

Tools/Frameworks

Итог внедрения

ozon fintech

Запуск

В большинстве релизов запускается 20-35% от общего количества e2e тестов

Без потери качества. Честно.

Решения

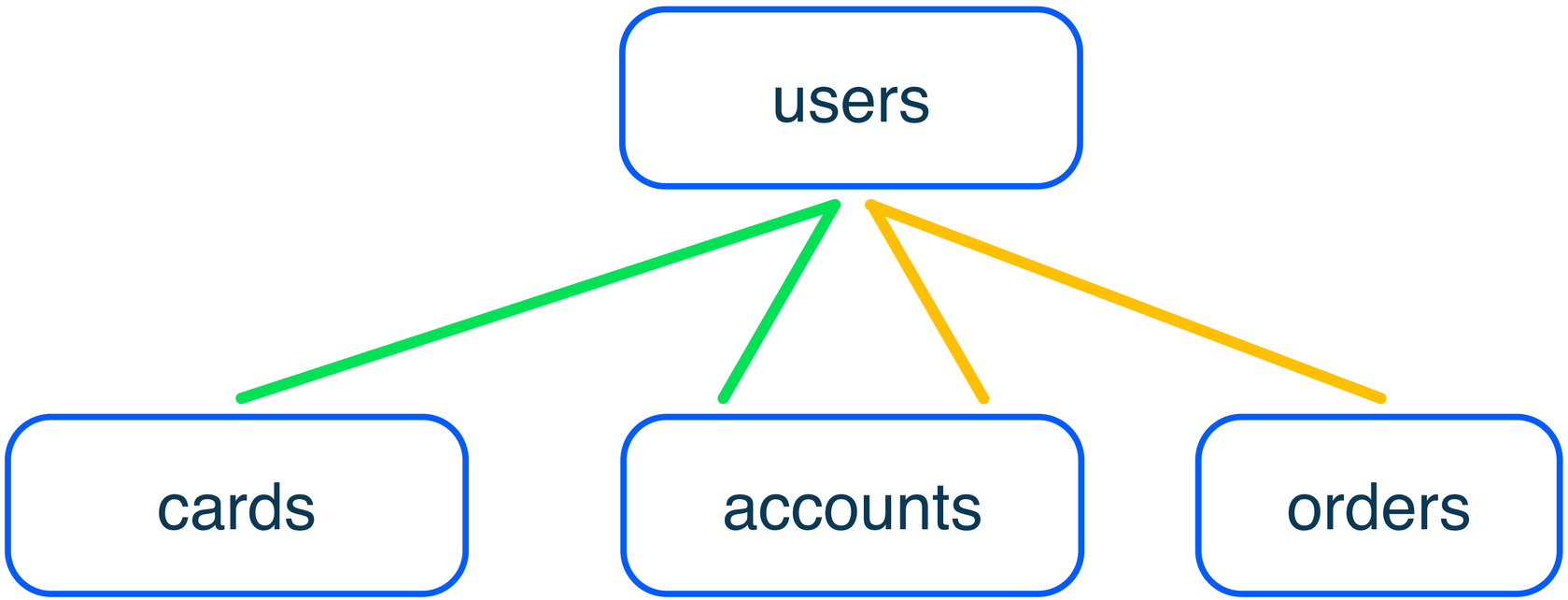
ozon fintech

Стабилизировать тестовый контур

Стабилизировать тестовый контур

ozon fintech

Версионирование



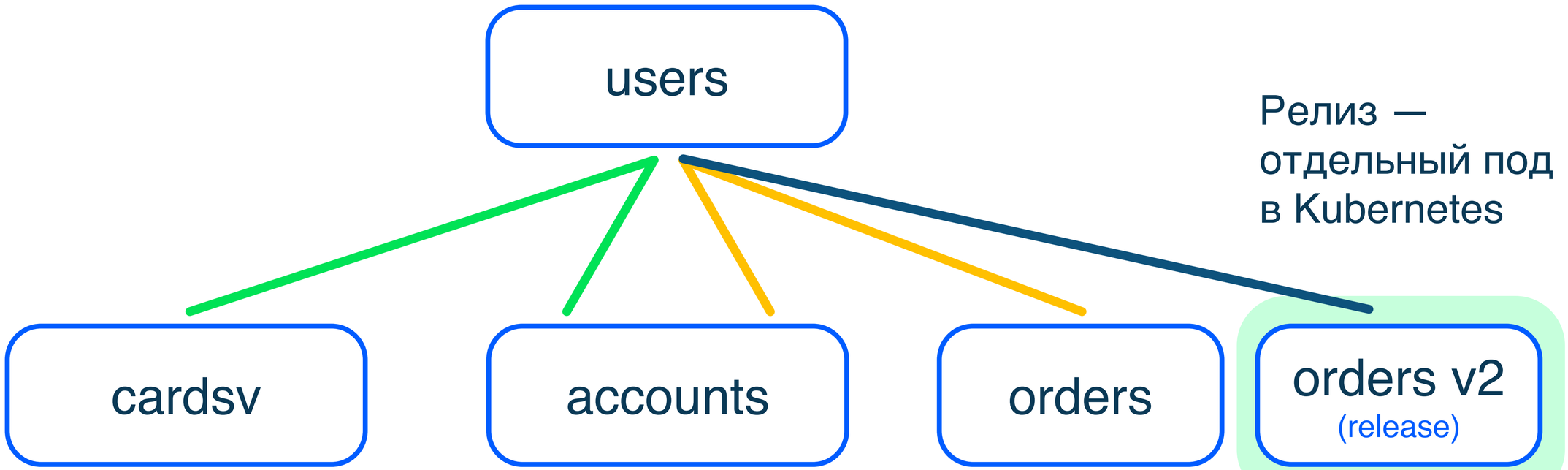
Пополнение из банкомата

Заказ товара

Стабилизировать тестовый контур

ozon fintech

Версионирование



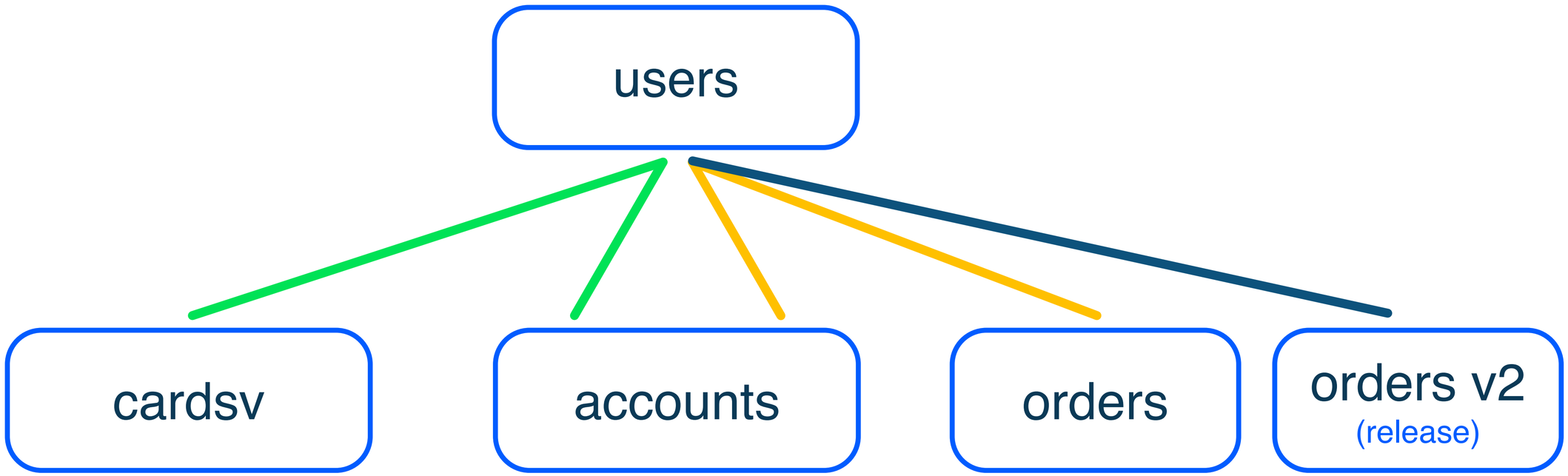
Пополнение из банкомата

Заказ товара

Стабилизировать тестовый контур

ozon fintech

Версионирование



Пополнение из банкомата

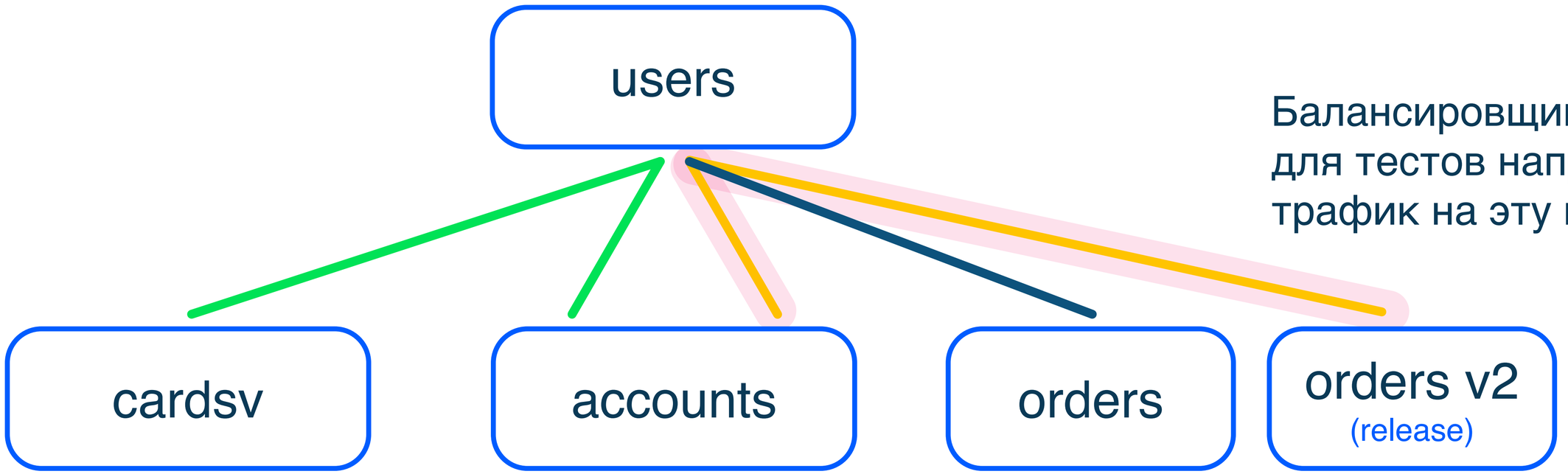
В метаданные: указать версию сервиса

Заказ товара (use release version)

Стабилизировать тестовый контур

ozon fintech

Версионирование



Балансировщик
для тестов направит
трафик на эту версию

В метаданные: указать версию сервиса

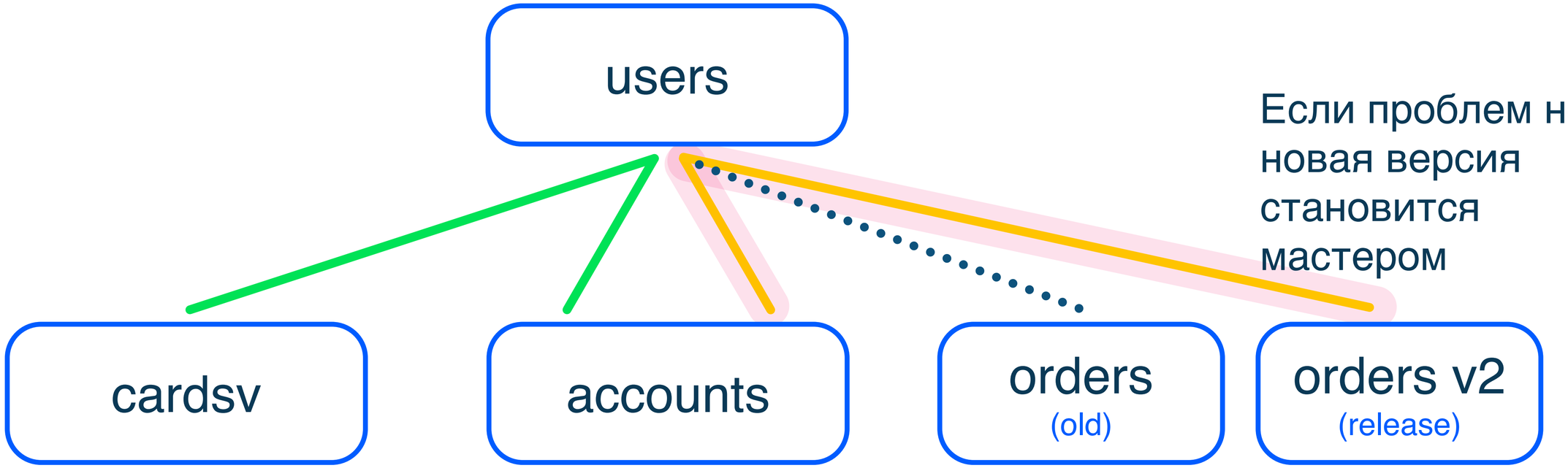
Пополнение из банкомата

Заказ товара (use release version)

Стабилизировать тестовый контур

ozon fintech

Версионирование



Если проблем нет — новая версия становится мастером

В метаданные: указать версию сервиса

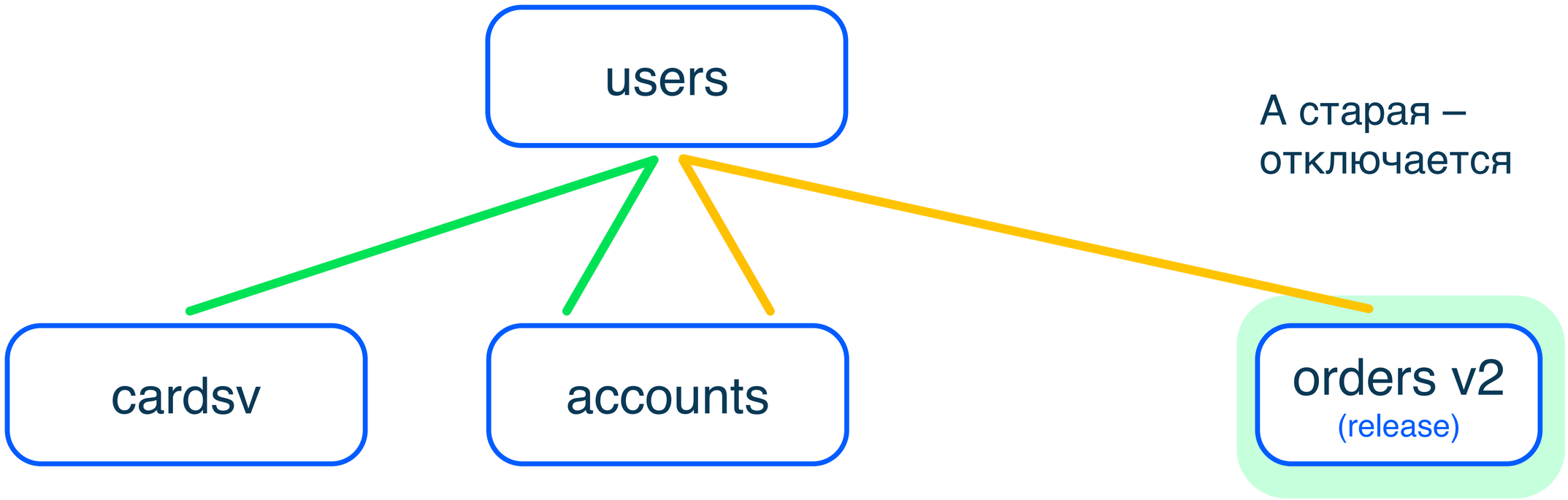
Пополнение из банкомата

Заказ товара (use release version)

Стабилизировать тестовый контур

ozon fintech

Версионирование



А старая –
отключается

Пополнение из банкомата

Заказ товара

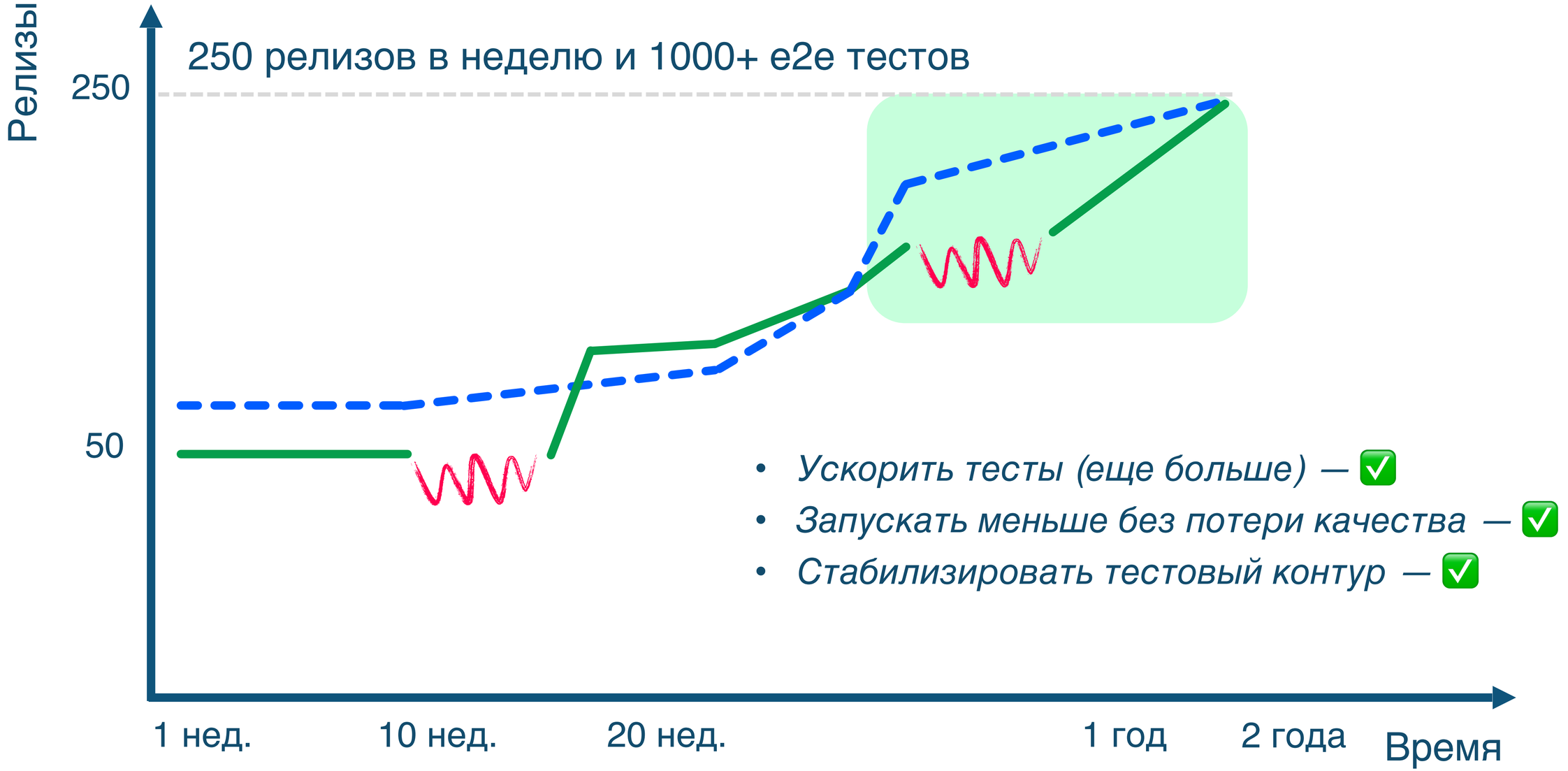
ozon fintech

Версионирование

Проблемные релизы
не ломают тесты и
не влияют на другие релизы

Итоги

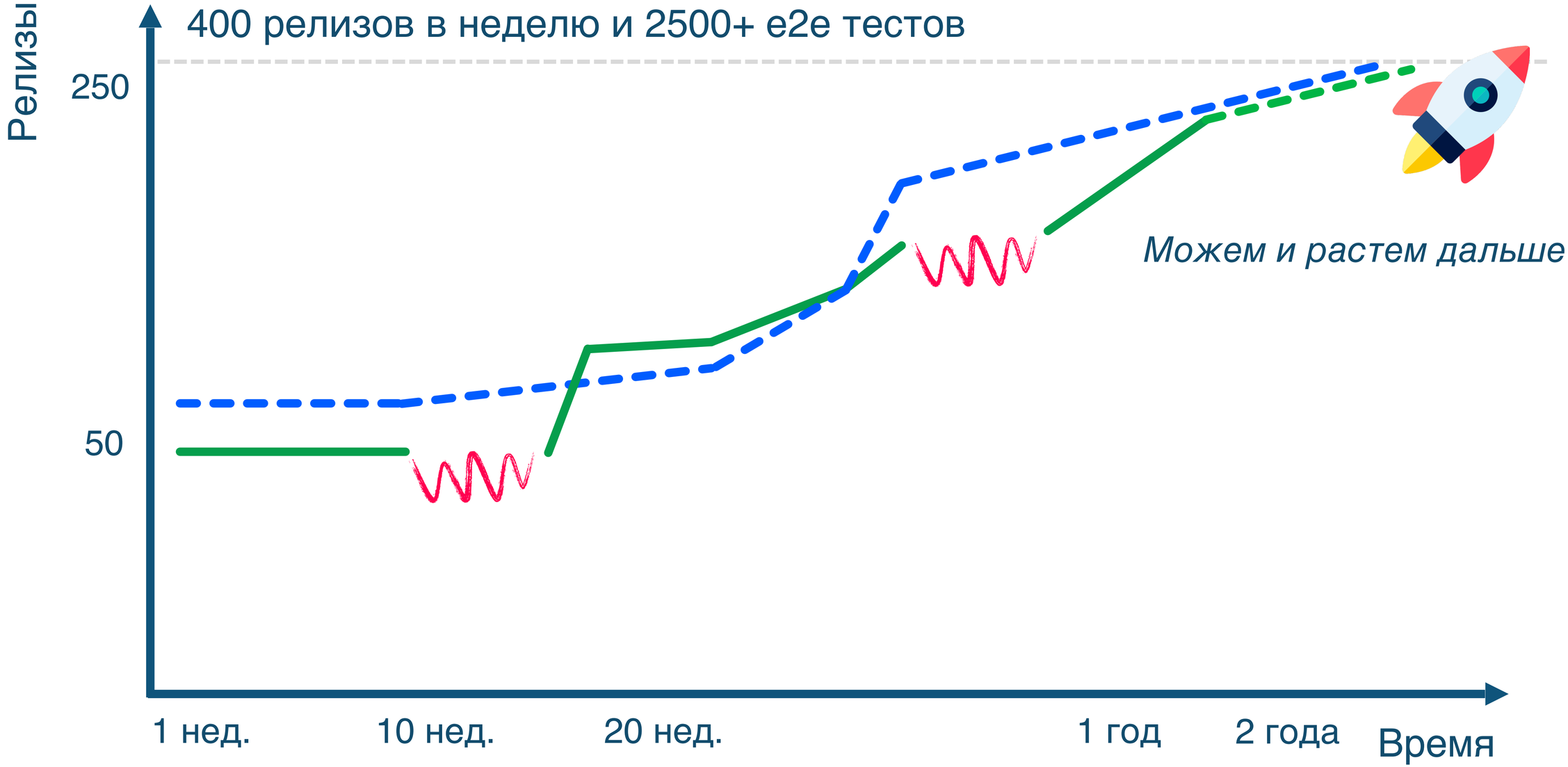
ozon fintech



- *Ускорить тесты (еще больше)* — ✅
- *Запускать меньше без потери качества* — ✅
- *Стабилизировать тестовый контур* — ✅

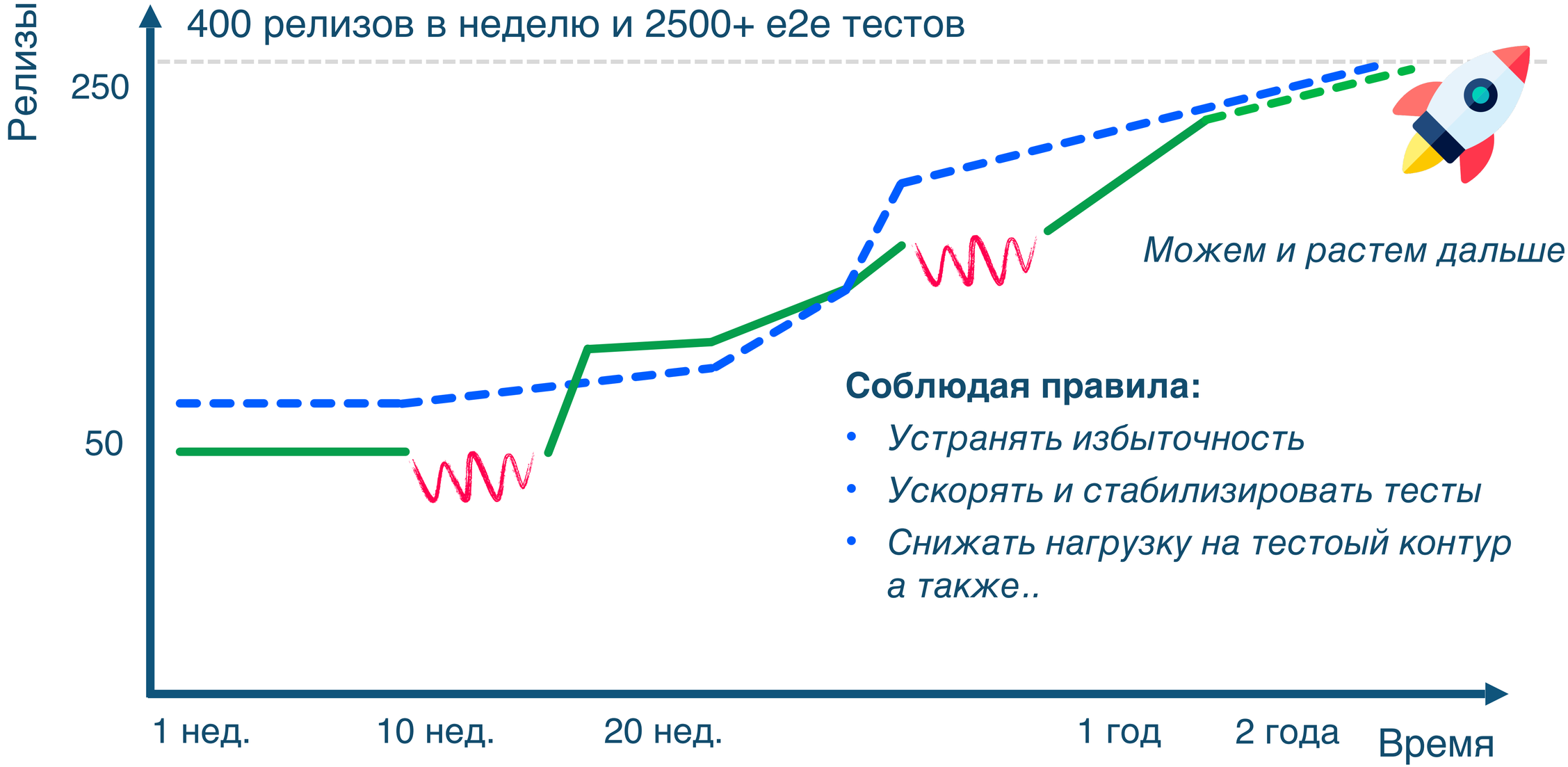
Дальше

ozon fintech



Дальше

ozon fintech



Можем и растем дальше

Соблюдая правила:

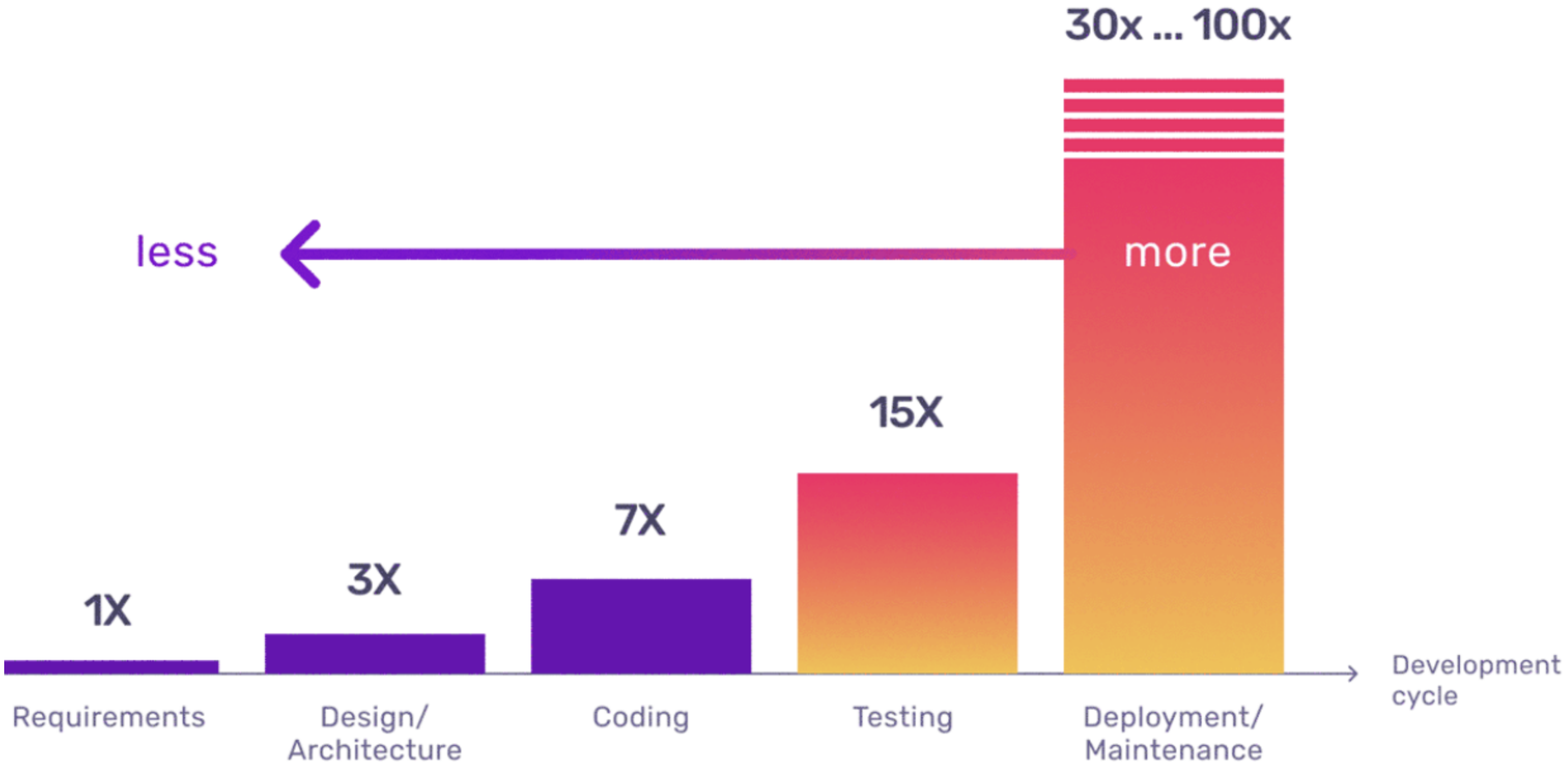
- *Устранять избыточность*
- *Ускорять и стабилизировать тесты*
- *Снижать нагрузку на тестоый контур*
  а также..

Реагировать
на более ранних этапах

Не менее актуальное

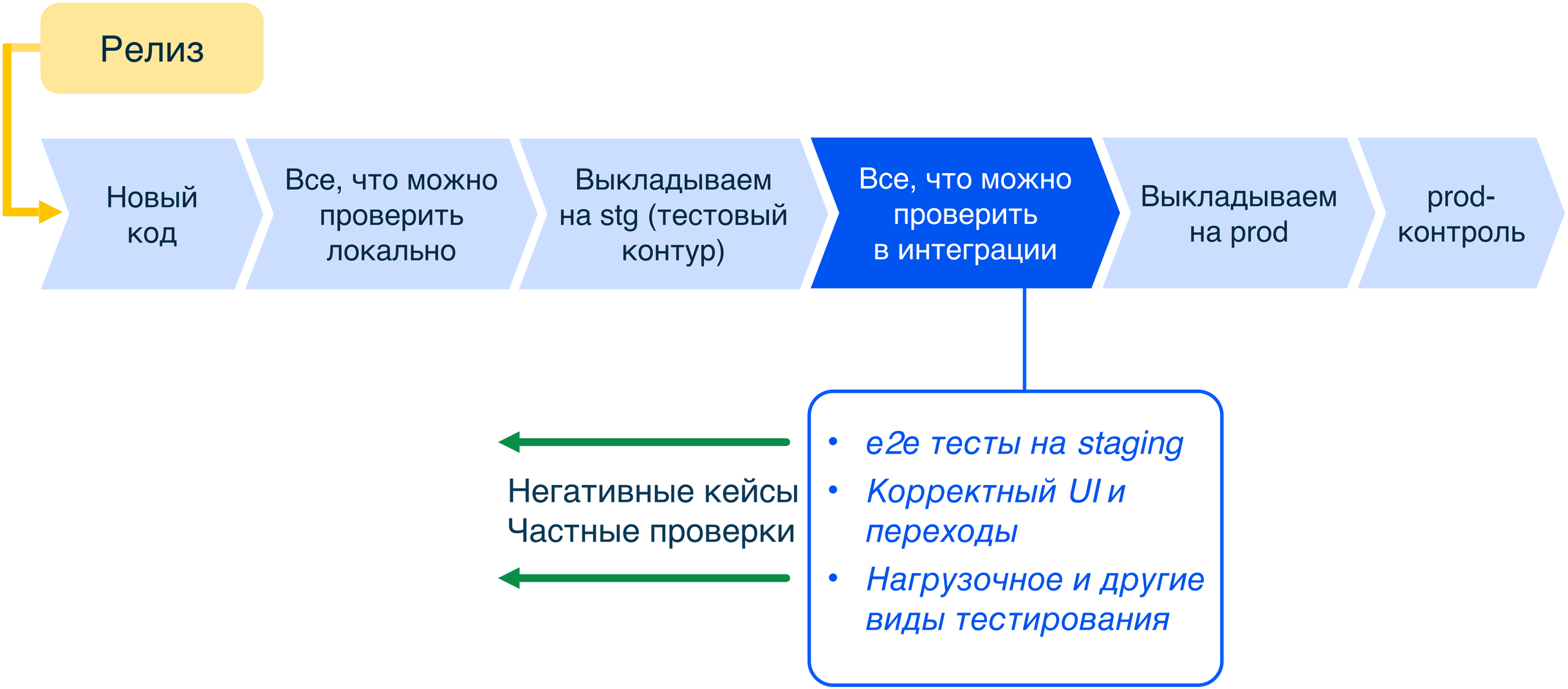
ozon fintech

Реагировать на более ранних этапах



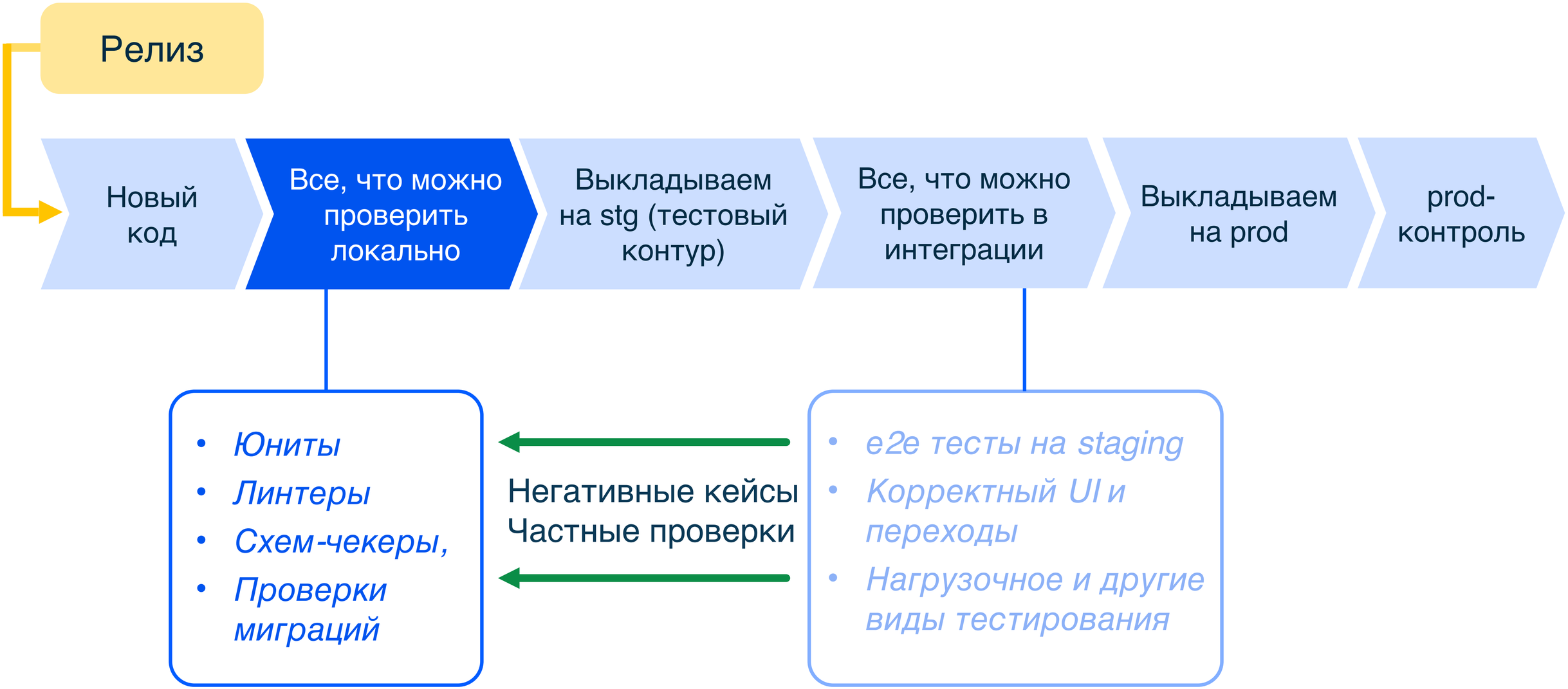
Много активности из области e2e

ozon fintech



В область ранних проверок

ozon fintech



Область ранних проверок

ozon fintech

Есть интересные доклады.
Что в них слышал:

- Тесты на бд
- Компонентные
- Генерация автотестов
- Моки и другие фокусы

…

Доклад Станислава Башкирцева на тему «Как разработчику научиться строить тестовую пирамиду»

Область ранних проверок

ozon fintech

Чего не услышал:

- Понятно и легко держать высокое покрытие в ситуациях:
  - Сервис с нуля
  - Слияние нескольких сервисов/дербан монолита
  - legacy-проект с кучей хендлеров и без тестов
- Как оценивать покрытие и находить критические пробелы в нем
- Как сделать эти процессы понятные многим

Ситуация: в крупной компании

ozon fintech

Что там по низкоуровневым тестам?

Вроде пишутся.

ловим типичный баг валидации

Ты же сказал, что тесты пишутся…

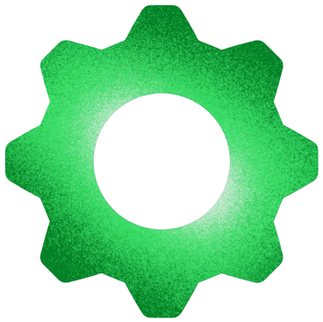
Я сказал «вроде пишутся».

Много микросервисов

(от 100 до 300 на направлении)

ozon fintech

- Какое покрытие на каждом?
- Покрыты ли критичные места в коде/логике?
- Улучшилось ли покрытие у сервиса за полгода?
- А у команды улучшилось ли покрытие?



Так много вопросов…

ozon fintech

Нашли и отточили решения и подходы,
чтобы ответить на все эти вопросы

Как мы действовали

ozon fintech

1. Как добиться большего покрытия меньшими усилиями
2. Какое покрытие на конкретном сервисе
3. Покрыты ли критичные места в коде/логике
4. Что покрывать в первую очередь
5. Улучшилось ли покрытие у сервиса за полгода
6. Улучшила ли команда покрытие за полгода

Хорошее покрытие

ozon fintech

Малыми усилиями

- Нужны новые тесты. Какие?
- Типизированные тесты такого типа могут считаться заменой e2e
- Высвобождает ресурсы как тестировщиков так и разработчиков
- Повышает доверие

Решение – лучшее от e2e и unit-тестов

ozon fintech

1 Давай возьмем сильные стороны тестов от каждого уровня (**unit** и **e2e**)

unit	e2e
Быстрые и независимые.	Один тест дает проверку многих компонентов продукта.
Дают реальную метрику покрытия в %.	То самое регрессионное тестирование, которое показывает, что продукт не деградирует.
Не зависят от контура / зависят только от собственных данных в БД.	Может описываться реальный сценарий пользования продуктом.

Реализация: сервис

ozon fintech

Инициализация

```
func NewUserService(ctx, extClients) *Implementation{
    storage := repo.InitStorage(config.GetValue("pg_dsn"))
   nicePkg := pkg.GetHelpers()
    return NewUserImplementation(storage, nicePkg, extClients)
}
```

Логика контрактов

```
func (i *Implementation) CreatePerson(req..)&model.CreatePersonResponse {...}
func (i *Implementation) GetPerson(req..)&model.GetPersonResponse {...}
func (i *Implementation) UpdatePerson(req..)&model.UpdatePersonResponse {...}
func (i *Implementation) DeletePerson(req..)&model.DeletePersonResponse {...}
```

swagger-контракты

Реализация: тест

ozon fintech

```
func NewUserService(ctx, extClients) *Implementation{
    storage := repo.InitStorage(config.GetValue("pg_dsn"))
   nicePkg := pkg.GetHelpers()
    return NewUserImplementation(storage, nicePkg, extClients)
}
```

```
func TestCreatePerson(t *testing.T) {
    db := InitDB("TEST_DSN")
    extClientMocks := NewExtClientMock(controller)
  app := NewUserService(
            storage, nicePackage, externalClientMock)
  res, err := app.CreatePerson("Vasya")
  require.NoError(t, err)
  require.NotNil(t, res.ID)
}
```

Локальная база данных

Реализация: тест

ozon fintech

```
func NewUserService(ctx, extClients) *Implementation{
    storage := repo.InitStorage(config.GetValue("pg_dsn"))
   nicePkg := pkg.GetHelpers()
    return NewUserImplementation(storage, nicePkg, extClients)
}
```

```
func TestCreatePerson(t *testing.T) {
    db := InitDB("TEST_DSN")
    extClientMocks := NewExtClientMock(controller)
  app := NewUserService(
            storage, nicePackage, externalClientMock)
  res, err := app.CreatePerson("Vasya")
  require.NoError(t, err)
  require.NotNil(t, res.ID)
}
```

Локальная база данных

Мок внешних вызовов

Реализация: тест

ozon fintech

```
func NewUserService(ctx, extClients) *Implementation{
    storage := repo.InitStorage(config.GetValue("pg_dsn"))
   nicePkg := pkg.GetHelpers()
    return NewUserImplementation(storage, nicePkg, extClients)
}
```

```
func TestCreatePerson(t *testing.T) {
    db := InitDB("TEST_DSN")
    extClientMocks := NewExtClientMock(controller)
  app := NewUserService(
            storage, nicePackage, externalClientMock)
  res, err := app.CreatePerson("Vasya")
  require.NoError(t, err)
  require.NotNil(t, res.ID)
}
```

- Локальная база данных
- Мок внешних вызовов
- Инициализация сервиса как объект

Реализация: тест

ozon fintech

```
func NewUserService(ctx, extClients) *Implementation{
    storage := repo.InitStorage(config.GetValue("pg_dsn"))
   nicePkg := pkg.GetHelpers()
    return NewUserImplementation(storage, nicePkg, extClients)
}
```

```
func TestCreatePerson(t *testing.T) {
    db := InitDB("TEST_DSN")
    extClientMocks := NewExtClientMock(controller)
  app := NewUserService(
            storage, nicePackage, externalClientMock)
  res, err := app.CreatePerson("Vasya")
  require.NoError(t, err)
  require.NotNil(t, res.ID)
}
```

- Локальная база данных
- Мок внешних вызовов
- Инициализация сервиса как объект
- Вызов метода и тесты

POST /person Add a new person to the store

Запуск теста

ozon fintech

- Запускаем тесты **с инструментами оценки покрытия**

  go test ./... -covermode atomic -coverprofile=cover.out

Запуск теста

ozon fintech

- Запускаем тесты **с инструментами оценки покрытия**

  go test ./... -covermode atomic -coverprofile=cover.out

- У нас есть результат — артефакт **cover.out**

Запуск теста

ozon fintech

- Запускаем тесты **с инструментами оценки покрытия**

  go test ./... -covermode atomic -coverprofile=cover.out

- У нас есть результат — артефакт **cover.out**
- Сформируем **тестовый отчет**

  go tool cover -func=cover.out

Запуск теста

ozon fintech

- Запускаем тесты **с инструментами оценки покрытия**

  go test ./... -covermode atomic -coverprofile=cover.out

- У нас есть результат — артефакт **cover.out**
- Сформируем **тестовый отчет**

  go tool cover -func=cover.out

И получим:

```
./internal/app/userService/create_person.go:23:     CreatePerson                 59.0%
./internal/app/userService/create_person.go:78:     validateCreatePersonRequest  100.0%
./internal/app/userService/create_person.go:91:     createPersonInDB             75.0%
./internal/pkg/store/person_store.go:304:           CreatePerson                 75.0%
... total 60%
```

Запуск теста

ozon fintech

- Запускаем тесты **с инструментами оценки покрытия**

  go test ./... -covermode atomic -coverprofile=cover.out

- У нас есть результат — артефакт **cover.out**
- Сформируем **тестовый отчет**

  go tool cover -func=cover.out

И получим:

```
./internal/app/userService/create_person.go:23:     CreatePerson                 59.0%
./internal/app/userService/create_person.go:78:     validateCreatePersonRequest  100.0%
./internal/app/userService/create_person.go:91:     createPersonInDB             75.0%
./internal/pkg/store/person_store.go:304:           CreatePerson                 75.0%
... total 60%
```

И у нас есть суммарное покрытие

Запуск теста

ozon fintech

- Запускаем тесты **с инструментами оценки покрытия**

  go test ./... -covermode atomic -coverprofile=cover.out

- У нас есть результат — артефакт **cover.out**
- Сформируем **тестовый отчет**

  go tool cover -func=cover.out

И получим:

```
./internal/app/userService/create_person.go:23:
./internal/app/userService/create_person.go:78:
./internal/app/userService/create_person.go:91:
./internal/pkg/store/person_store.go:304:
... total 60%
```

И у нас есть суммарное покрытие

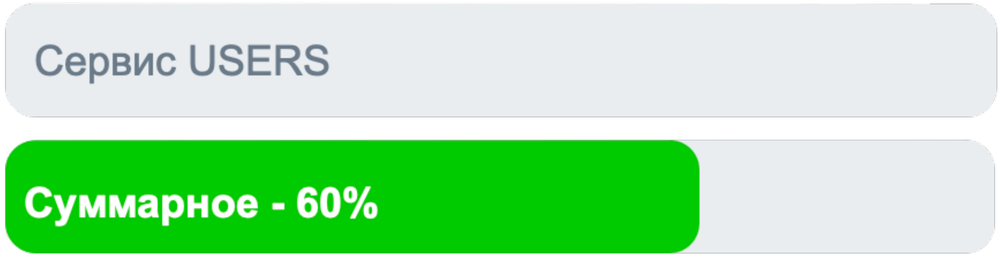
И покрытие хэндлера:

```
CreatePerson                    59.0%
validateCreatePersonRequest    100.0%
createPersonInDB                75.0%
CreatePerson                    75.0%
```

Почему эти цифры? CreatePerson 59.0% и total 60%

ozon fintech

О чем говорит % общего покрытия

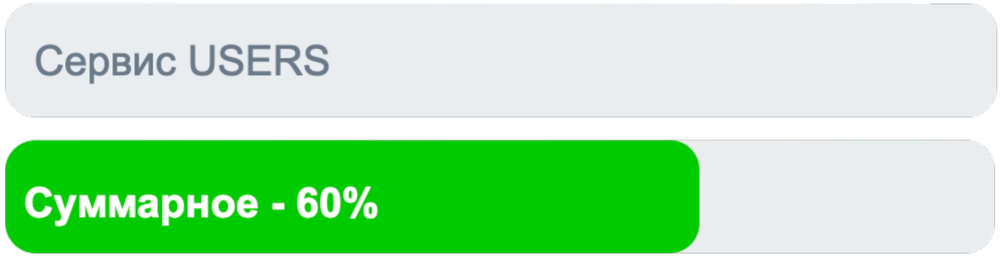


Хорошо / плохо
Стало лучше / стало хуже

Почему эти цифры? CreatePerson 59.0% и total 60%

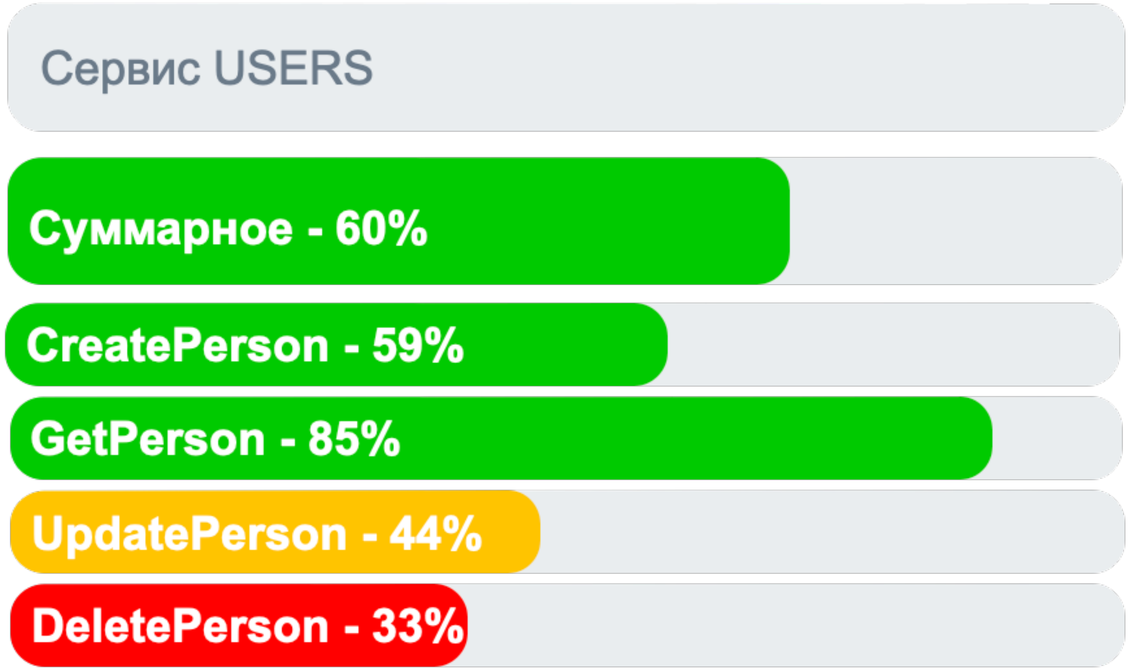
ozon fintech

О чем говорит % общего покрытия



Хорошо / плохо
Стало лучше / стало хуже

А что если...так



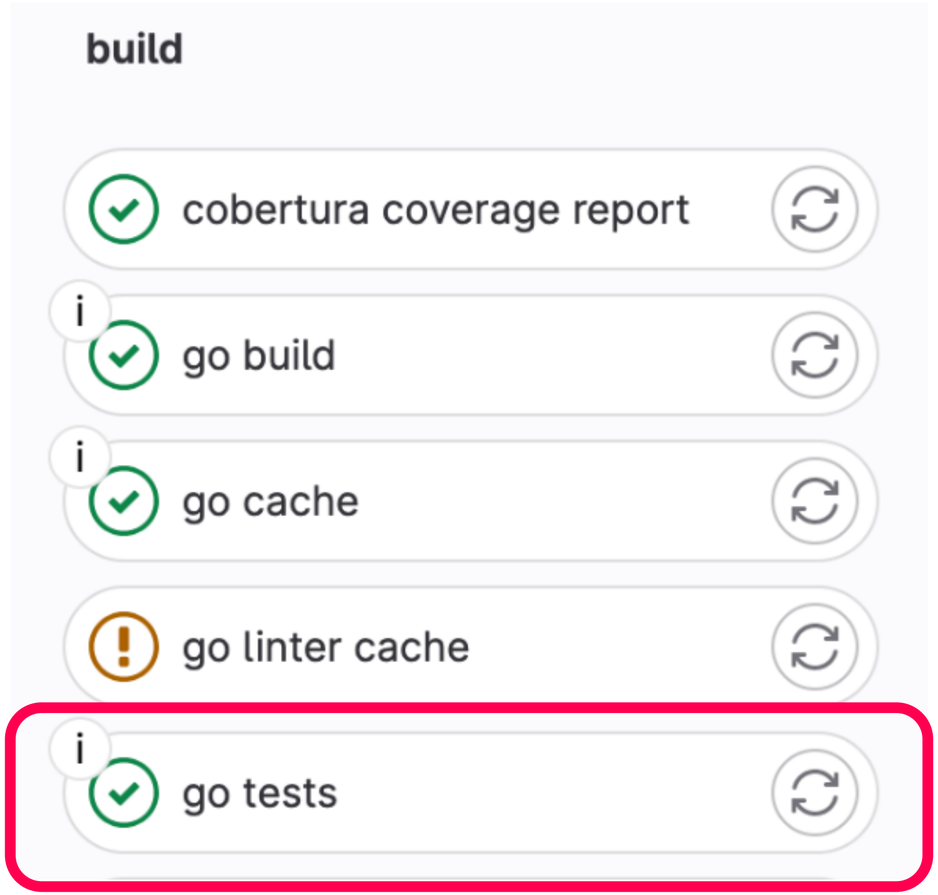
Итог

ozon fintech

- Один приличный тест с инфраструктурой -> много покрытия
- Артефакт с числом покрытия сервиса и его хэндлера

CreatePerson 59.0%
…total 60%

- И мы всегда можем посмотреть эти значения при запуске в ci/cd



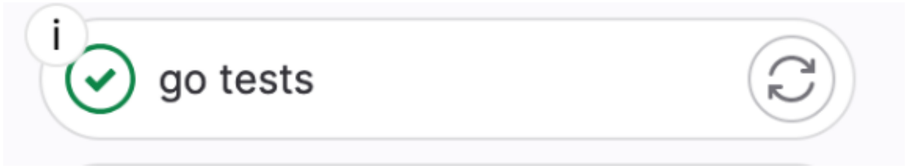
Как мы действовали

ozon fintech

1. Как добиться большего покрытия меньшими усилиями
2. Какое покрытие на конкретном сервисе
3. Покрыты ли критичные места в коде/логике
4. Что покрывать в первую очередь
5. Улучшилось ли покрытие у сервиса за полгода
6. Улучшила ли команда покрытие за полгода

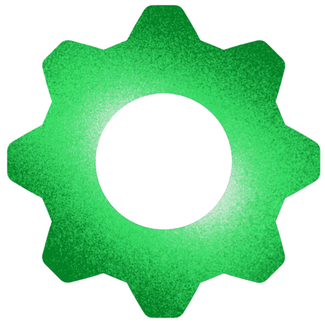
Смотрим покрытие нескольких сервисов

ozon fintech



CreatePerson 59.0% …total 60%

- Вручную ходить по репозиториям и пайплайнам – долго
- Была выгрузка в Grafana – неудобно
- Решили для удобства сохранять цифры о покрытии (для этого пришлось написать еще один сервис)

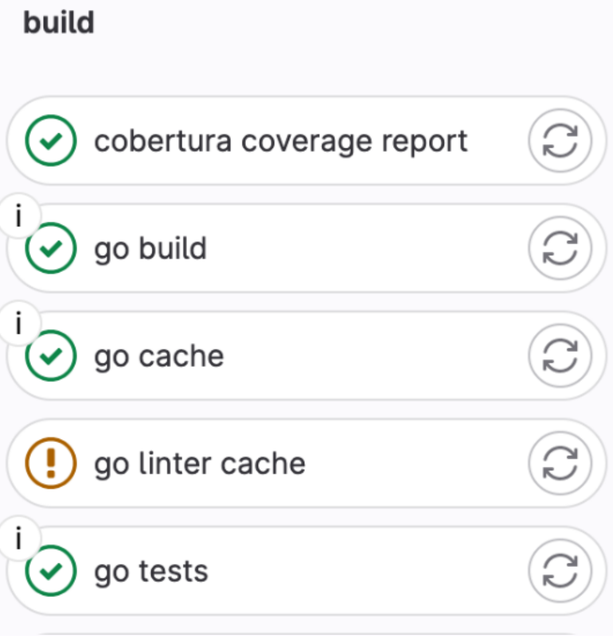


Цифры для отчета

CreatePerson 59.0% и **total 60%**

ozon fintech

Распознаем



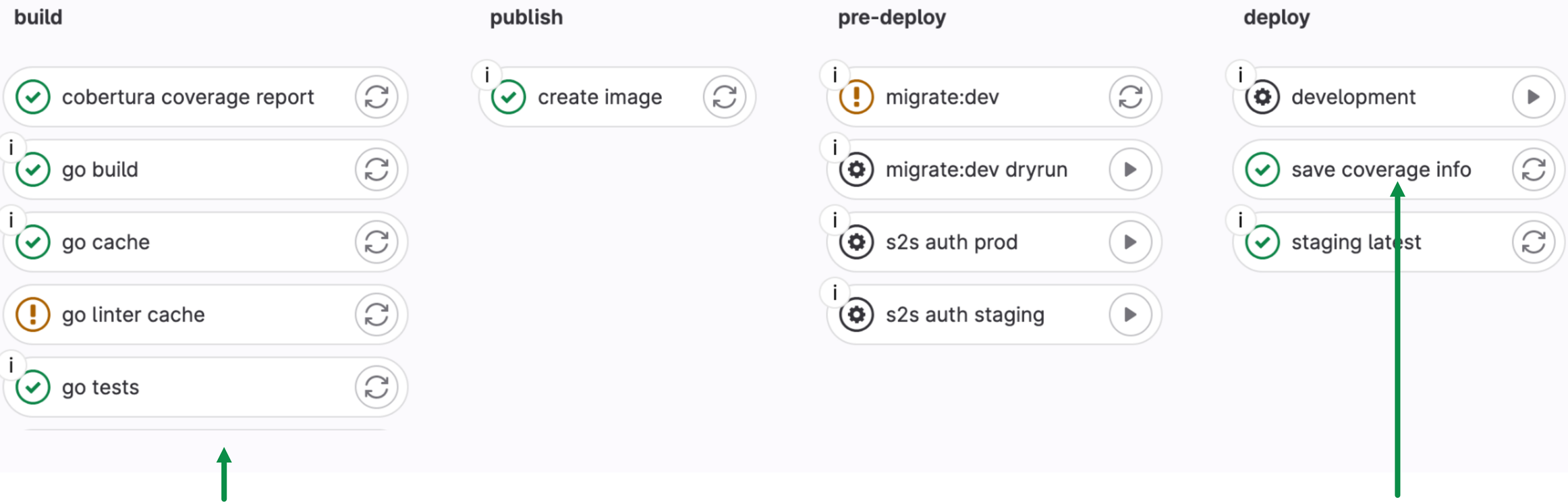
Получаем cover.out

Цифры для отчета

CreatePerson 59.0% и **total 60%**

ozon fintech

Распознаем



Получаем cover.out

Парсим cover.out
отправляем
в хранилище

Сервис для информации о покрытии

Команда — **fintech_sdet_team**

Владелец — **Егор Стручин**

- Сервис go (grpc/http) + postgres + UI

Общие сведения

Tracked	Untracked	покрытие < 20%	покрытие 20 - 40 %	Кол-во сервисов
9	**0**	**0**	**0**	**9**

Сервис ↓	Покрытие ↓	Цель	Изменения	Информация по хэндлерам	
hive-ng gitlab ITC Подготовка тестовых пользователей для банка (только стейджинг) генерируются пользователи и сущности со случайными данными, любое совпадение с реальными данными исключено	76.40 %	75.00 % до: 2023.10.01	Неделя: -0.20 % Месяц: +0.30 % 3 месяца: +5.00 % С 01.04.2023: +24.20 %	23 handlers	2 bad coverage
optiqa gitlab ITC сервис настройки e2e тестов длительность названия тестов схема работы tracedurationconfigoptimisation	73.20 %	70.00 % до: 2023.10.01	Неделя: +0.30 % Месяц: +3.80 % 3 месяца: +8.30 % С 01.04.2023: +35.30 %	26 handlers	5 bad coverage
phantom-ng gitlab ITC Сервис для подмены внешних вызовов и реализации заглушек, имитирующих функционал внешних сервисов. Мок-сервисы: НСПК, TSYS, Clarke	71.00 %	75.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: 0.00 % С 01.04.2023: +5.10 %	16 handlers	4 bad coverage
phantom-nspk gitlab ITC Имитатор НСПК (СБП платежи)	77.70 %	80.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: +0.20 % С 01.04.2023: +2.70 %	14 handlers	1 bad coverage

- Сервис go (grpc/http) + postgres + UI
- Записи покрытия

Команда — fintech_sdet_team

Владелец — **Егор Стручин**

Общие сведения

Tracked	Untracked	покрытие < 20%	покрытие 20 - 40 %	Кол-во сервисов
9	0	0	0	9

Сервис	Покрытие	Цель	Изменения	Информация по хэндлерам	
hive-ng gitlab ITC Подготовка тестовых пользователей для банка (только стейджинг) генерируются пользователи и сущности со случайными данными, любое совпадение с реальными данными исключено	76.40 %	75.00 % до: 2023.10.01	Неделя: -0.20 % Месяц: +0.30 % 3 месяца: +5.00 % С 01.04.2023: +24.20 %	23 handlers	2 bad coverage
optiqa gitlab ITC сервис настройки e2e тестов длительность названия тестов схема работы tracedurationconfigoptimisation	73.20 %	70.00 % до: 2023.10.01	Неделя: +0.30 % Месяц: +3.80 % 3 месяца: +8.30 % С 01.04.2023: +35.30 %	26 handlers	5 bad coverage
phantom-ng gitlab ITC Сервис для подмены внешних вызовов и реализации заглушек, имитирующих функционал внешних сервисов. Мок-сервисы: НСПК, TSYS, Clarke	71.00 %	75.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: 0.00 % С 01.04.2023: +5.10 %	16 handlers	4 bad coverage
phantom-nspk gitlab ITC Имитатор НСПК (СБП платежи)	77.70 %	80.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: +0.20 % С 01.04.2023: +2.70 %	14 handlers	1 bad coverage

Команда — **fintech_sdet_team**

Владелец — **Егор Стручин**

- Сервис go (grpc/http) + postgres + UI
- Записи покрытия+историю

Общие сведения

Tracked	Untracked	покрытие < 20%	покрытие 20 - 40 %	Кол-во сервисов
9	**0**	**0**	**0**	**9**

Сервис ↓	Покрытие ↓	Цель	Изменения	Информация по хэндлерам	
hive-ng gitlab ITC Подготовка тестовых пользователей для банка (только стейджинг) генерируются пользователи и сущности со случайными данными, любое совпадение с реальными данными исключено	76.40 %	75.00 % до: 2023.10.01	Неделя: -0.20 % Месяц: +0.30 % 3 месяца: +5.00 % С 01.04.2023: +24.20 %	23 handlers	2 bad coverage
optiqa gitlab ITC сервис настройки e2e тестов длительность названия тестов схема работы tracedurationconfigoptimisation	73.20 %	70.00 % до: 2023.10.01	Неделя: +0.30 % Месяц: +3.80 % 3 месяца: +8.30 % С 01.04.2023: +35.30 %	26 handlers	5 bad coverage
phantom-ng gitlab ITC Сервис для подмены внешних вызовов и реализации заглушек, имитирующих функционал внешних сервисов. Мок-сервисы: НСПК, TSYS, Clarke	71.00 %	75.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: 0.00 % С 01.04.2023: +5.10 %	16 handlers	4 bad coverage
phantom-nspk gitlab ITC Имитатор НСПК (СБП платежи)	77.70 %	80.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: +0.20 % С 01.04.2023: +2.70 %	14 handlers	1 bad coverage

Команда — **fintech_sdet_team**

Владелец — **Егор Стручин**

- Сервис go (grpc/http) + postgres + UI
- Записи покрытия+историю
- Структура команд и summary

Общие сведения

Tracked	Untracked	покрытие < 20%	покрытие 20 - 40 %	Кол-во сервисов
9	0	0	0	9

Сервис ↓	Покрытие ↓	Цель	Изменения	Информация по хэндлерам	
hive-ng gitlab ITC Подготовка тестовых пользователей для банка (только стейджинг) генерируются пользователи и сущности со случайными данными, любое совпадение с реальными данными исключено	76.40 %	75.00 % до: 2023.10.01	Неделя: -0.20 % Месяц: +0.30 % 3 месяца: +5.00 % С 01.04.2023: +24.20 %	23 handlers	2 bad coverage
optiqa gitlab ITC сервис настройки e2e тестов длительность названия тестов схема работы traceduration config optimisation	73.20 %	70.00 % до: 2023.10.01	Неделя: +0.30 % Месяц: +3.80 % 3 месяца: +8.30 % С 01.04.2023: +35.30 %	26 handlers	5 bad coverage
phantom-ng gitlab ITC Сервис для подмены внешних вызовов и реализации заглушек, имитирующих функционал внешних сервисов. Мок-сервисы: НСПК, TSYS, Clarke	71.00 %	75.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: 0.00 % С 01.04.2023: +5.10 %	16 handlers	4 bad coverage
phantom-nspk gitlab ITC Имитатор НСПК (СБП платежи)	77.70 %	80.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: +0.20 % С 01.04.2023: +2.70 %	14 handlers	1 bad coverage

Команда — fintech_sdet_team

Владелец — **Егор Стручин**

Общие сведения

Tracked	Untracked	покрытие < 20%	покрытие 20 - 40 %	Кол-во сервисов
9	0	0	0	9

Сервис ↓	Покрытие ↓	Цель	Изменения	Информация по хэндлерам	
hive-ng gitlab ITC Подготовка тестовых пользователей для банка (только стейджинг) генерируются пользователи и сущности со случайными данными, любое совпадение с реальными данными исключено	76.40 %	75.00 % до: 2023.10.01	Неделя: -0.20 % Месяц: +0.30 % 3 месяца: +5.00 % С 01.04.2023: +24.20 %	23 handlers	2 bad coverage
optiqa gitlab ITC сервис настройки e2e тестов длительность названия тестов схема работы tracedurationconfigoptimisation	73.20 %	70.00 % до: 2023.10.01	Неделя: +0.30 % Месяц: +3.80 % 3 месяца: +8.30 % С 01.04.2023: +35.30 %	26 handlers	5 bad coverage
phantom-ng gitlab ITC Сервис для подмены внешних вызовов и реализации заглушек, имитирующих функционал внешних сервисов. Мок-сервисы: НСПК, TSYS, Clarke	71.00 %	75.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: 0.00 % С 01.04.2023: +5.10 %	16 handlers	4 bad coverage
phantom-nspk gitlab ITC Имитатор НСПК (СБП платежи)	77.70 %	80.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: +0.20 % С 01.04.2023: +2.70 %	14 handlers	1 bad coverage

- Сервис go (grpc/http) + postgres + UI
- Записи покрытия+историю
- Структура команд и summary
- Полезные ссылки

Команда — fintech_sdet_team

Владелец — **Егор Стручин**

Общие сведения

Tracked	Untracked	покрытие < 20%	покрытие 20 - 40 %	Кол-во сервисов
9	0	0	0	9

Сервис	Покрытие	Цель	Изменения	Информация по хэндлерам	
hive-ng gitlab ITC Подготовка тестовых пользователей для банка (только стейджинг) генерируются пользователи и сущности со случайными данными, любое совпадение с реальными данными исключено	76.40 %	75.00 % до: 2023.10.01	Неделя: -0.20 % Месяц: +0.30 % 3 месяца: +5.00 % С 01.04.2023: +24.20 %	23 handlers	2 bad coverage
optiqa gitlab ITC сервис настройки e2e тестов длительность названия тестов схема работы tracedurationconfigoptimisation	73.20 %	70.00 % до: 2023.10.01	Неделя: +0.30 % Месяц: +3.80 % 3 месяца: +8.30 % С 01.04.2023: +35.30 %	26 handlers	5 bad coverage
phantom-ng gitlab ITC Сервис для подмены внешних вызовов и реализации заглушек, имитирующих функционал внешних сервисов. Мок-сервисы: НСПК, TSYS, Clarke	71.00 %	75.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: 0.00 % С 01.04.2023: +5.10 %	16 handlers	4 bad coverage
phantom-nspk gitlab ITC Имитатор НСПК (СБП платежи)	77.70 %	80.00 % до: 2023.10.01	Неделя: 0.00 % Месяц: 0.00 % 3 месяца: +0.20 % С 01.04.2023: +2.70 %	14 handlers	1 bad coverage

- Сервис go (grpc/http) + postgres + UI
- Записи покрытия+историю
- Структура команд и summary
- Полезные ссылки
- Цели по покрытию

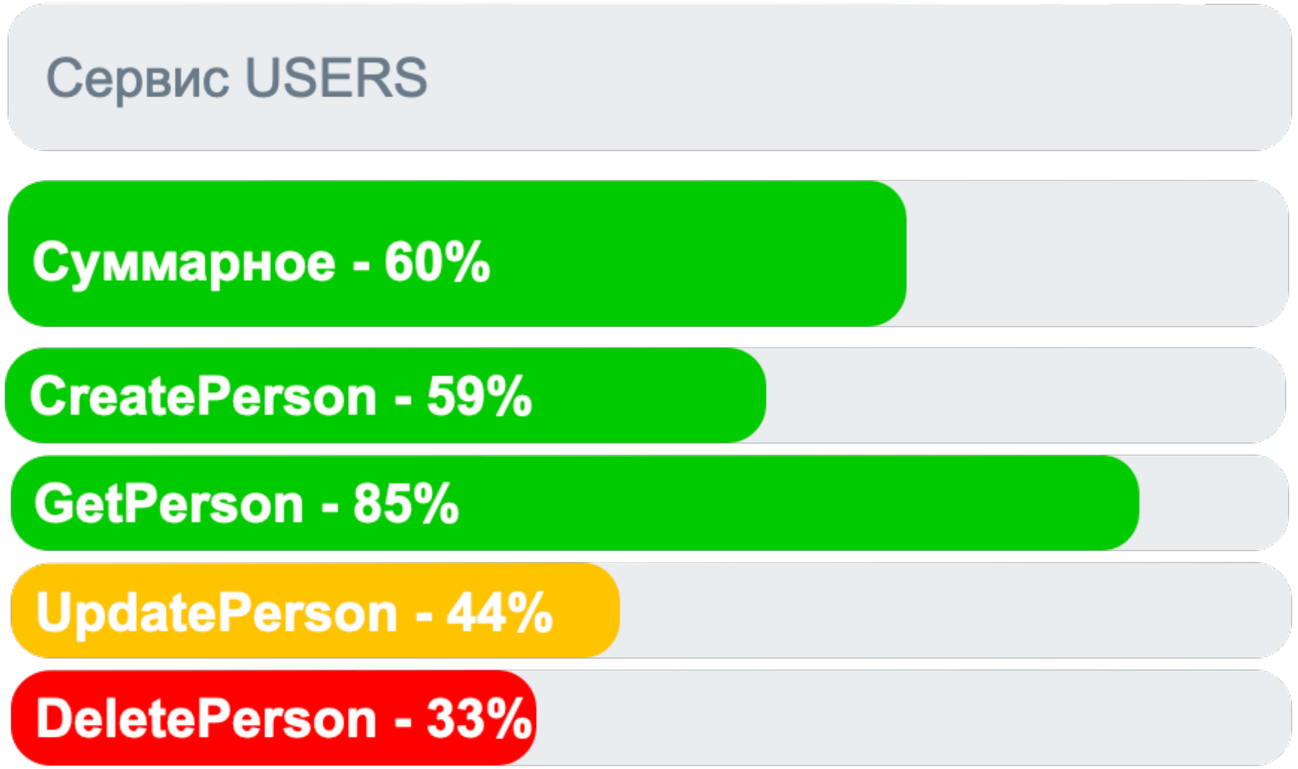
Как мы действовали

ozon fintech

1. Как добиться большего покрытия меньшими усилиями
2. Какое покрытие на конкретном сервисе
3. Покрыты ли критичные места в коде/логике
4. Что покрывать в первую очередь
5. Улучшилось ли покрытие у сервиса за полгода
6. Улучшила ли команда покрытие за полгода

Оценивать покрытие, находить критические пробелы

ozon fintech



Цифры – ориентир!

Проверим, а **что** мы проверяем

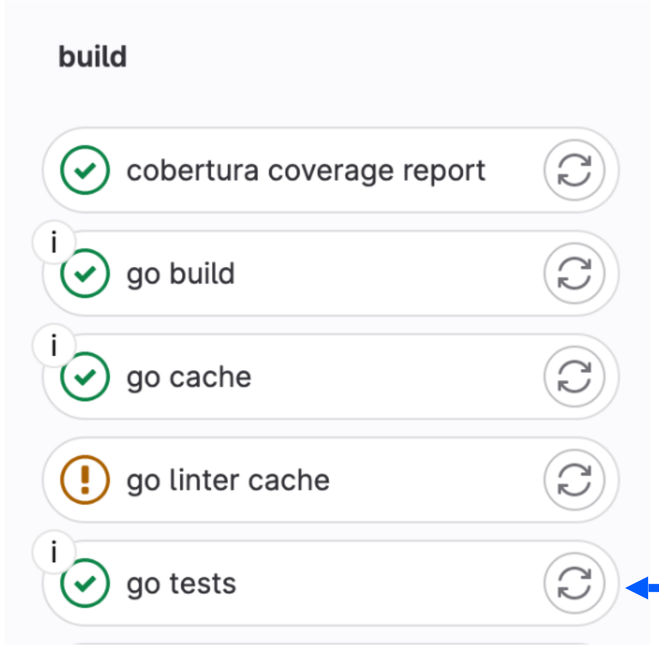
(под пристальной оценкой тестировщика и разработчика)

Визуализируй - понимай

ozon fintech

cover.out отсюда

CreatePerson 59.0%
total 60%

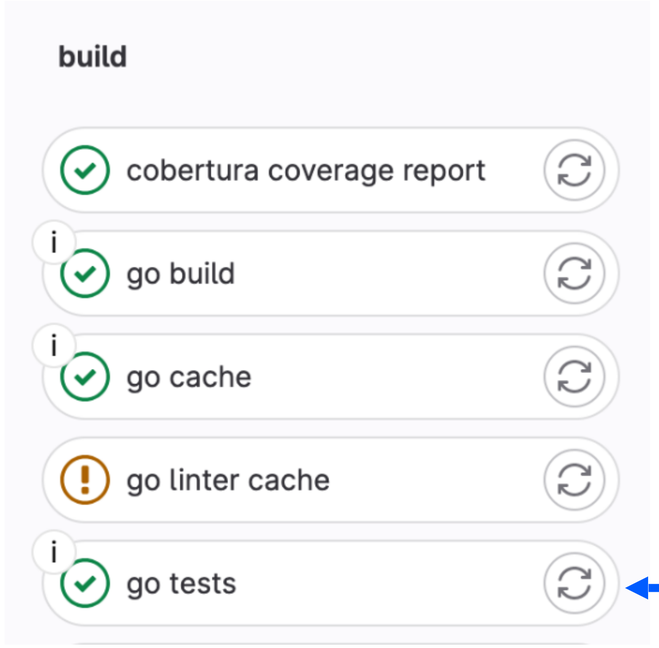


Визуализируй - понимай

ozon fintech

cover.out отсюда -> go tool cover -html cover.out -o cover.html

CreatePerson 59.0%
total 60%

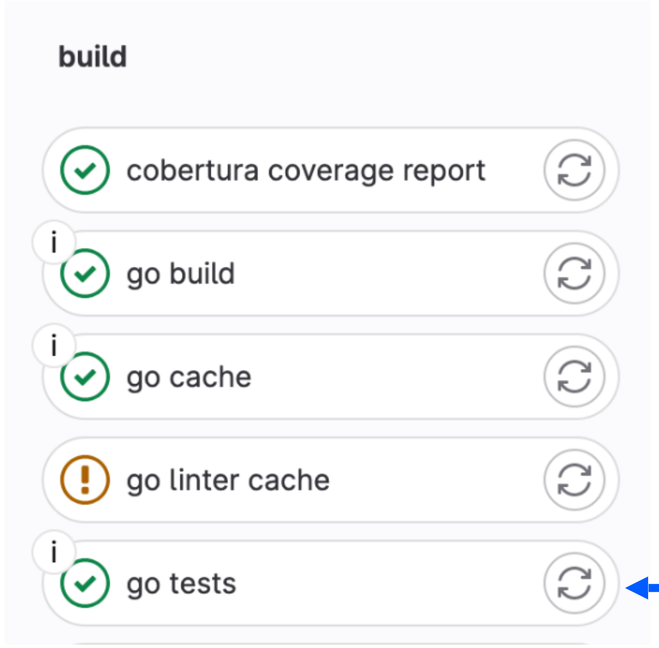


Визуализируй - понимай

ozon fintech

cover.out отсюда -> go tool cover -html cover.out -o cover.html

CreatePerson 59.0%
total 60%



/internal/app/ı ı/create_person.go (63.3%) not tracked no coverage low coverage * *

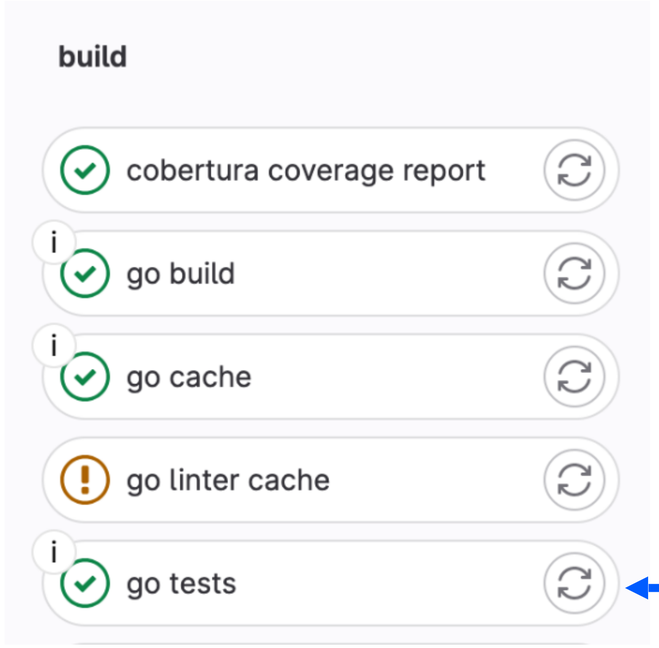
```go
// CreatePerson .
func (i *Implementation) CreatePerson(ctx context.Context, req *desc.CreatePersonRequest) (*desc.Person, error) {
	err := validateCreatePersonRequest(req)
	if err != nil {
		return nil, status.Errorf(codes.InvalidArgument, "Validation failed: %s", err.Error())
	}

	phone, err := utils.NormalizePhone(req.Phone)
	if err != nil {
		return nil, status.Errorf(codes.InvalidArgument, "Validation phone failed: %s", err.Error())
	}
	var email string
	if req.Email != nil {
		email, err = utils.NormalizeEmail(*req.Email)
		if err != nil {
			return nil, status.Errorf(codes.InvalidArgument, "Validation email failed: %s", err.Error())
		}
	}

	person := &model.Person{
		OzonID:              zero.IntFrom(req.OzonId),
		Phone:               zero.StringFrom(phone),
		Email:               zero.StringFrom(email),
		IdentificationLevel: desc.IdentificationLevel_ANON,
		Source:              desc.Person_ECOM_BANK,
		Bank:                desc.Bank_ECOM,
		DataLevel:           desc.IdentificationLevel_ANON,
	}

	err = i.storage.WithTransaction(ctx, func(tx sql.Transaction) error {
		err = createPersonInDB(ctx, tx, person, req.Reason, i)
		return nil
	})

	switch {
	case err == nil:
	case errors.Is(err, store.ErrEntityAlreadyExistByPhone):
```

Визуализируй-понимай

ozon fintech

cover.out отсюда -> go tool cover -html cover.out -o cover.html

CreatePerson 59.0%
total 60%



/internal/app/... /create_person.go (63.3%) not tracked no coverage low coverage * *

```
// CreatePerson .
func (i *Implementation) CreatePerson(ctx context.Context, req *desc.CreatePersonRequest) (*desc.Person, error) {
        err := validateCreatePersonRequest(req)
        if err != nil {
                return nil, status.Errorf(codes.InvalidArgument, "Validation failed: %s", err.Error())
        }

        phone, err := utils.NormalizePhone(req.Phone)
        if err != nil {
                return nil, status.Errorf(codes.InvalidArgument, "Validation phone failed: %s", err.Error())
        }
        var email string
        if req.Email != nil {
                email, err = utils.NormalizeEmail(*req.Email)
                if err != nil {
                        return nil, status.Errorf(codes.InvalidArgument, "Validation email failed: %s", err.Error())
                }
        }

        person := &model.Person{
                OzonID:              zero.IntFrom(req.OzonId),
                Phone:               zero.StringFrom(phone),
                Email:               zero.StringFrom(email),
                IdentificationLevel: desc.IdentificationLevel_ANON,
                Source:              desc.Person_ECOM_BANK,
                Bank:                desc.Bank_ECOM,
                DataLevel:           desc.IdentificationLevel_ANON,
        }

        err = i.storage.WithTransaction(ctx, func(tx sql.Transaction) error {
                err = createPersonInDB(ctx, tx, person, req.Reason, i)
                return nil
        })

        switch {
        case err == nil:
        case errors.Is(err, store.ErrEntityAlreadyExistByPhone):
```

ПРОБЕЛЫ

Как связать пробел и сценарий

ozon fintech

Та область, где разработчик и тестировщик на общем поле

Непокрытые участки кода

- Это либо конкретный сценарий использования сервиса
- Либо что-то более локальное, что не так влияет на работу сервиса

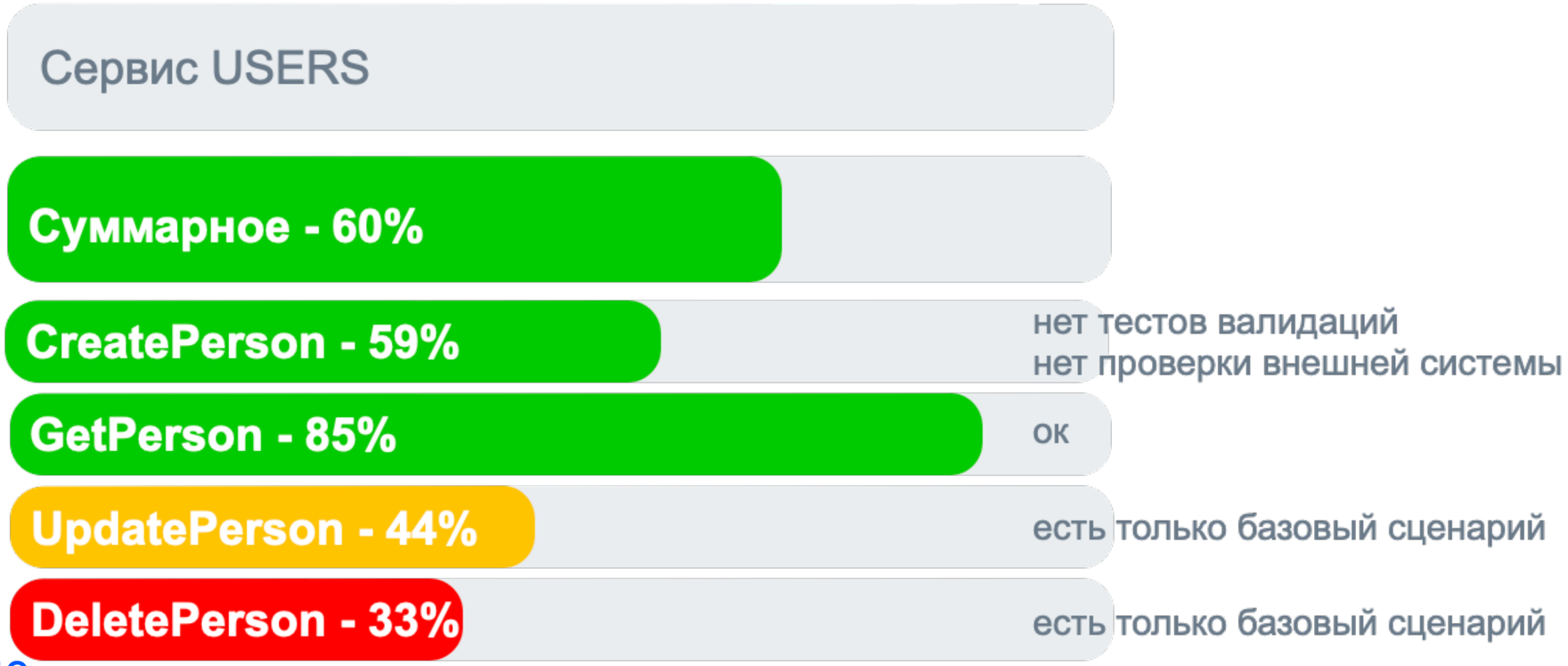
Как связать пробел и сценарий

ozon fintech

Та область, где разработчик и тестировщик на общем поле

Непокрытые участки кода

- Это либо конкретный сценарий использования сервиса
- Либо что-то более локальное, что не так влияет на работу сервиса



ТЕСТ АУДИТ

Тест-аудит

Реагировать, ставить задачи, покрывать проблемные места

Тест-аудит

ozon fintech

Реагировать, ставить задачи, покрывать проблемные места

А когда его проводить?

Тест-аудит

ozon fintech

Реагировать, ставить задачи, покрывать проблемные места

- Новая фича (задействует хэндлеры и участки кода)

А когда его проводить?

Тест-аудит

ozon fintech

Реагировать, ставить задачи, покрывать проблемные места

- Новая фича (задействует хэндлеры и участки кода)
- Фидбэк с инцидентов
(выявить типовую ошибку , найти похожие места)

А когда его проводить?

Тест-аудит

ozon fintech

Реагировать, ставить задачи, покрывать проблемные места

- Новая фича (задействует хэндлеры и участки кода)
- Фидбэк с инцидентов
  (выявить типовую ошибку , найти похожие места)
- Метрики

А когда его проводить?

Тест-аудит

ozon fintech

Реагировать, ставить задачи, покрывать проблемные места

- Новая фича (задействует хэндлеры и участки кода)
- Фидбэк с инцидентов
  (выявить типовую ошибку , найти похожие места)
- Метрики
  - активность на продуктовой среде

А когда его проводить?

rps из gRPC методов | rps из HTTP методов

HANDLER	AVG	MIN	MAX
CheckAnonCredentials	15.22	2.05	27.24
CloseClientProduct	0.77	—	10.97
CreateAddress	—	—	0.02
CreateAnonPerson	0.52	0.04	1.02
CreateClient	0.52	0.04	1.01

Тест-аудит

ozon fintech

Реагировать, ставить задачи, покрывать проблемные места

- Новая фича (задействует хэндлеры и участки кода)
- Фидбэк с инцидентов
  (выявить типовую ошибку , найти похожие места)
- Метрики
  - активность на продуктовой среде

А когда его проводить?

- Низкое покрытие / нет e2e тестов

rps из gRPC методов | rps из HTTP методов

HANDLER	AVG	MIN	MAX
CheckAnonCredentials	15.22	2.05	27.24
CloseClientProduct	0.77	—	10.97
CreateAddress	—	—	0.02
CreateAnonPerson	0.52	0.04	1.02
CreateClient	0.52	0.04	1.01

Handler ↓	Покрытие ↓	e2e ↓
CheckAnonCredentials	77.80 %	16
CheckCodeword	70.60 %	10
CloseClientProduct	76.20 %	55
CloseInactivePersons	0.00 %	—

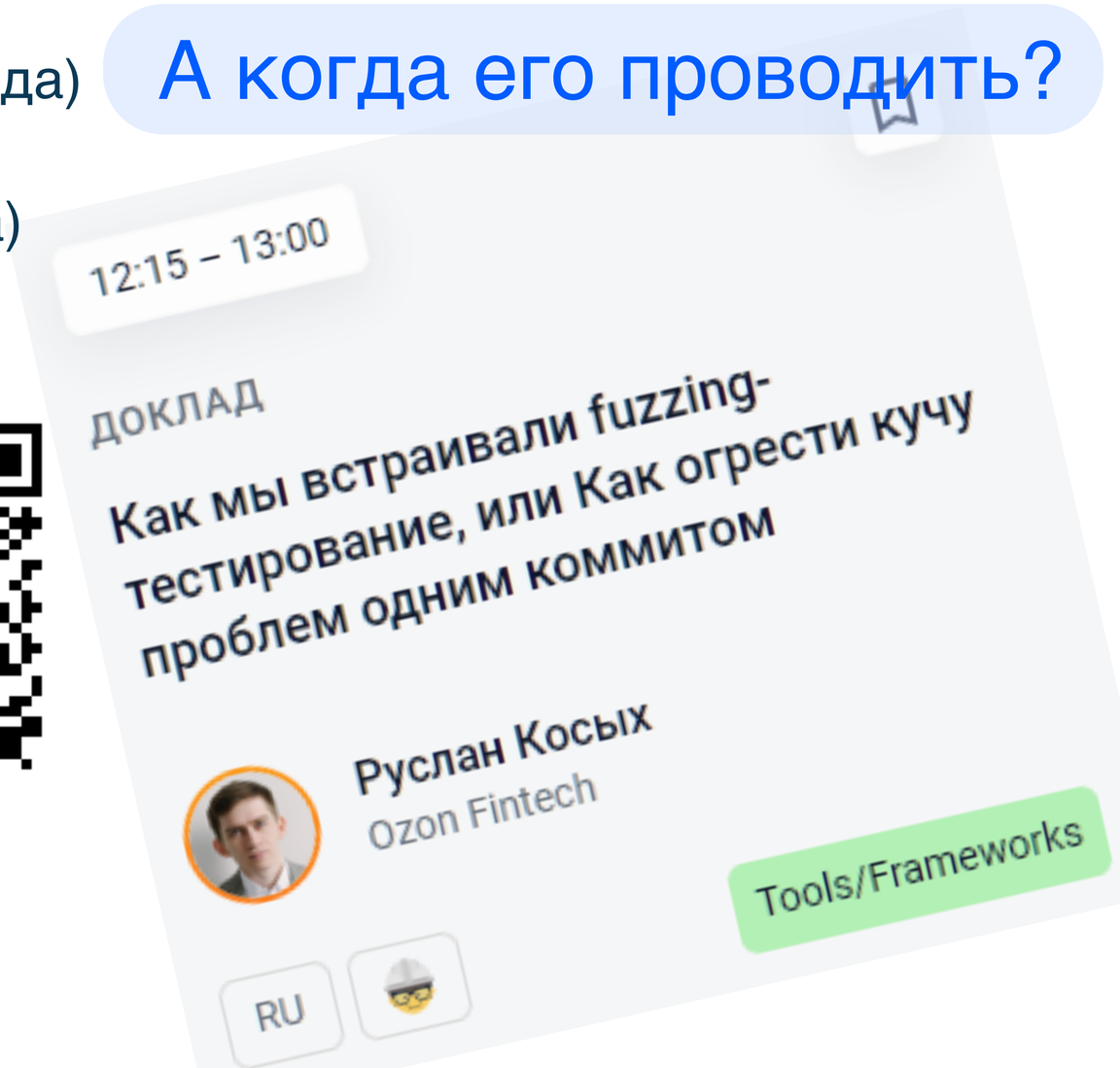
Тест-аудит

ozon fintech

Реагировать, ставить задачи, покрывать проблемные места

- Новая фича (задействует хэндлеры и участки кода)
- Фидбэк с инцидентов
  (выявить типовую ошибку , найти похожие места)
- Метрики
  - активность на продуктовой среде
  - Низкое покрытие / нет e2e тестов

А когда его проводить?



EXTRA:

инструменты типа Fuzzing тестирования

Аудировали-аудировали…

ozon fintech

И по итогу сформировали подход качественных тестов

Если писать тесты такого типа – будет минимум проблем. Какого?

Аудировали-аудировали…

ozon fintech

И по итогу сформировали подход качественных тестов

Если писать тесты такого типа – будет минимум проблем. Какого?

- **Позитивные/Негативные сценарии**
  (стандартные бизнесовые, когда все хорошо) и (случаи, когда «нет денег», «блокировки»)

```
res, err := app.CreatePerson("Vasya")
  require.NoError(t, err)
  require.NotNil(t, res.ID)
    require.Equal(t, res.Name, "Vasya")
```

Аудировали-аудировали…

ozon fintech

И по итогу сформировали подход качественных тестов

Если писать тесты такого типа – будет минимум проблем. Какого?

- **Позитивные/Негативные сценарии**
  (стандартные бизнесовые, когда все хорошо) и (случаи, когда «нет денег», «блокировки»)

- **Валидация**
  (четко указан формат входных данных и любое отклонение вызывает конкретные ошибки)

```
res, err := app.CreatePerson("Mr.username = 1' or '1' = '1')) LIMIT 1/")
 require.ErrorIs(t, err, "Invalid input")
```

Аудировали-аудировали…

ozon fintech

И по итогу сформировали подход качественных тестов

Если писать тесты такого типа – будет минимум проблем. Какого?

- **Позитивные/Негативные сценарии**
  (стандартные бизнесовые, когда все хорошо) и (случаи, когда «нет денег», «блокировки»)

- **Валидация**
  (четко указан формат входных данных и любое отклонение вызывает конкретные ошибки)

- **Технически-негативные** *(ошибки бд/таймауты/ошибки интеграции/пустые ответы…)*
  а также эмуляция нештатной работы внешних систем и проверки деградации)

```
extClientMocks.Expect().CheckABC().Return(nil, errors.New("failure"))
    ...
res, err := app.CreatePerson("Vasya")
require.NoError(t, err)
require.ErrorIs(t, err, "Ext service failure")
```

Аудировали-аудировали…

ozon fintech

И по итогу сформировали подход качественных тестов

Если писать тесты такого типа – будет минимум проблем. Какого?

- **Позитивные/Негативные сценарии**
  (стандартные бизнесовые, когда все хорошо) и (случаи, когда «нет денег», «блокировки»)

- **Валидация**
  (четко указан формат входных данных и любое отклонение вызывает конкретные ошибки)

- **Технически-негативные** *(ошибки бд/таймауты/ошибки интеграции/пустые ответы…)*
  а также эмуляция нештатной работы внешних систем и проверки деградации)

- **Комплексные**
  (Работа нескольких хэндлеров или сложные сценарии/логика, в том числе асинхронная)

Последние вопросы

ozon fintech

1. Как добиться большего покрытия меньшими усилиями
2. Какое покрытие на конкретном сервисе
3. Покрыты ли критичные места в коде/логике
4. Что покрывать в первую очередь
5. Улучшилось ли покрытие у сервиса за полгода
6. Улучшила ли команда покрытие за полгода

Следить за изменениями. Какие задачи это решает?

ozon fintech

Улучшилось ли покрытие у сервиса за полгода?

Улучшила ли команда покрытие за полгода?

- Еще одна непопулярная, но достаточно частая ситуация

Следить за изменениями. Какие задачи это решает?

ozon fintech

Улучшилось ли покрытие у сервиса за полгода?

Улучшила ли команда покрытие за полгода?

- Еще одна непопулярная, но достаточно частая ситуация

Тесты писать надо
Пример есть.
Помогающие есть.
Но тесты не пишутся

Следить за изменениями. Какие задачи это решает?

ozon fintech

Улучшилось ли покрытие у сервиса за полгода?

Улучшила ли команда покрытие за полгода?

- Еще одна непопулярная, но достаточно частая ситуация

Тесты писать надо
Пример есть.
Помогающие есть.
Но тесты не пишутся

И тут действуем
в несколько этапов:

1. Публичность общих цифр
2. Публичность просадок покрытия
3. Блокировка непокрытого тестами кода

Публичность общих цифр

ozon fintech

- Поставить рядом несколько команд
- Стимулирует быть лучше, включается соревновательный эффект

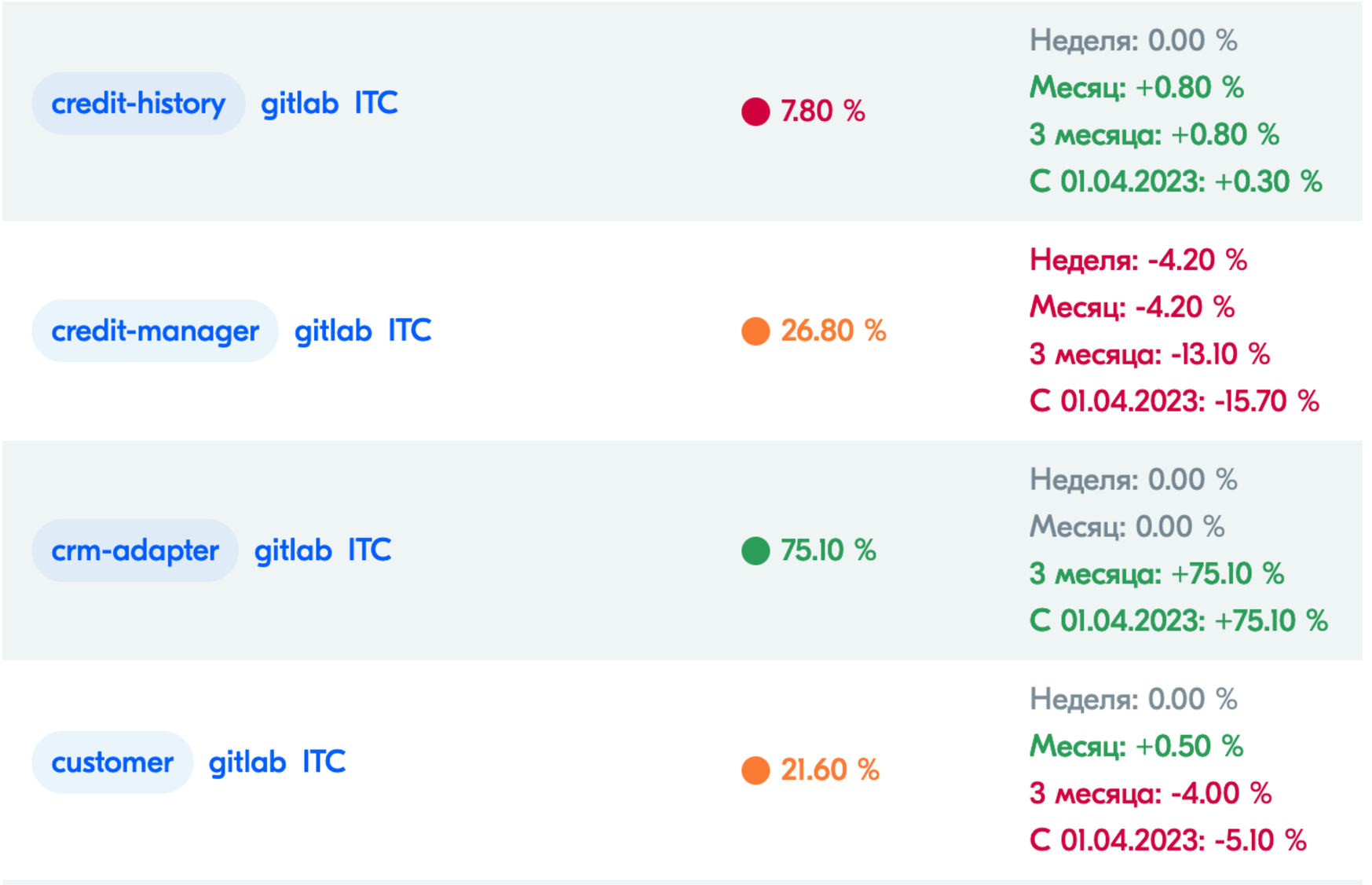
+ Косвенный! ориентир по производительности

Команда и ответственный	Покрытие ↓	Цель по покрытию	Детали ↓
purchases_team ITC Ответственный: Илья Муромец	89.86 % 	85.00 % до: 2023.10.01	13 services
loyalty_team ITC Ответственный: Алеша Попович	 68.74 %	60.00 % до: 2023.10.01	10 services
accounts_team ITC 2 teams Ответственный: Добрыня Никитич	36.15 % 	35.00 % до: 2023.10.01	16 services
applications_team ITC Ответственный: Василий Иванович	41.14 %	40.00 % до: 2023.10.01	11 services
backoffice_team ITC Ответственный: Петька Амбразурович	25.18 %	25.00 % до: 2023.10.01	5 services

Публичность просадок покрытия

ozon fintech

- **История изменений**
  и составление списка деградирующих сервисов



Публичность просадок покрытия

ozon fintech

- **Чат разработки**
  Регулярные отчеты о списке «деградирующих» сервисов и ответственных

- **Критичный сервис? Появление большого босса в беседе**
  - призыв к действию и в итоге результат

Команда accounts_team (@Sergey Sergeev), просадка покрытия с 2023-04-01

Project	Coverage	OwnerSubTeam		
credit-manager	-3.30	@Aleksandr Ivanov	Gitlab	QACID
customer	-5.10	@Daniil Kirin	Gitlab	QACID
monitoring	-2.70	@Artem Kornev	Gitlab	QACID

25 replies Following

Блокировка нового непокрытого кода (когда ничего больше не помогает)

ozon fintech

Pipeline #17589711 passed with warnings for 9d92086f on release/GG-54353-1 1 week ago

Test coverage 63.10% (-0.20%) from 1 job

```
39        -          db, ok := fcrepo.Balancer.Next(
      38  +          db, ok := fcrepo.Balancer.Next(
40    39             // Определяем поля фильтров по
41    40
42    41             if !ok {
      ↓ ↑  @@ -161,17 +160,17 @@ func (fcrepo DBFil
           controlF fi
161   160            case desc.ListControlsRequest_M
162   161                    {
163   162                            if req.GetSortB
164       -                                   stmt =
      163 +                                   stmt =
           table.Controls.DetectedAt.ASC())
165   164                            } else {
166       -                                   stmt =
      165 +                                   stmt =
           table.Controls.DetectedAt.ASC())
167   166                            }
```

Мгновенно подсветит проблему

Снизит человеческий фактор

Итоги

ozon fintech

- **Можно применять решения**

- **Для личного удобства**

Итоги

ozon fintech

- **Можно применять решения:**
  - сервис генерации данных
  - система моков внешних сервисов
  - балансировщик тестов
  - система трассировок
  - тест-дизайн юнит-тестов
  - система хранения покрытия
- **Для личного удобства - и позже применяем в массах**

Итоги

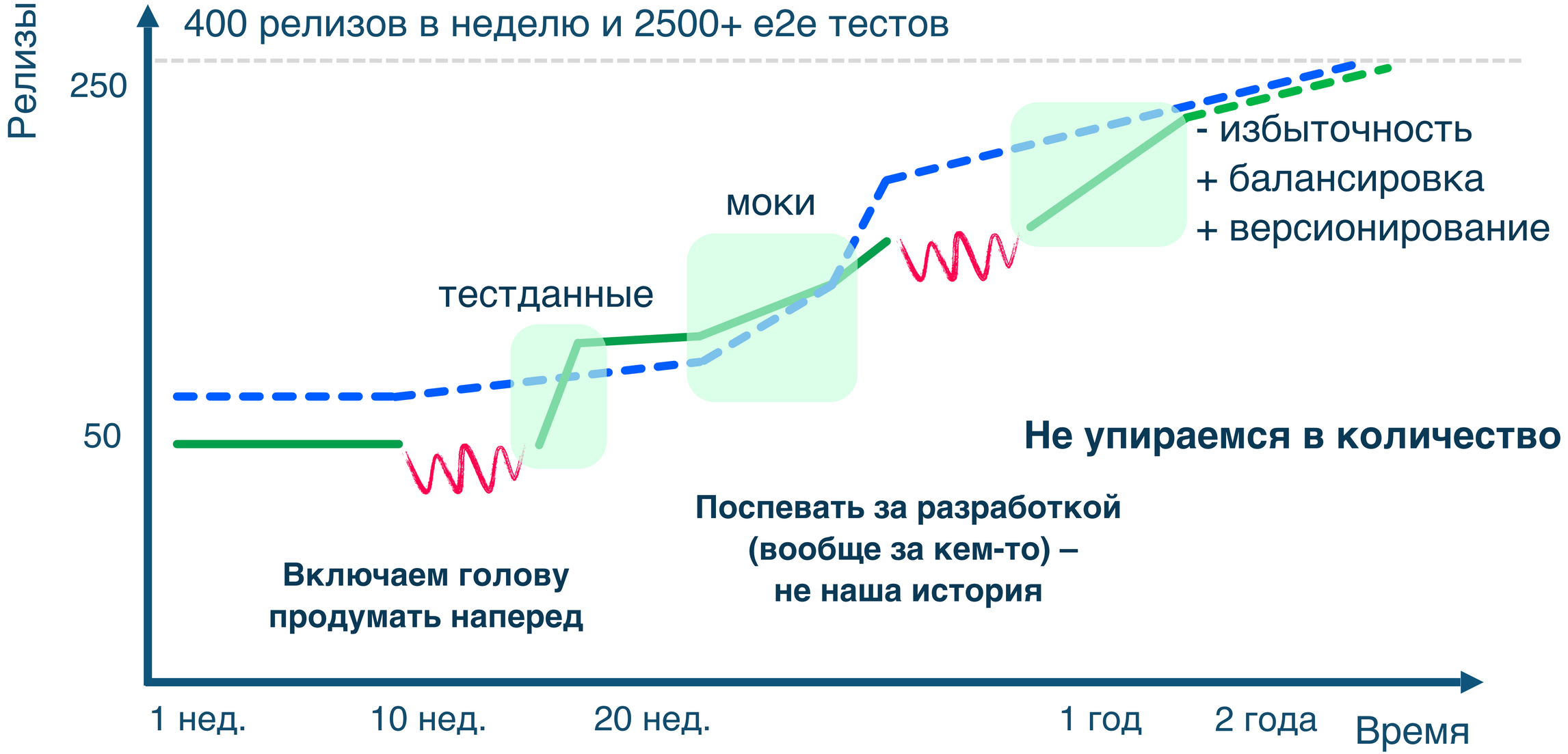
ozon fintech

- **Можно применять решения:**
  - сервис генерации данных
  - система моков внешних сервисов
  - балансировщик тестов
  - система трассировок
  - тест-дизайн юнит-тестов
  - система хранения покрытия
- **Для личного удобства - и позже применяем в массах**
- **Улучшаем разработку, тестопригодность, контролируемость и предсказуемость**

- **Не упираемся в количество**

Итоги

ozon fintech



Итоги

ozon fintech

Делаем мир счастливей
(только не для тех, кто не пишет тесты)

ozon fintech

Конец

Подумать

ozon fintech

- **Побудить разработчиков следить за e2e-прогонами**
- **Можно заблокировать выкатку (ci-cd) в случае рисков от тест.аналитики**
- **% покрытия строчек кода через e2e**
- **Развитие тестопригодности**

- **Поспевать за разработкой (вообще за кем-то) – не наша история**
- **Всегда надо включать голову, продумать наперед**

Попробовать

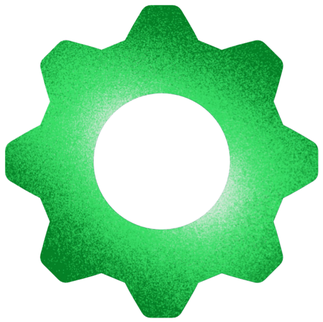
ozon fintech

Сейчас типизированно отслеживаем % покрытия
http/grpc хэндлеров go (scratch)

Пробуем вариант оценки:

- обработчиков сообщений брокера
- триггеры асинхронных задач/задач по расписанию
- работу стейтмашины
- graphQL запросы / мутаторы

+ оценка в web репозиториях:
api хэндлеры , виджеты и объекты..



ozon fintech

Конец

(точно)

ozon fintech

Спасибо за внимание

Егор Стручин, тимлид группы тестирования микросервисов

estruchin@ozon.ru