

ozon{tech

Как мы оптимизировали запуски e2e и дооптимизировались

Алексей Романов, ведущий инженер по АТ в Ozon.Fintech

Обо мне



Алексей Романов

 @lexcorp

- Ведущий инженер по АТ в **Ozon.Fintech**
- Пишу **сложные моки**
- Оптимизирую запуск **интеграционных тестов**

План выступления

ozon.tech

1. Контекст

2. Проблематика

3. Открытие

4. Реализация

5. Осознание

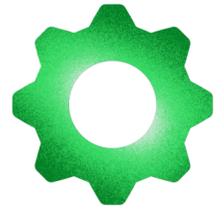
6. Результаты

7. Выводы



1

Контекст



Ozon Fintech v1

OZONtech

300+

релизов в неделю

200+

сервисов

750+

прогонов в неделю

1500+

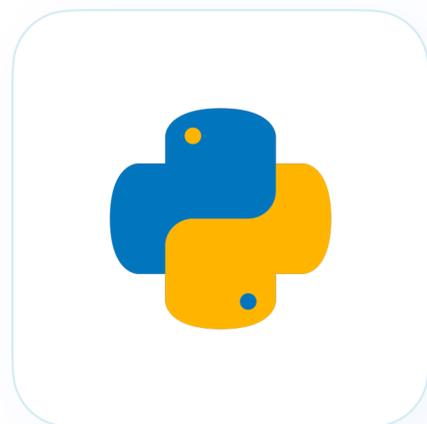
автотестов

~15-25 мин

прогон по сервисам

Погружаемся в контекст

Стек автотестов



Разметка тестов



```
1 @pytest.mark.feature1
2 def test_feature1():
3     with allure.step("Step 1"):
4         smth do
```

Пример



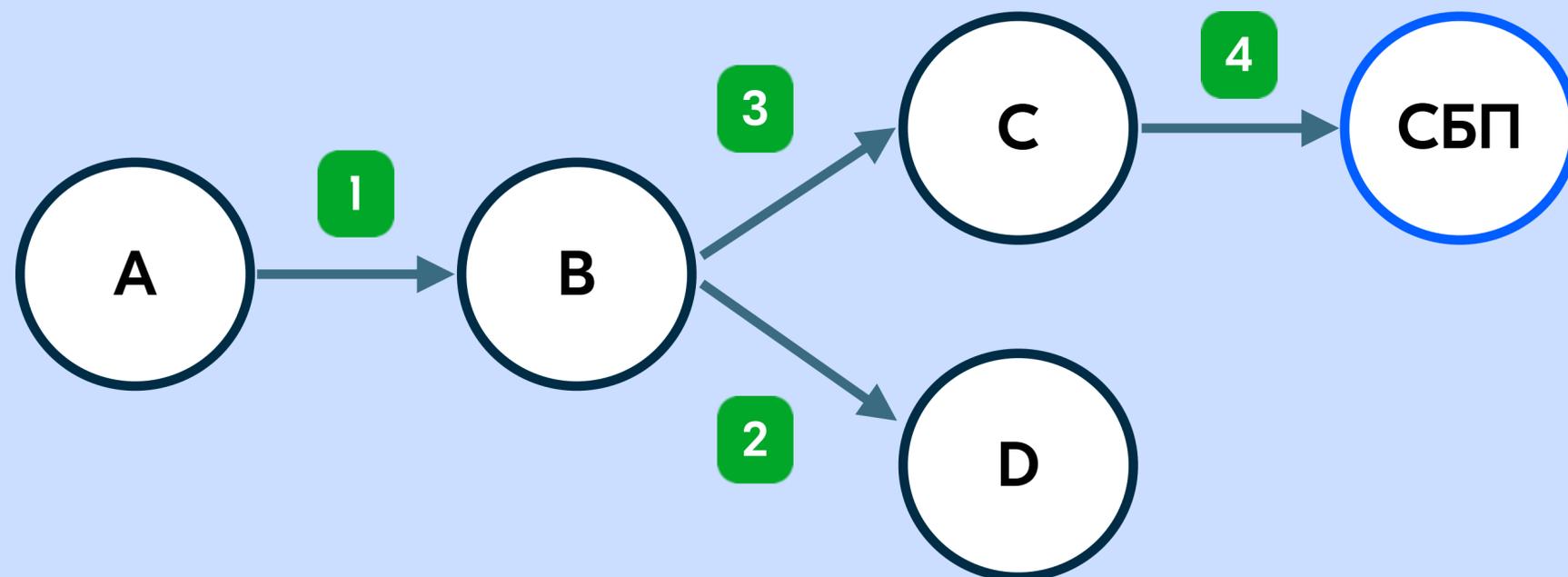
Для каждого сервиса выбираем список марок,
по которым будет запуск

2

Несогласие с порядком вещей

Проблемы ручной разметки

Перевод средств по СБП

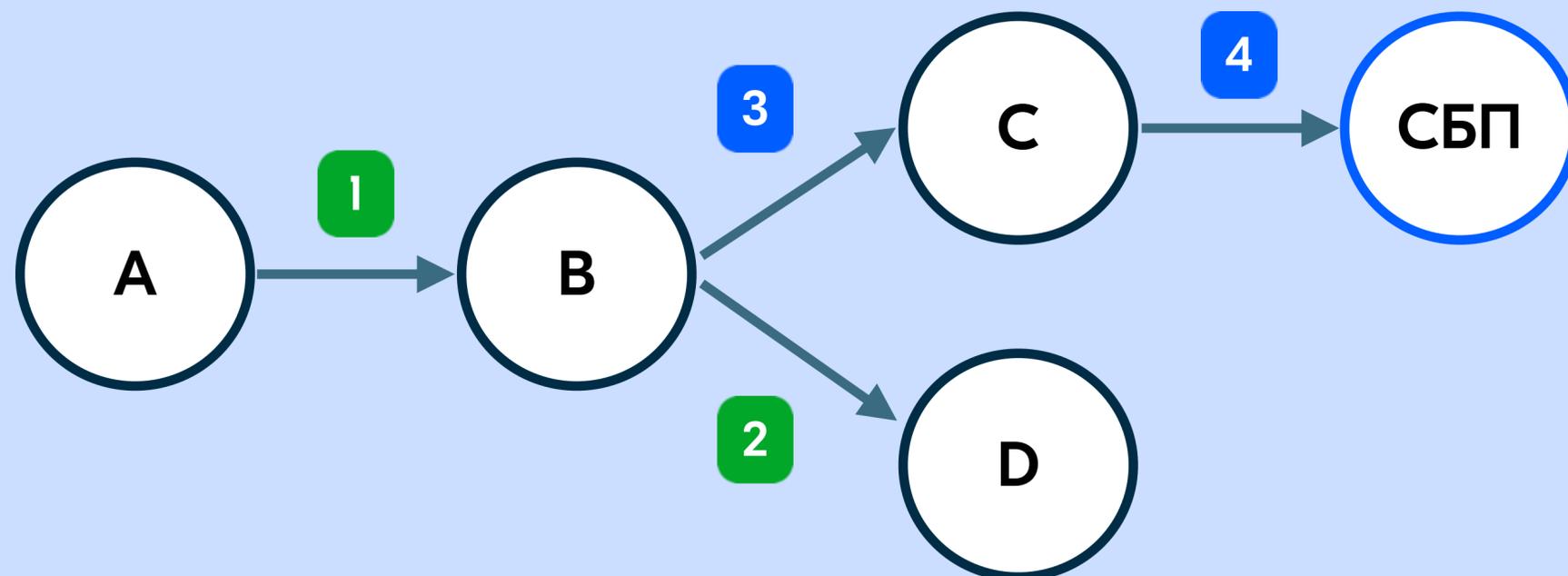


Тест 1

Успешный перевод средств по СБП

Проблемы ручной разметки

Перевод средств по СБП



При запуске тестов для сервиса C запустится тест, который не затрагивает этот сервис

Тест 1

sbp

Успешный перевод средств по СБП

Тест 2

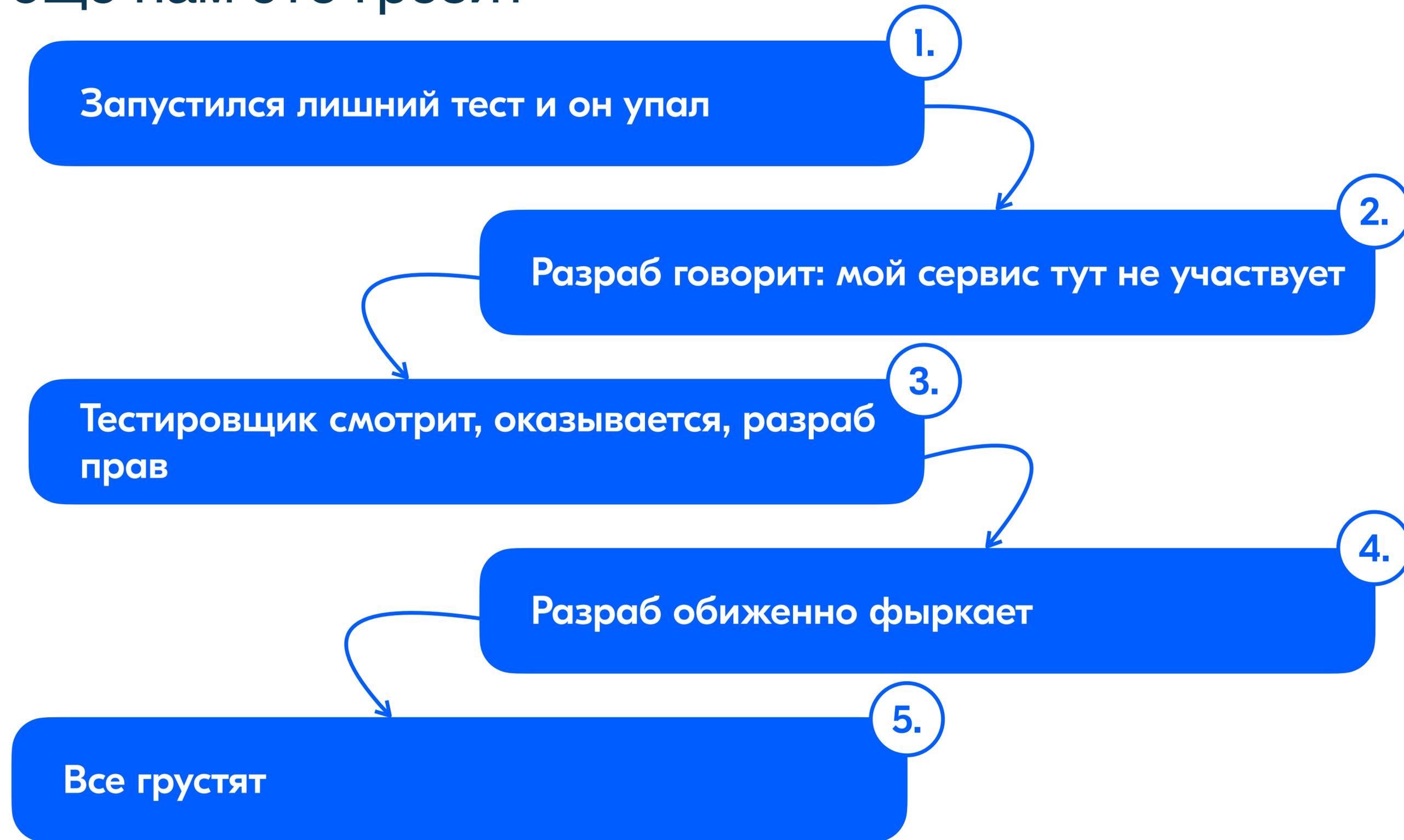
sbp

Отказ. Недостаточно средств

Чем нам это грозит



Чем еще нам это грозит



Но может быть и по-другому

1. Тест затрагивает сервис, но не попал в разметку

2. Во время релиза этот тест был сломан, но в пайплайне не запустился

3. У разраба зеленые тесты. Поехали в прод!

4. Пока все опомнились — баг уже в проде

5. Все грустят



Ну и что?

- Только автотесты
- 300+ релизов в неделю

Группируем угрозы

1. Запускаем лишние тесты

2. Можем не запустить нужные тесты

3. Нет доверия к разметке тестов

4. Большая нагрузка на СТГ
(~130 rps на gateway)

5. Долгий прогон тестов (20-25 минут)

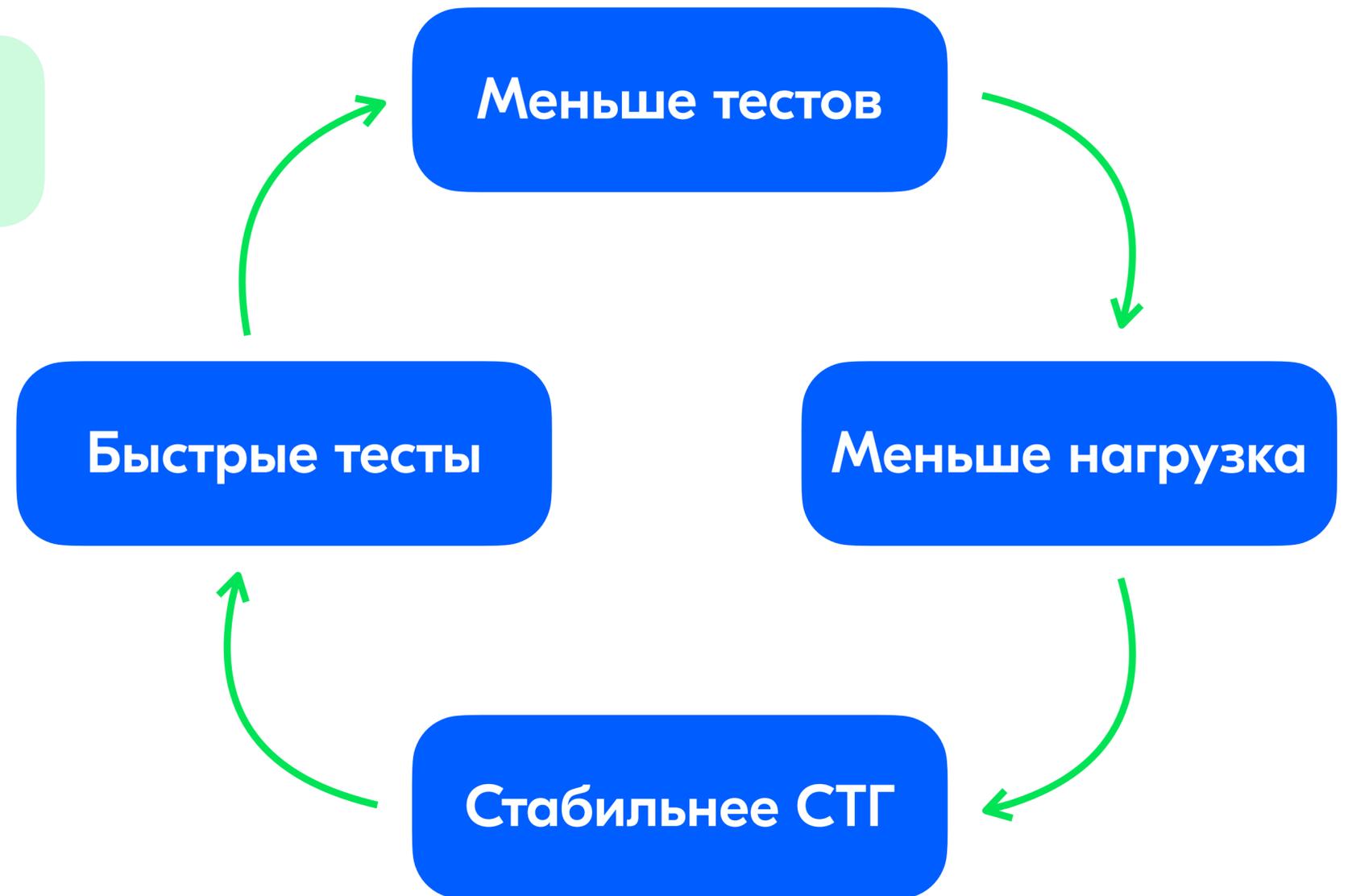
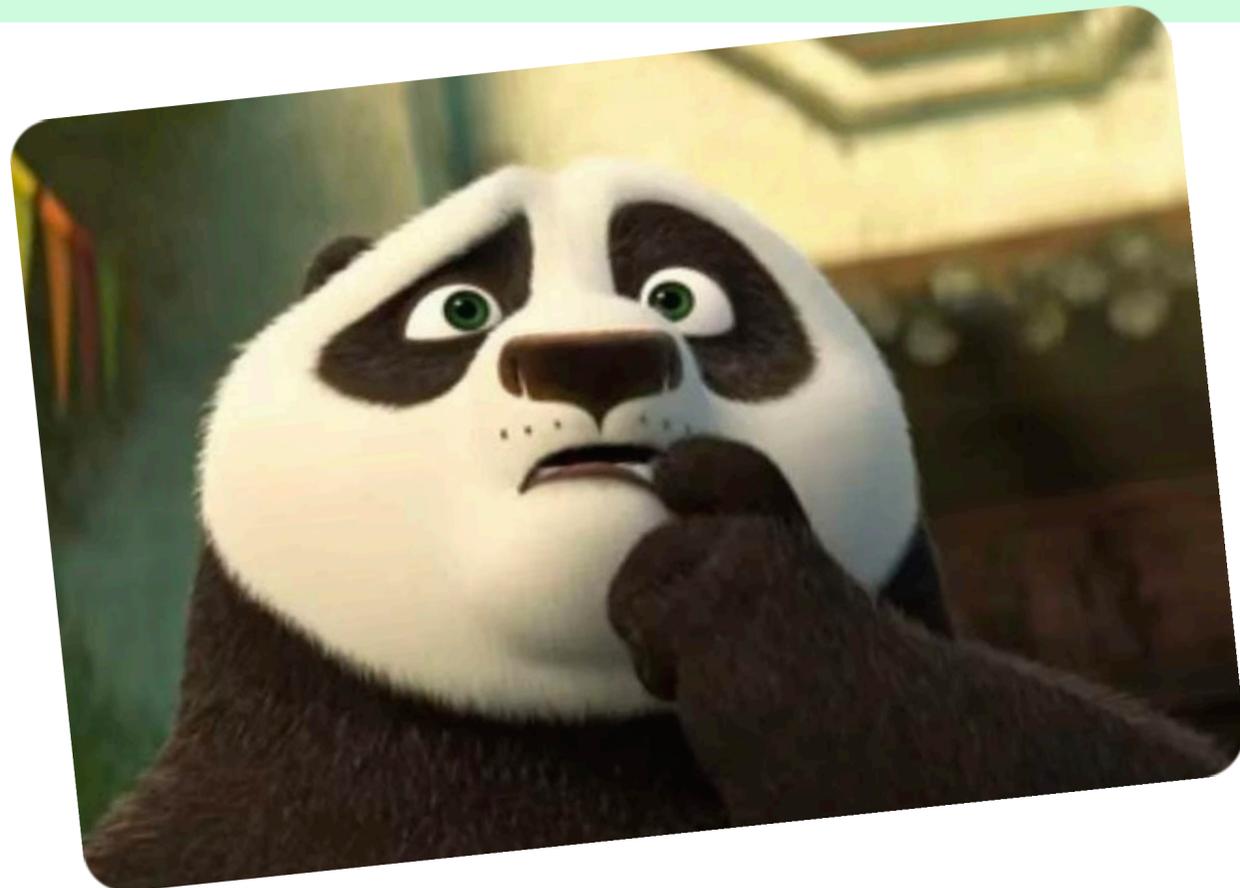
6. Руками размечать 1500 тестов

7. Следить за актуальностью разметки



А чего мы тогда хотим

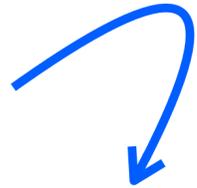
1. Автоматическая разметка тестов для сервисов
2. Запускать для сервиса только те тесты, которые он затрагивает



ВСЁ!

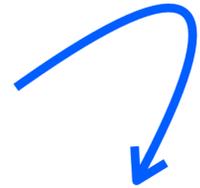
или нет?)

Чего хотим



1. Автоматическая разметка тестов для сервисов
2. Запускать для сервиса только те тесты, которые он затрагивает

Что поможет

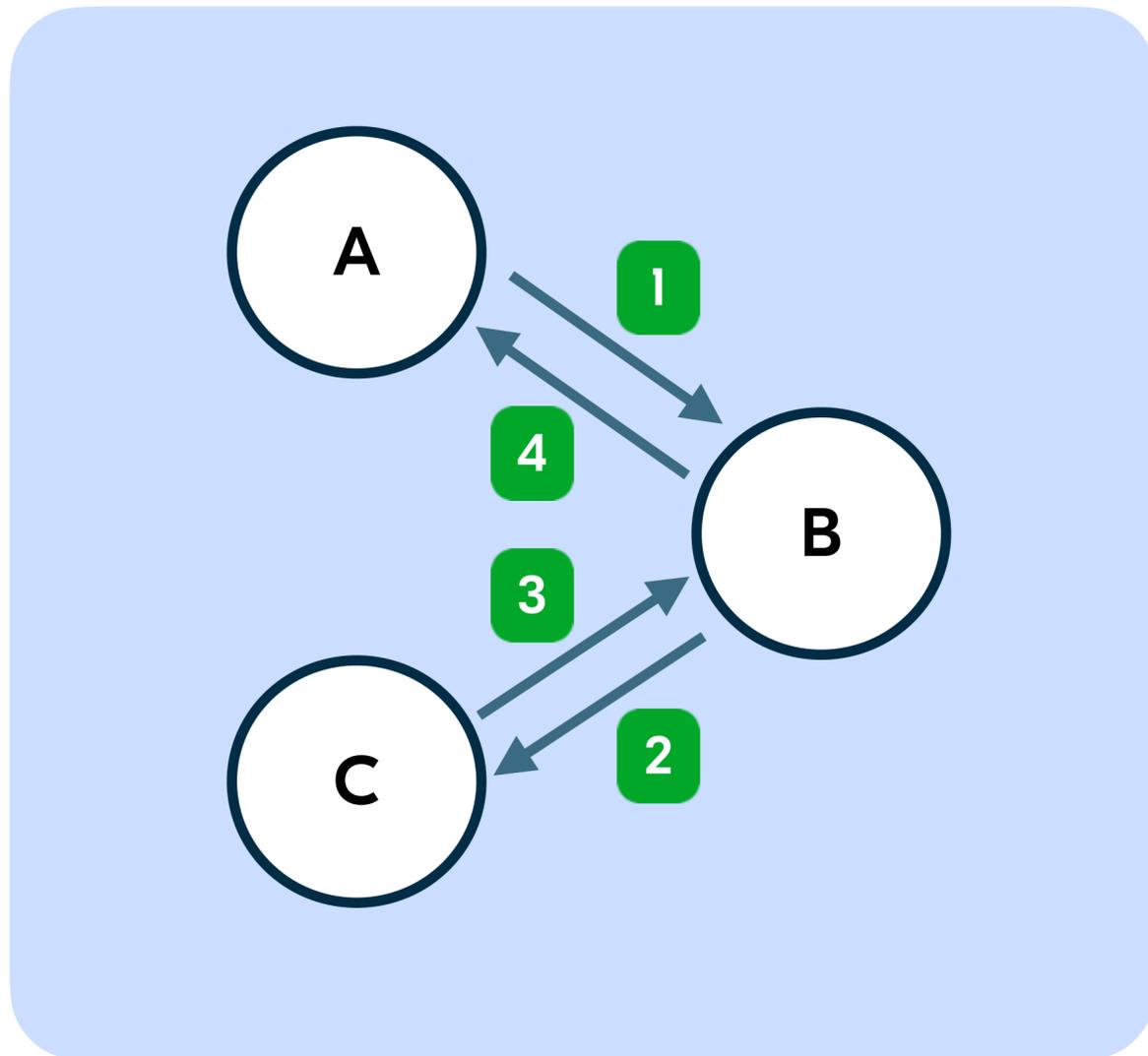


Система, которая знает о запросе буквально **ВСЕ**

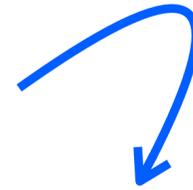


3

Трейсинг



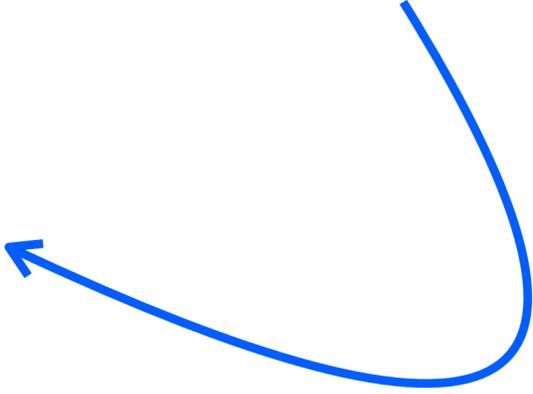
Что полезно

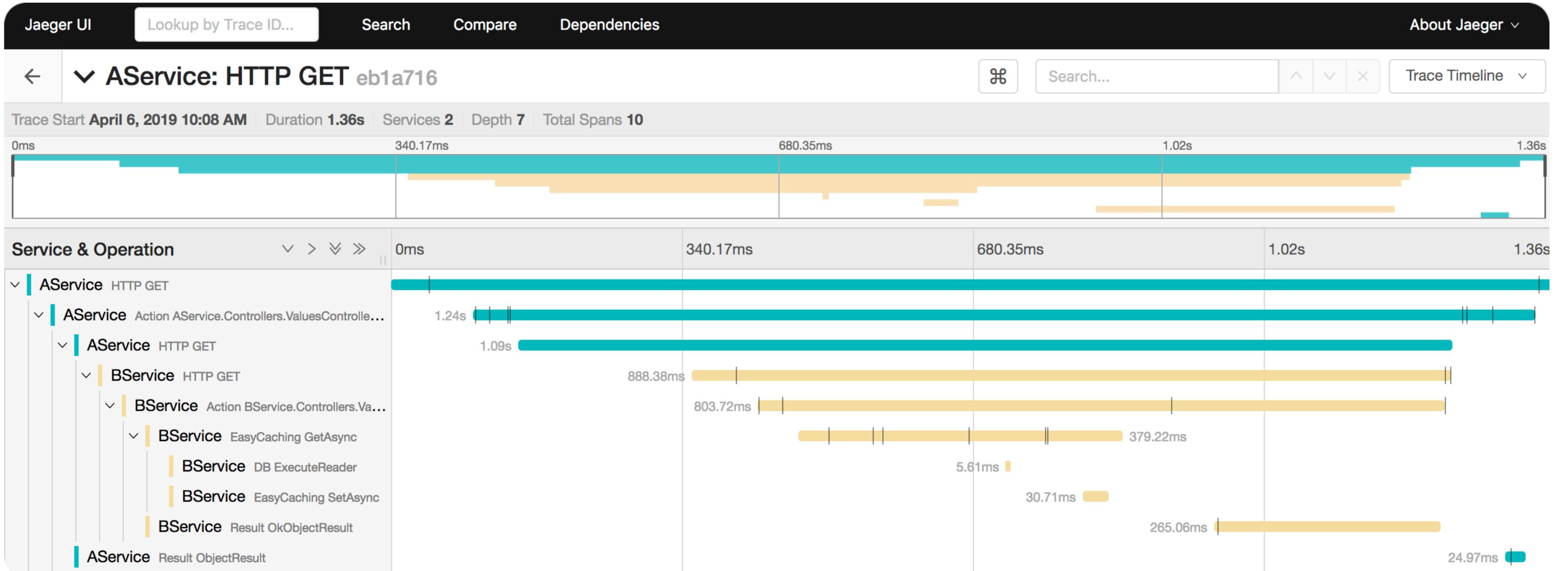


- Наглядно посмотреть, какой путь был у запроса
- Для каждого шага запроса посмотреть метаданные
- Достать все это через API
- Уникальный идентификатор запроса (TraceID)
- Opensource

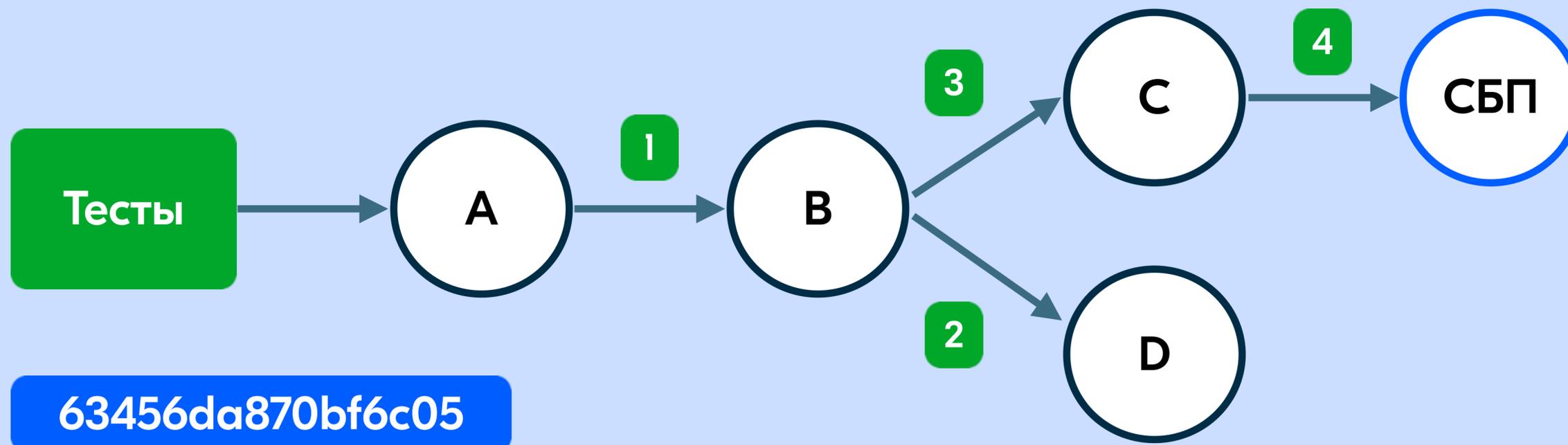


Послушать тут

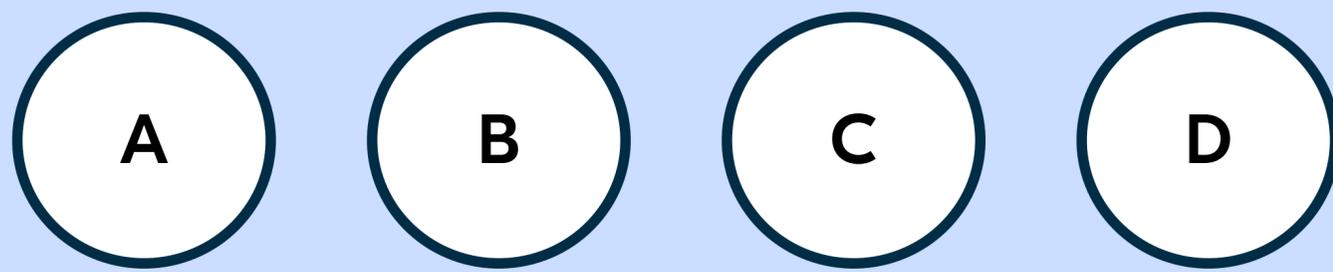




Перевод средств по СБП



63456da870bf6c05



1. Система, которая знает о запросе все

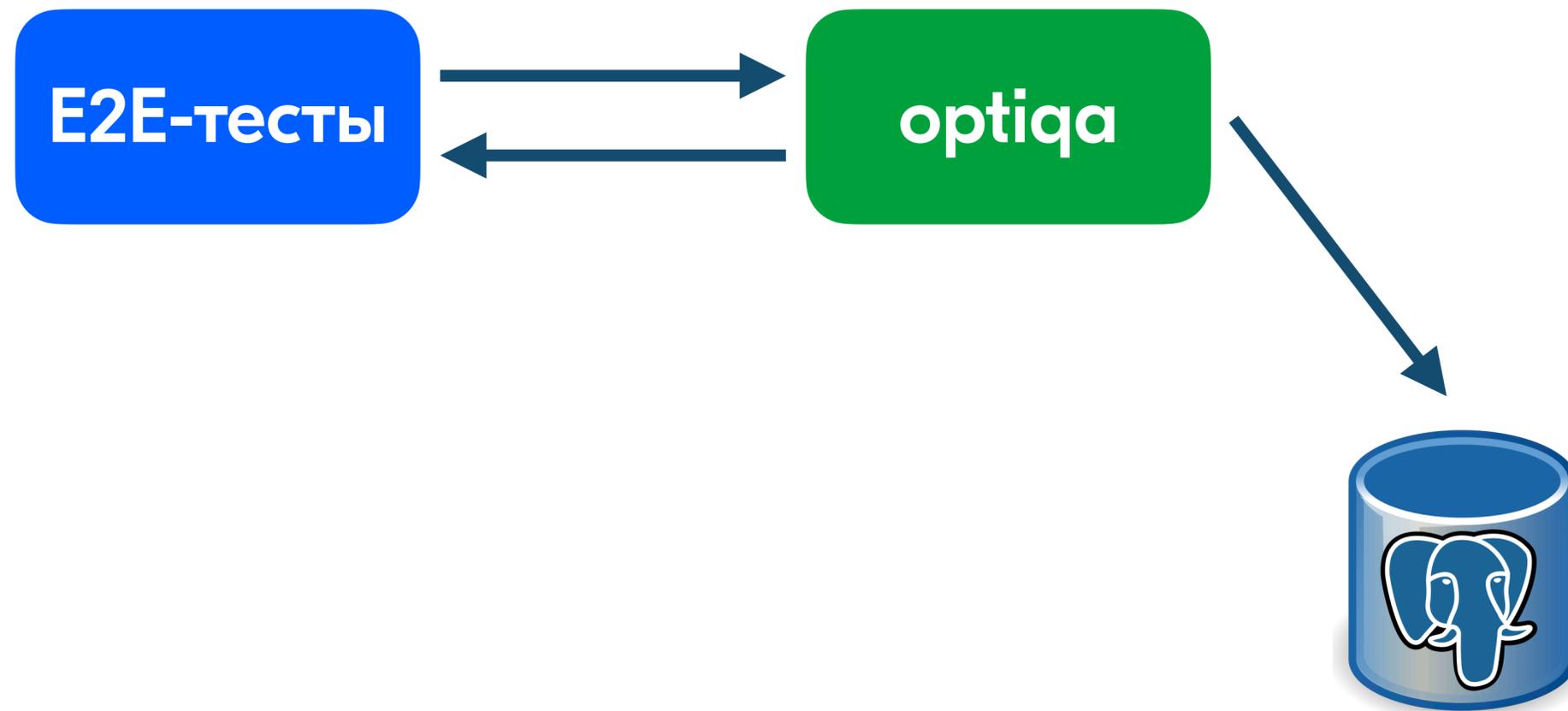
2. Сервис, который будет хранить и предоставлять данные для тестов

Требования к сервису

1. По traceID забирать информацию из Jaeger
2. Составлять зависимость тест — сервисы
3. Эффективно обновлять зависимости от новых данных
4. Быстро отдавать необходимую информацию
5. Сервис должен называться optiqa



Как это будет выглядеть

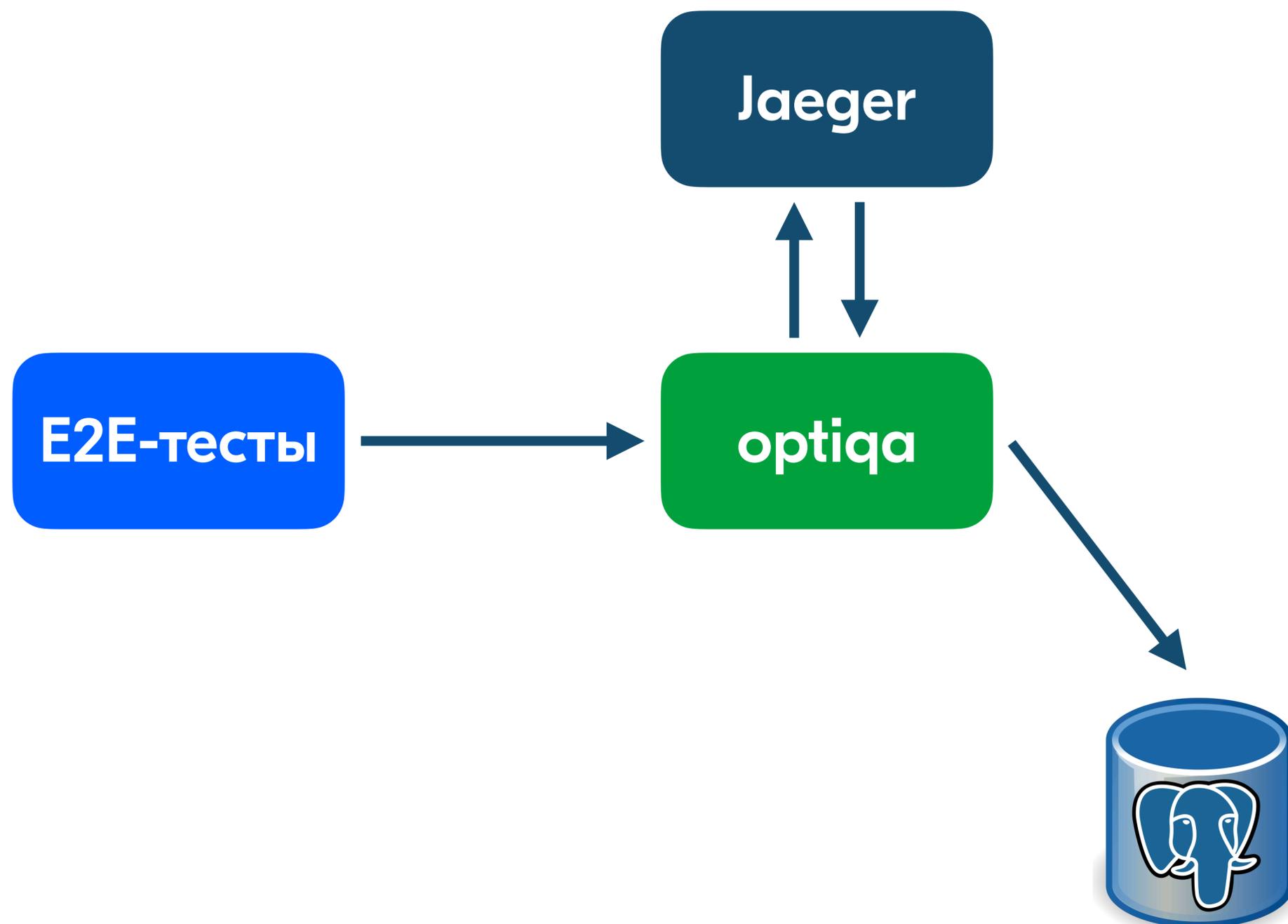


1. E2E идут в оптику за тестами для сервиса

2. Оптика достает из базы тесты, которые собрала по сервису

3. E2E запускают полученные тесты

После прогона



1. Тесты отдают все трейсы в связке с тестами

2. Оптика получает из jaeger информацию по сервисам и сохраняет ее

3. Оптика сохраняет всю полученную информацию в базе

1.

Забрать тесты
из оптики

2.

Запустить тесты
и получить
результаты

3.

Отправить
полученные
данные в оптику

4

Реализация

Go

PostgreSQL

СУБД

Scratch

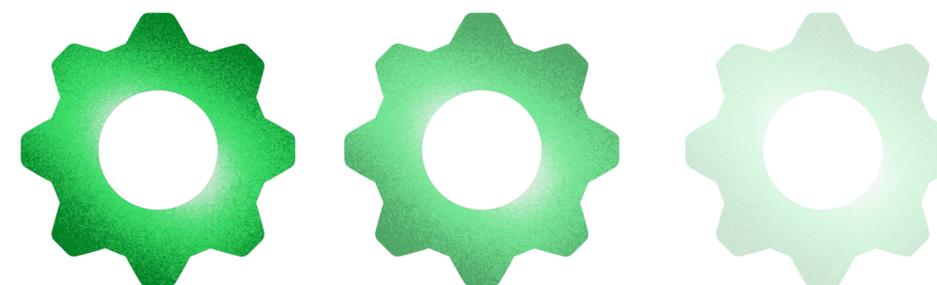
Платформа

Prometheus

Метрики

GitLab

CI/CD



```
@pytest.hookimpl(tryfirst=True)
def pytest_collection_modifyitems(items):
    tests_to_run = optiqa_api.get_tests(service=config.SERVICE)
    for item in items:
        is_in_test_to_run = item.nodeid in tests_to_run
        if is_in_test_to_run:
            item.add_marker(to_be_launched)
```

Откуда брать данные для оптики

1. Класть трейсы в соответствующий шаг allure отчета

✓ Request: 'GET' ' http://
1 sub-step

ШАГ

✓ trace link http://
1 parameter

Trace

trace_id

d493166e65cd1ac0

Откуда брать данные для оптики

1. Класть трейсы в соответствующий шаг allure отчета

2. После прогона тестов парсить отчет

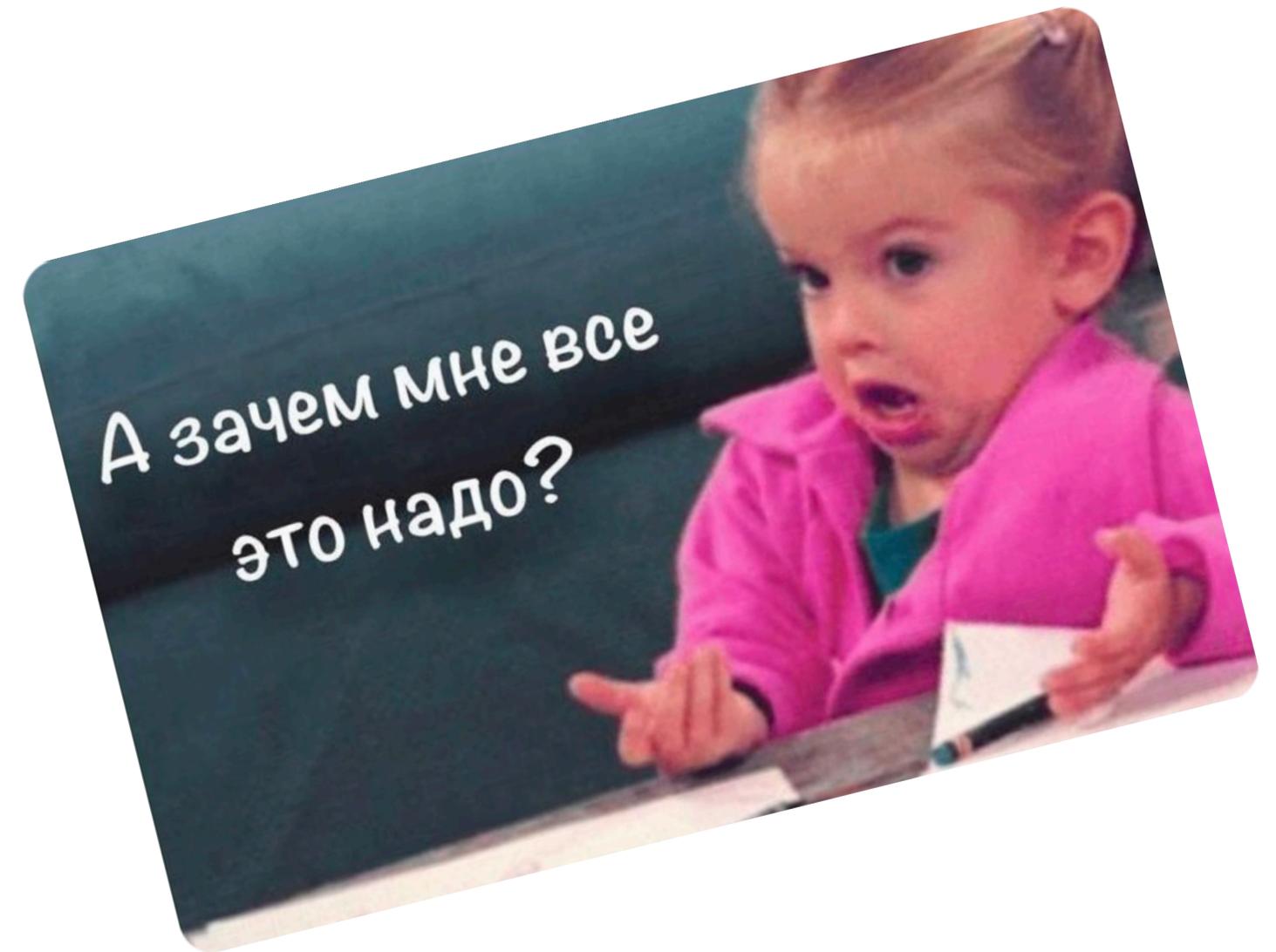


А давайте сохранять еще и статистику

1. Время выполнения теста

2. Статус теста

3. Рераны теста

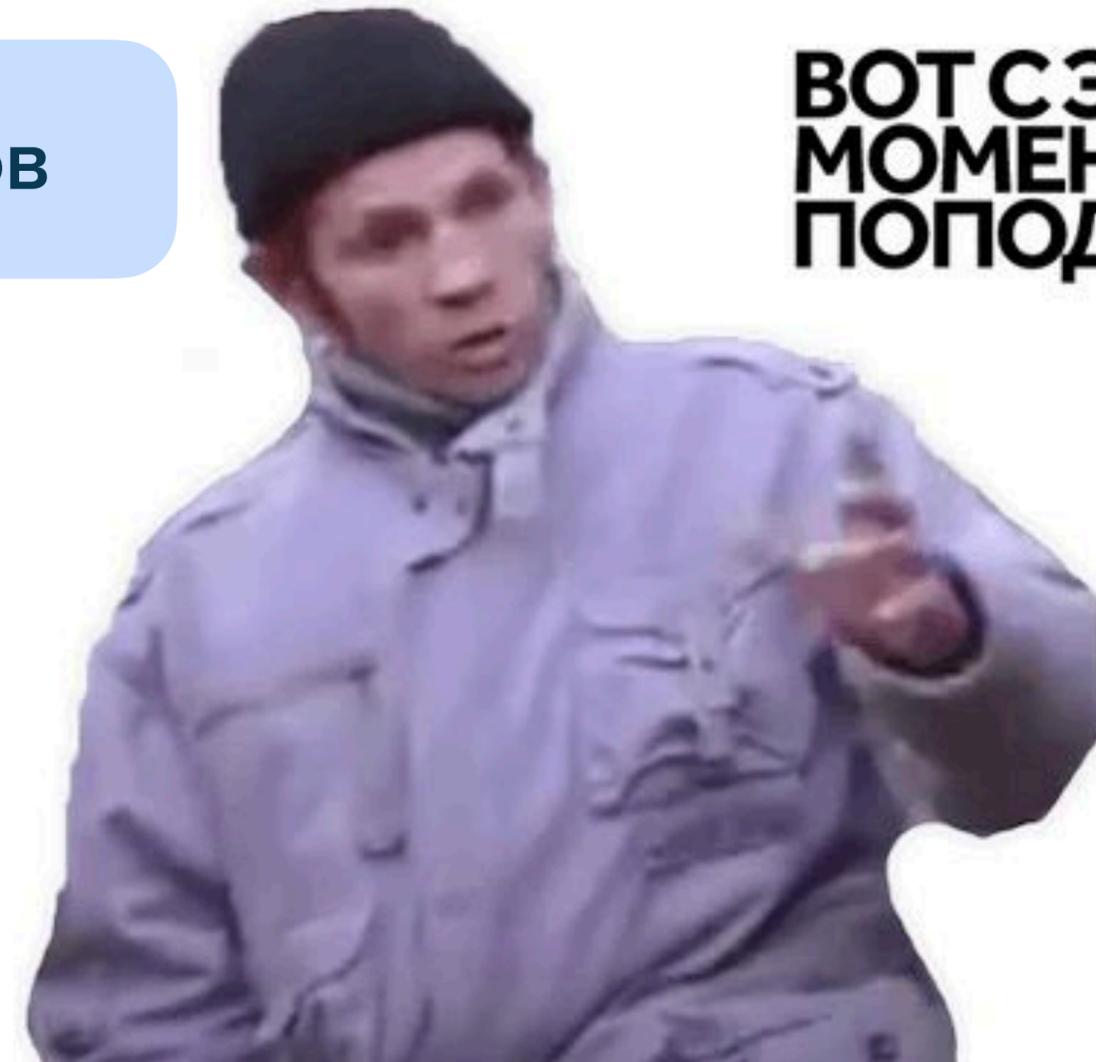


А зачем

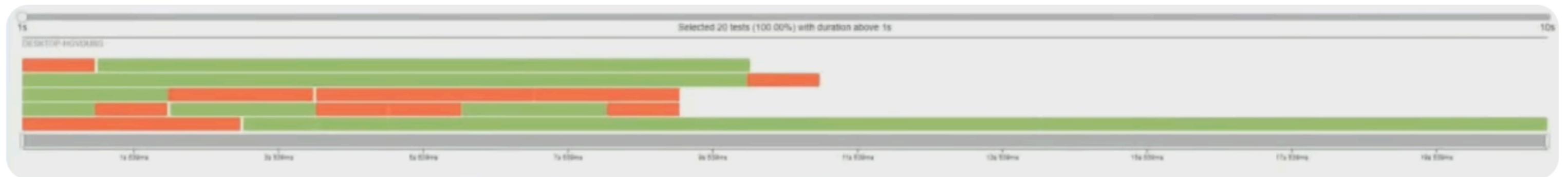
1. Собираем статистику по тестам

2. Можем ускорить выполнение тестов

**ВОТ С ЭТОГО
МОМЕНТА
ПОПОДРОБНЕЕ**

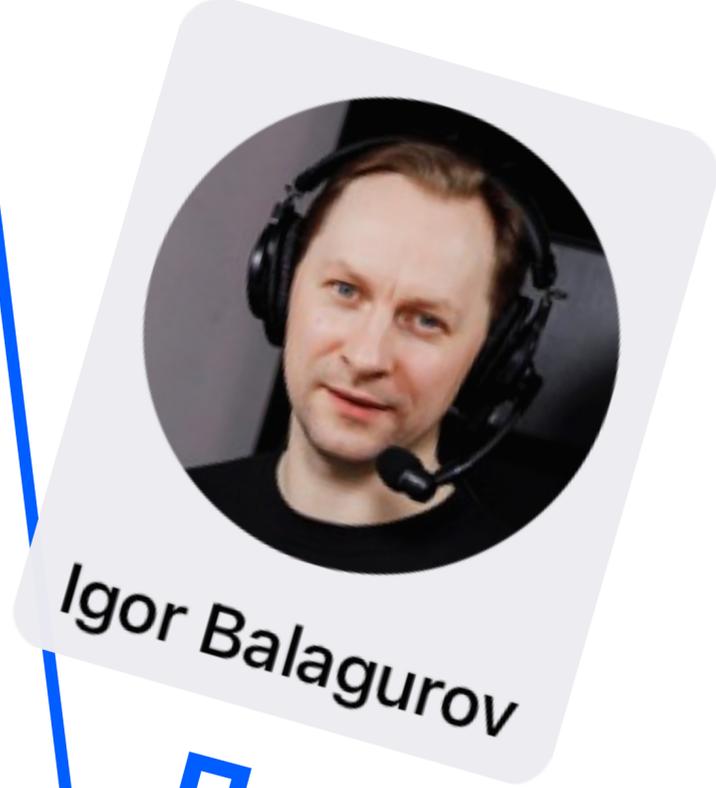


1. Без статистики сомнительная балансировка



2. Сортируем тесты по времени выполнения





**Послушать
тут**

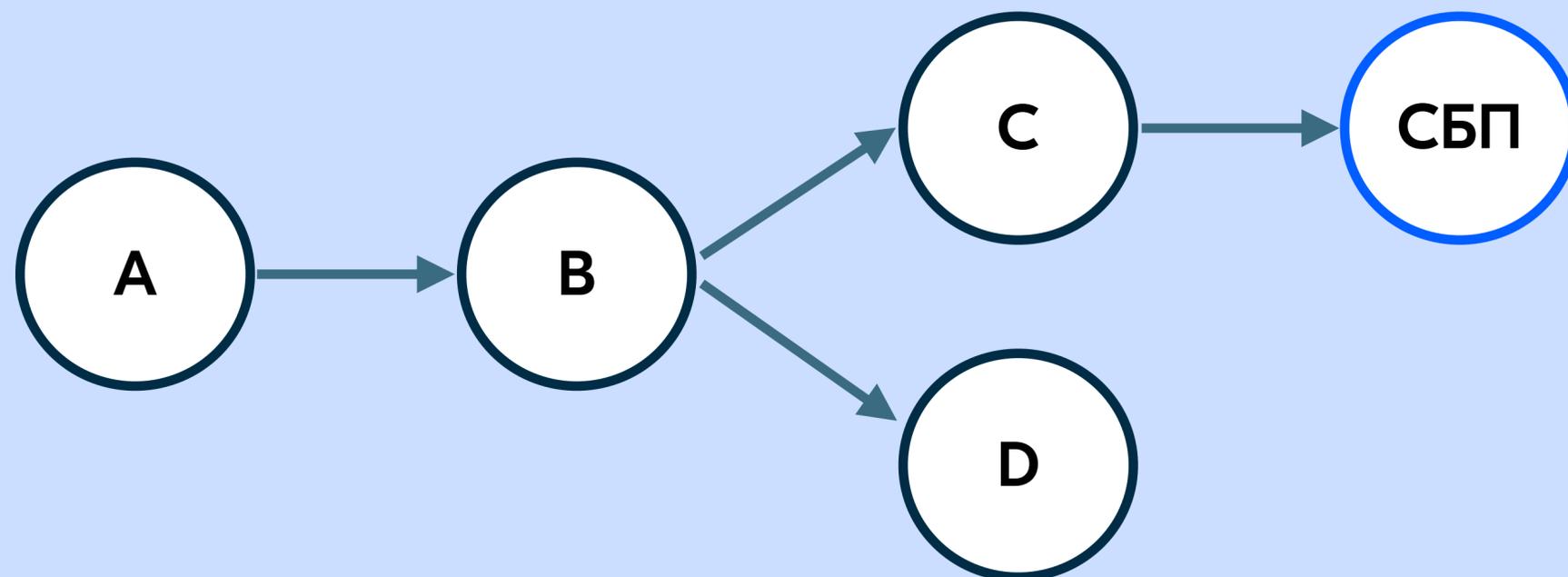


5

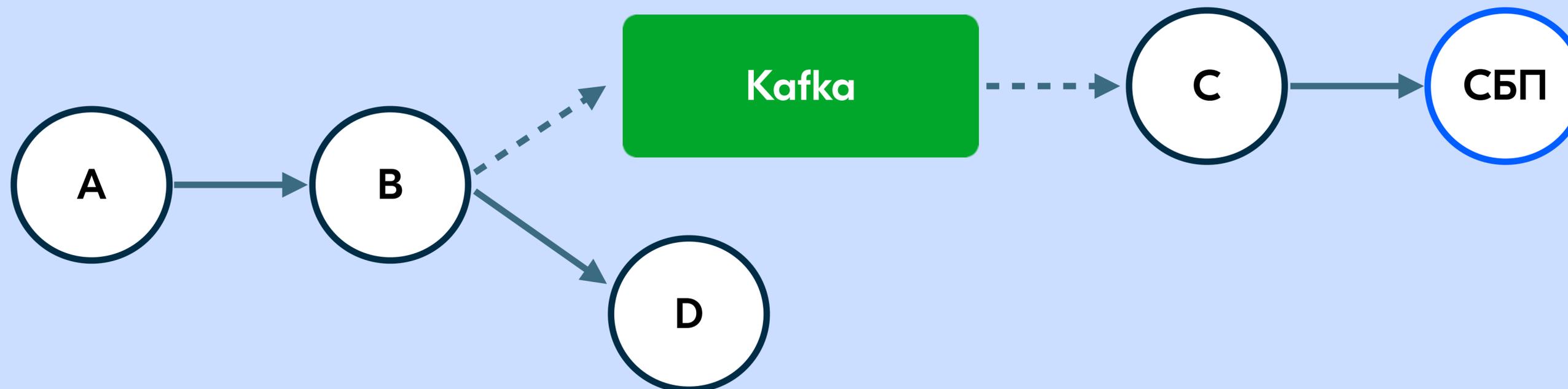
Осознание

Было так

Перевод средств по СБП



Перевод средств по СБП



Будут ли пробрасываться трейсы через кафку?

НЕТ!



А что это значит

1. Jaeger не знает об асинхронных операциях



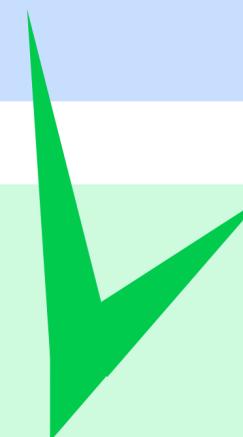
Мы теряем трейсы, Карл!

Где потеряли, там и найдем

Jaeger **не знает** об асинхронных операциях

1. Рассказать ему об асинхронных операциях

2. Просто марать такие тесты





```
@pytest.mark.force_mark("service1")  
def test_success_smth():  
    with allure.step("Step 1"):  
        smth do
```

1. Да, опять вернулись к маркам

2. Но теперь размечать нужно только конкретные кейсы

Фильтрация тестов (было)

```
@pytest.hookimpl(tryfirst=True)
def pytest_collection_modifyitems(items):
    tests_to_run = optiqa_api.get_tests(service=config.SERVICE)
    for item in items:
        is_in_test_to_run = item.nodeid in tests_to_run
        if is_in_test_to_run:
            item.add_marker(to_be_launched)
```

```
@pytest.hookimpl(tryfirst=True)
def pytest_collection_modifyitems(items):
    tests_to_run = optiqa_api.get_tests(service=config.SERVICE)
    for item in items:
        is_in_test_to_run = item.nodeid in tests_to_run
        own_forced_mark = self._has_forced_mark(item, config.SERVICE)
        if is_in_test_to_run or own_forced_mark:
            item.add_marker(to_be_launched)
```

```
def _has_forced_mark(item: pytest.Item, clear_test_service_name: str) -> bool:  
    for mrk in item.own_markers:  
        if mrk.name == "forced_mark":  
            for mark_name in mrk.args:  
                if clear_test_service_name in mark_name:  
                    return True  
    return False
```



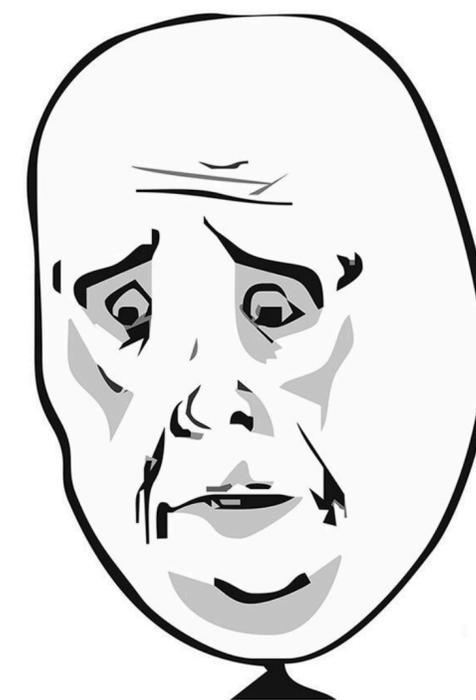
Но каково же было наше удивление (тут объяснить)

1. Срок хранения трейсов в оперативе **20 минут**

2. Персистентно хранится только **1%** всех трейсов

3. Сервис может не успеть зарепортировать информацию

*Мы теряем
информацию*



**У вас может быть
по-другому**

Основные проблемы

1. Есть вероятность не успеть получить трейс

2. Есть вероятность получить неполную информацию

Так может стоит просто
поднастроить jaeger ?))0)))

ДЕЙСТВИТЕЛЬНО...



Накидываем железо на вентилятор

1. Увеличим срок хранения трейсов
2. Все трейсы начнем хранить персистентно

Оба решения приведут к бесконечному разрастанию ресурсов на Jaeger

Но не решают корень проблемы

Настройки Jaeger

Срок хранения —
20 минут

1% трейсов хранится персистентно

Можем не успеть получить трейс

20 минут
на все
про все

15 мин
прогон

5 мин
остаток

1500
тестов

10 трейсов
на тест

$$1500 \times 10 / 300 = 50 \text{ rps}$$

Динамически регулируем время на сбор данных

Список запусков теста

Запуск теста

Запуск теста

Запуск теста

Запуск теста

1. Делаем окно запусков теста (например 5)

2. Берем множество сервисов из всех запусков

Не решает проблему
полностью, но **значительно**
сглаживает углы

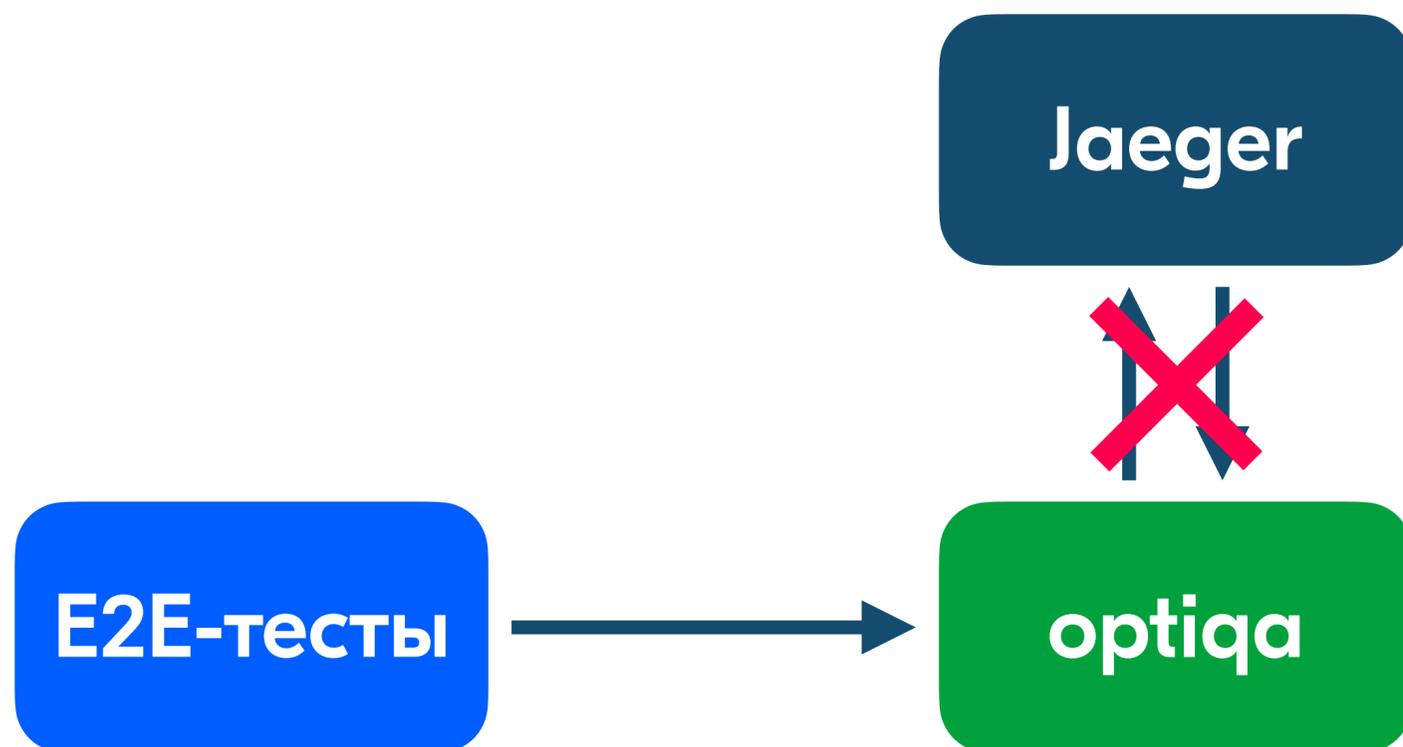
6

Результаты

~~6~~ 5.2

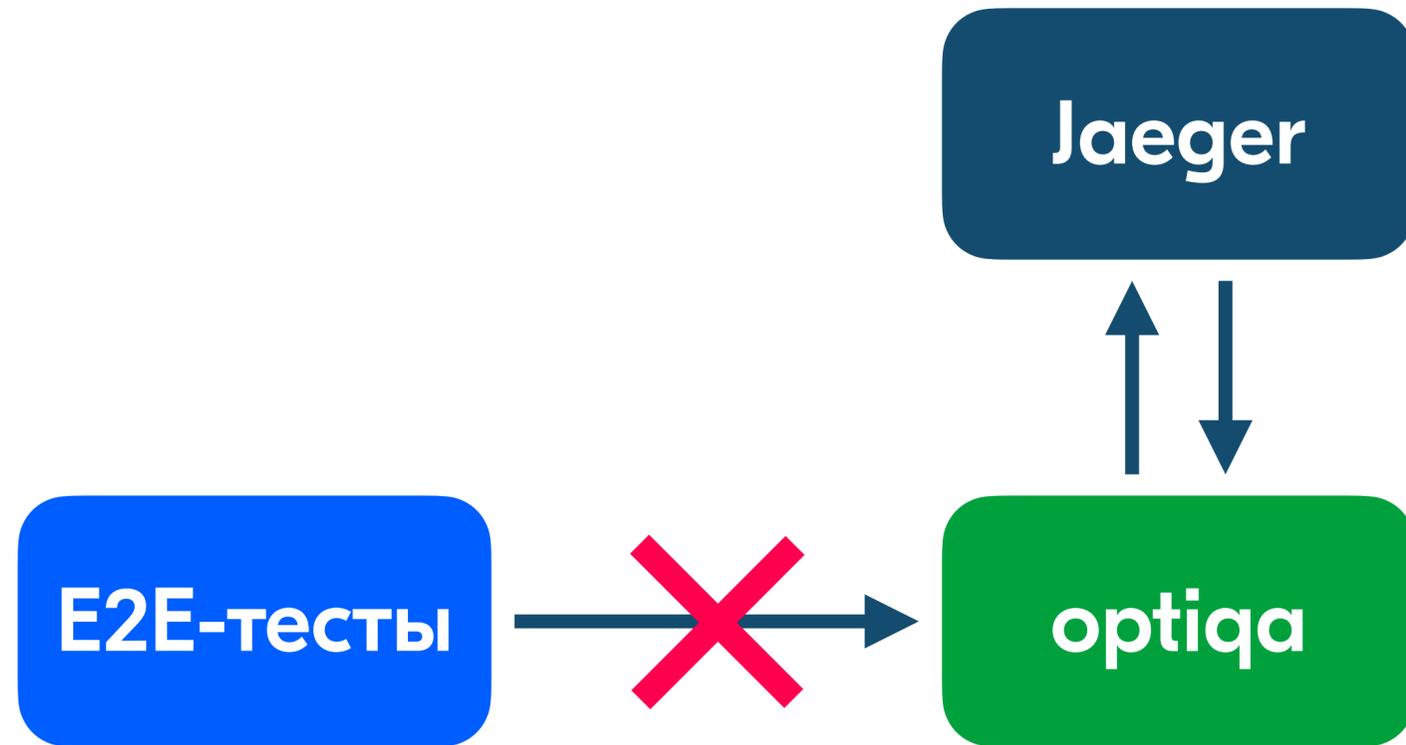
Риски

Jaeger-у плохо



1. Оооочень часто приходят неполные трейсы

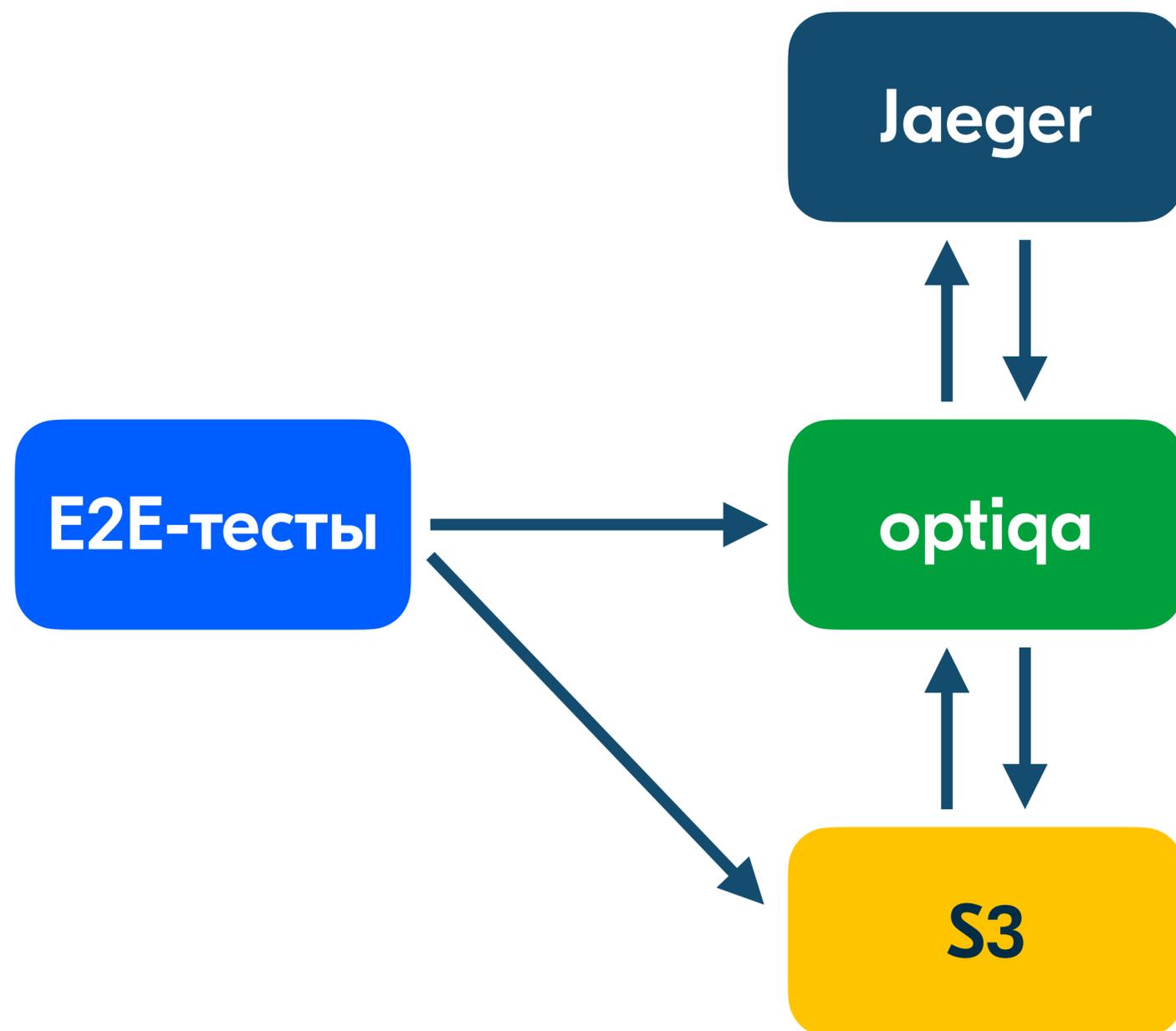
2. Выдаем невалидные данные



1. Невозможно забрать данные из оптики

2. Нет понимания что запускать





1. Заранее сохраняем слепок данных в S3

2. Если проблемы с Jaeger — берем подготовленные данные из S3

3. Если оптика недоступна — тесты берут подготовленные данные из S3

1. Нет оптики — нет и релизов



2. Возможны потери данных



3. Лежит Jaeger — нет обновления данных

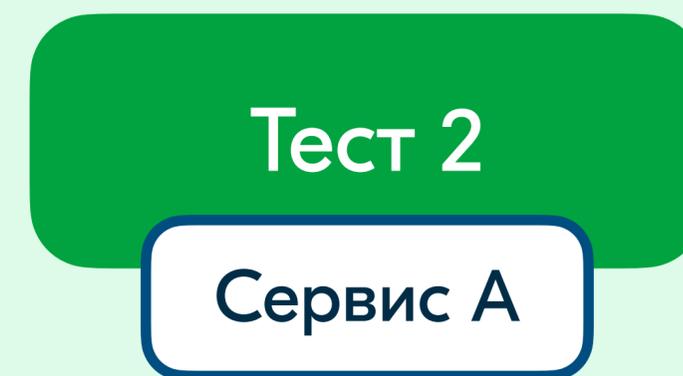
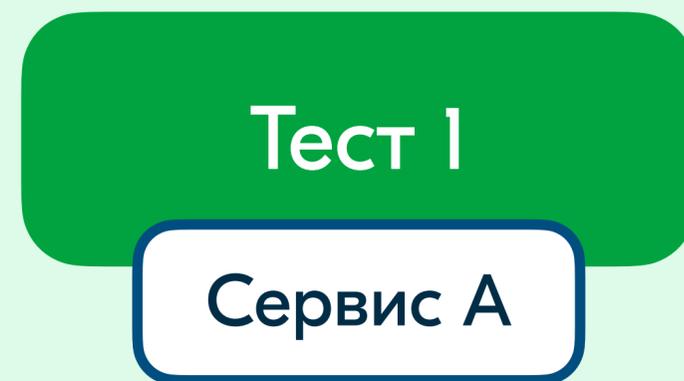
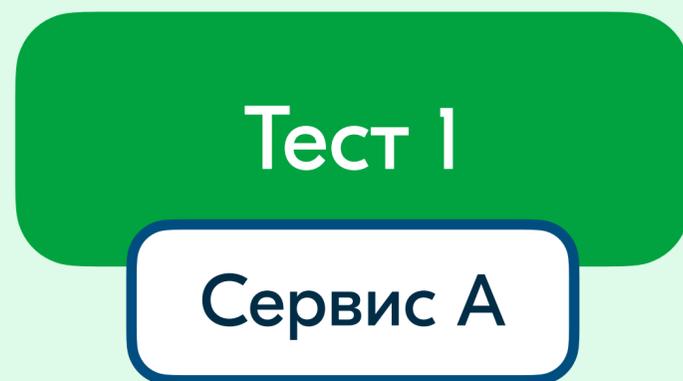
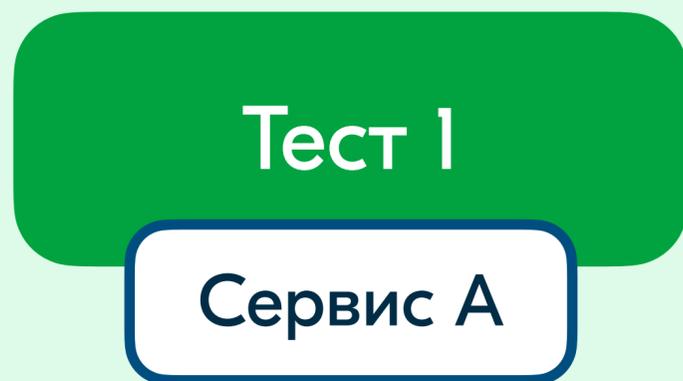


1. Нет оптики — возвращаем старые данные

2. Лежит Jaeger — возвращаем старые данные

3. Долгое обновление данных

Список запусков теста



1. Переименовали тест

2. Сделали прогон и получили данные

3. Запросили тесты по сервису А

4. Получили и Тест 1 и Тест 2

6

Результаты

1. Автоматическая разметка тестов
2. Добились доверия к разметке (с пружинами)
3. Снизили нагрузку от тестов
4. Уменьшили время прогонов тестов



5. Внедрили блокирующие тесты

6. Получили статистику по тестам для сервисов



399 тестов на СБП платежи

Тесты на сервис, который общается с СБП напрямую

Запуск по маркам

Количество: **399**

Время: **~4 минуты**

Запуск по трейсам

Количество: **163**

Время: **~2 минуты**

Поиска теста по traceID

Выберите действие

Поиск теста по traceId

Поиск связи тест-сервис

Trace ID

06e7ffe1b8225e04

Поиск

Результат

```
{
  "test": "test_2",
  "project": "project1",
  "createdAt": "2023-09-15 19:35:26"
}
```

Поиск связи тест—сервис

Выберите действие

Поиск теста по traceId

Поиск связи тест-сервис

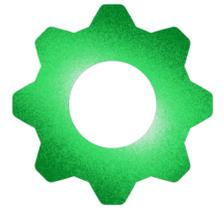
Название сервиса
service_1

Название теста
test_1

Поиск

Результат

handler	parameter	serviceName	testName
/handler/1	—	service_1	test_1
/handler/2	—	service_1	test_1



Ozon Fintech v2

400+

релизов в неделю

250+

сервисов

1000+

прогонов в неделю

2500+

автотестов

~10-13 мин

прогон по сервисам

Room to improve

1. Статистики по каждому тесту в UI

2. Статистика по стабильности сервисов в СТГ

3. Более умное определение тестов для сервисов



Хотим так же, но нет трейсинга

Можно использовать централизованные логи (Graylog)

- A. В тестах генерить уникальный идентификатор запроса
- B. Внедряем логирование идентификатора запроса в сервисы
- C. Забираем данные из системы логирования



Язык программирования и тестовый раннер

А. Возможность динамически выбрать, какие тесты запустить

В. Перед запуском сделать выборку и запустить тесты по имени



7

Выводы

1. Автоматически получать список тестов для сервиса — можно

- Уменьшает количество запускаемых тестов
- Уменьшает время прогона
- Повышает доверие к тестам



2. Но это дорого

- Нужна система трейсинга
- 14 чел/дней на MVP
- 2 чел/месяц на полностью готовое решение



3. Есть риски

- Любая ошибка в сервисе ведет к потере тестов
- Долгое обновление тестов



ozon{tech

Спасибо за ВНИМАНИЕ

Алексей Романов, ведущий инженер
по АТ в Ozon.Fintech

lr.romanov1999@mail.ru

 @lexcorp



Алексей Романов

 @lexcorp