

# DATABASE DELIVERY: THE BIG PROBLEM

tver  
vo/

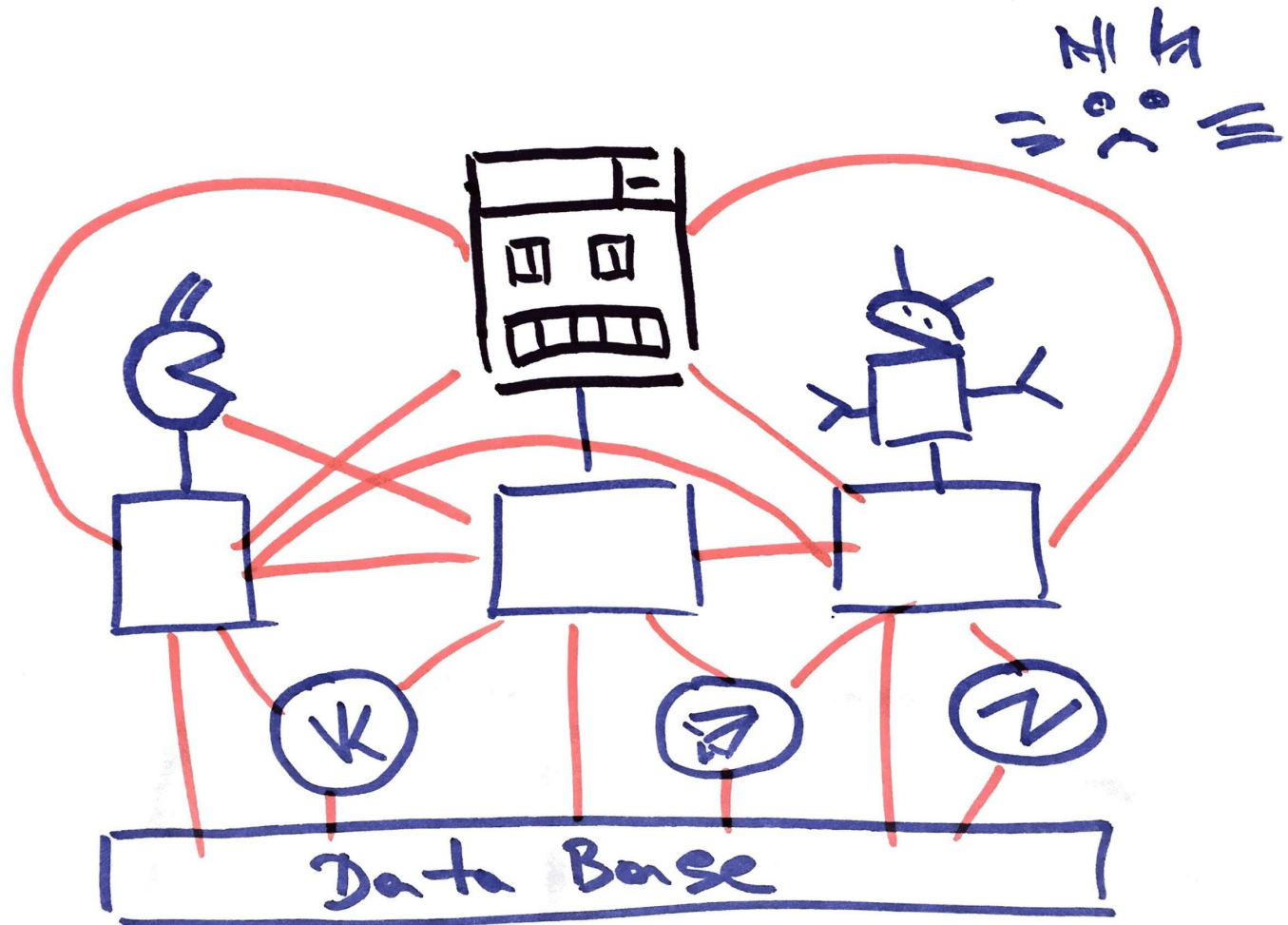


РОМАН ГОРДЕЕВ

# ЧТО БУДЕТ?

- 1. Рассмотрим основные проблемы доставки БД**
- 2. Изучим основные принципы и подходы управления схемой БД а также практики доставки БД**
- 3. Продемонстрируем практический пример с использованием Hqulbase**

КАК И ЛЮБАЯ ПОДОБНАЯ ИСТОРИЯ, ЭТА  
ИСТОРИЯ НАЧИНАЕТСЯ С  
ПРИЛОЖЕНИЯ...



КАКИЕ ВЫЗОВЫ?

1/ МНОЖЕСТВО РАЗЛИЧНЫХ ОКРУЖЕНИЙ,  
РАЗЛИЧАЮЩИХСЯ ПО КОНФИГУРАЦИИ,  
ТРЕБОВАНИЯМ И ДАННЫМ

2 / МНОГО УЧАСТНИКОВ, ОДНОВРЕМЕННО  
МОДИФИЦИРУЮЩИХ ПРИЛОЖЕНИЕ И  
НИЖЕЛЕЖАЩИЕ ДАННЫЕ И СТРУКТУРЫ

3 / НАЛИЧИЕ БОЛЬШОГО (ИЛИ НЕ ОЧЕНЬ)  
КОЛИЧЕСТВА ПРОДУКТОВЫХ БД

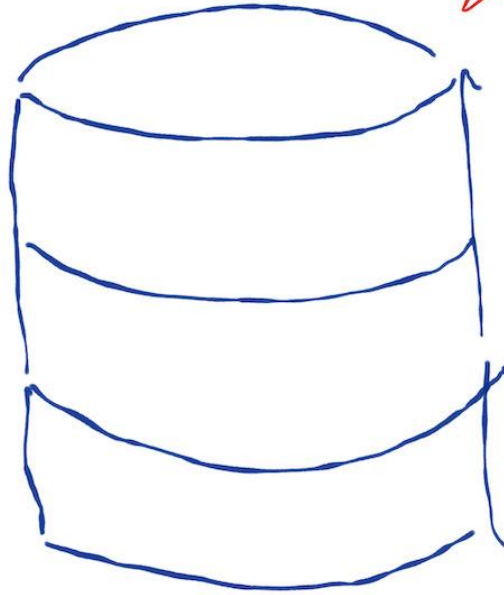


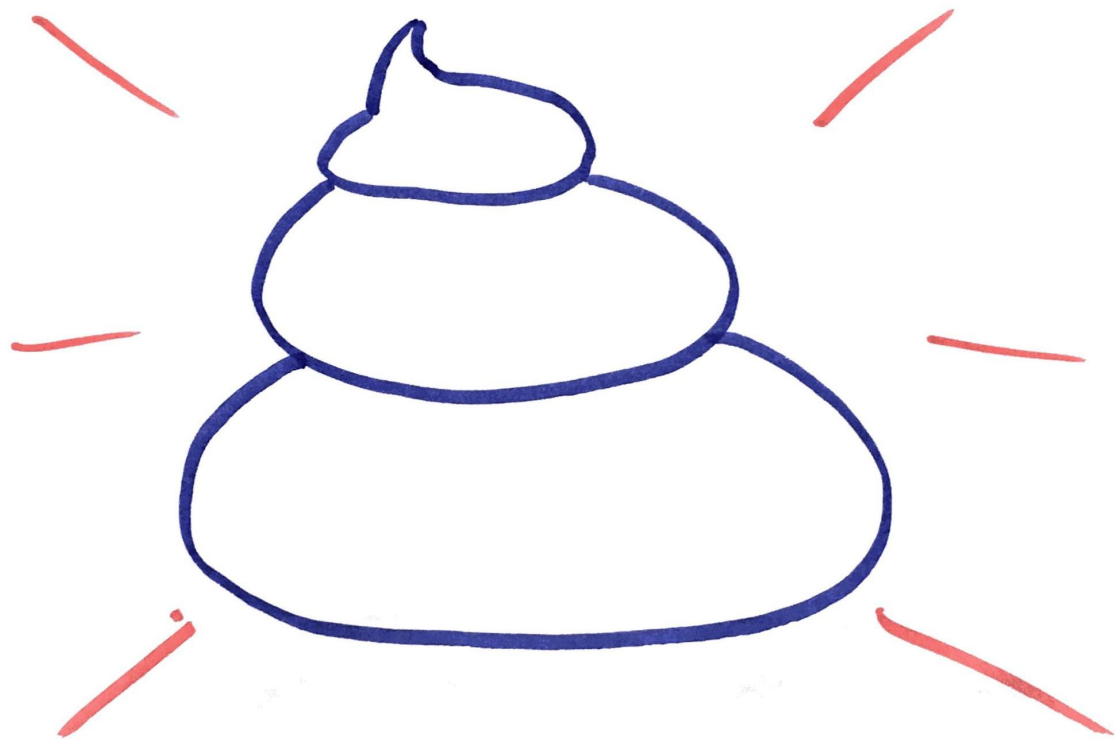
ЧТО МЕШАЕТ?

1/ ДАННЫЕ В БД

## 2 / ИНТЕГРАЦИЯ НА УРОВНЕ БД

Integration DB





# 3 / НЕСОГЛАСОВАННОСТЬ РАЗЛИЧНЫХ ОКРУЖЕНИЙ

# 4 / КОНФЛИКТЫ ПРИ ВЗАИМНЫХ МОДИФИКАЦИЯХ СХЕМЫ БД

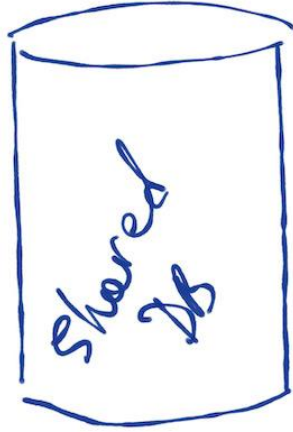
ЧТО МОЖНО СДЕЛАТЬ?

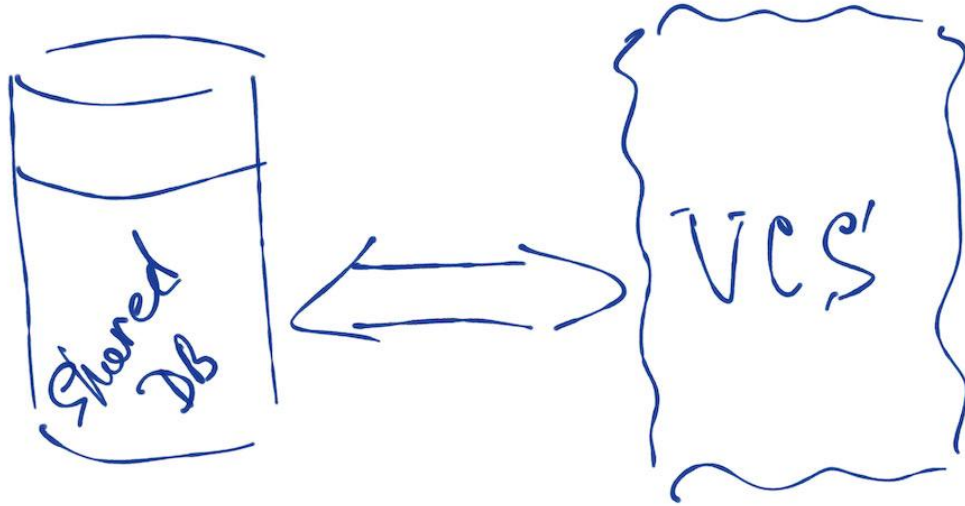


DATABASE AS A CODE (СХЕМА, СЛОВАРИ,  
ТЕСТОВЫЕ ДАННЫЕ)

ДОСТАВКА ИЗМЕНЕНИЙ БД ДОЛЖНА БЫТЬ  
ВКЛЮЧЕНА В ПРОЦЕСС ИНТЕГРАЦИОННОГО  
ТЕСТИРОВАНИЯ

КАЖДОМУ РАЗРАБОТЧИКУ СВОЯ БД



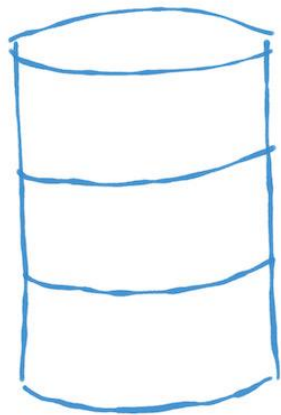


ПЕРМАНЕНТНО ПОДДЕРЖИВАТЬ  
СОГЛАСОВАННОСТЬ МОДЕЛИ ПРИЛОЖЕНИЯ  
И СХЕМЫ БД

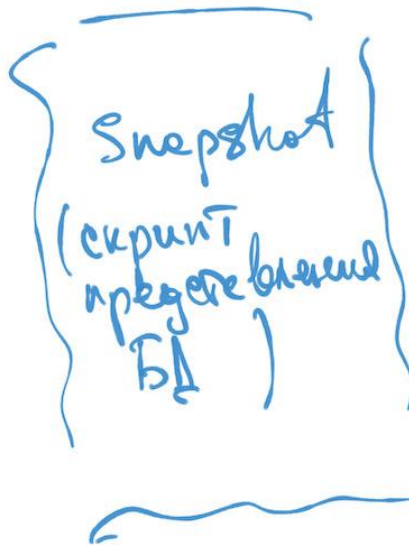
# ПОДХОДЫ К УПРАВЛЕНИЮ ДОСТАВКОЙ ИЗМЕНЕНИЙ БД

НА ОСНОВЕ ЭТАЛОНА (STATE-BASED)

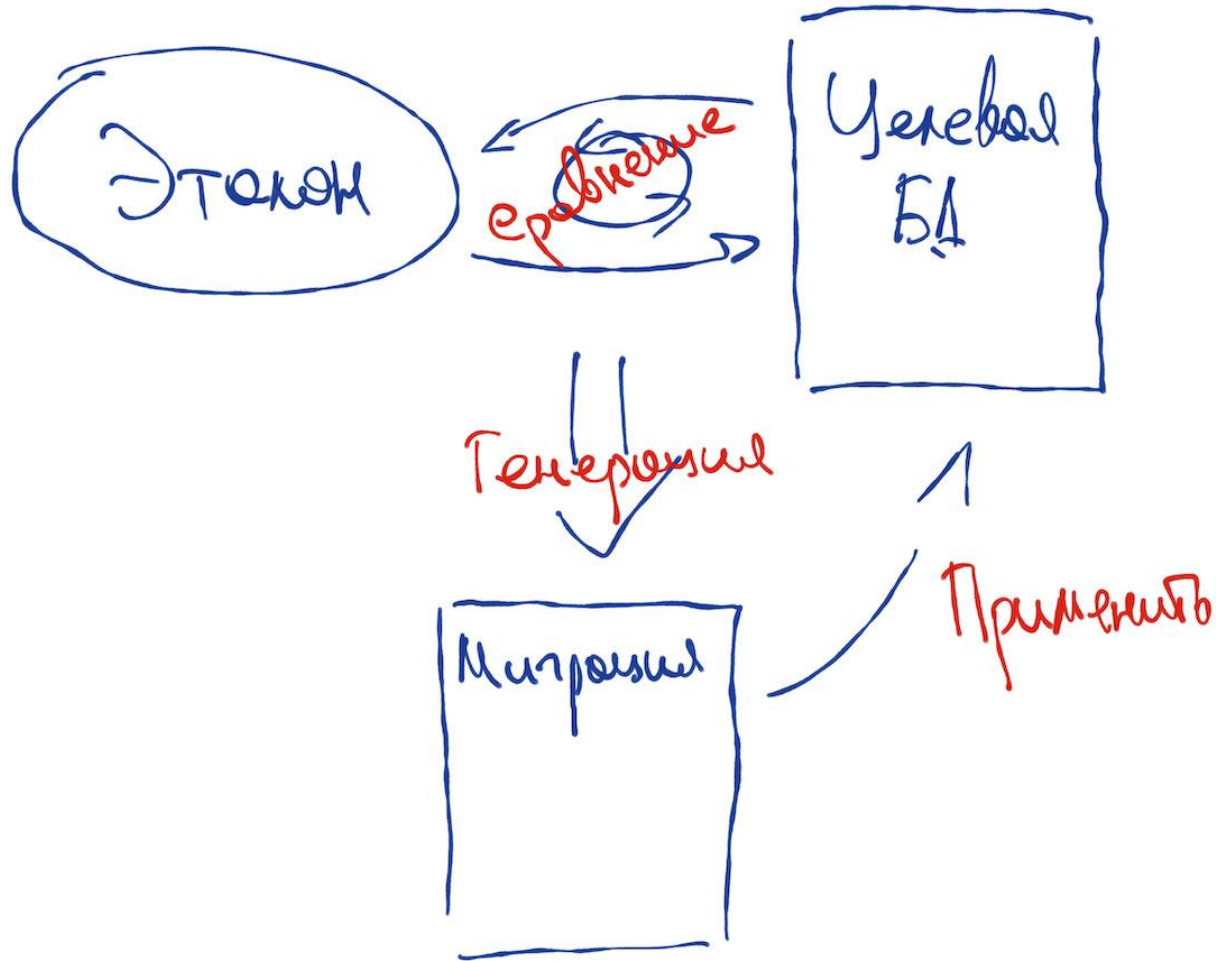




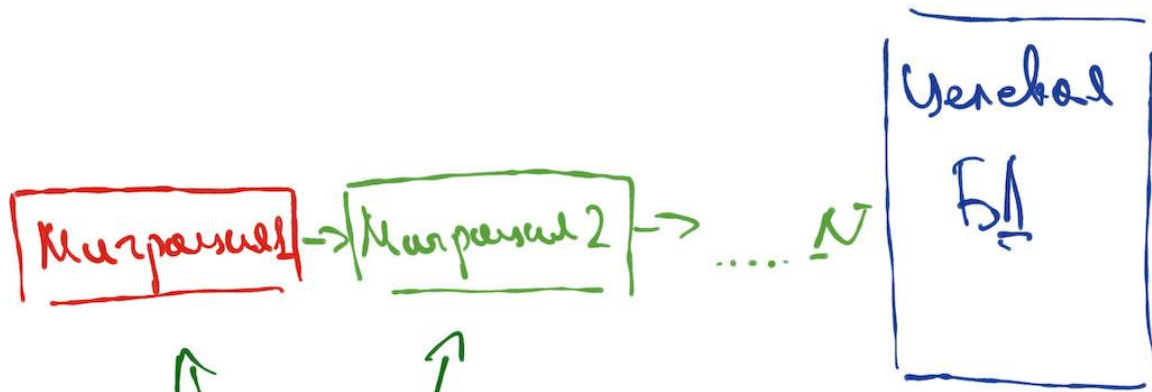
Референсная  
БД



The  
Origin



НА ОСНОВЕ ПОСЛЕДОВАТЕЛЬНОГО  
ПРИМЕНЕНИЯ ИНКРЕМЕНТАЛЬНЫХ  
ИЗМЕНЕНИЙ (MIGRATION-BASED)



Создается разработчиком

# ПРАКТИЧЕСКИЙ ПРИМЕР

# ЛЕГЕНДА

- 1. Переписываем легаси приложение с БД**
- 2. У нового приложения есть изменения в схеме БД**
- 3. В новом приложении хотим реализовать database delivery на основе миграций**

ШАГ 1. СИНХРОНИЗИРУЕМ СХЕМУ ЛЕГАСИ БД  
И СНЕПШОТА НОВОГО ПРИЛОЖЕНИЯ

ШАГ 2. ФОРМИРУЕМ НАЧАЛЬНУЮ  
МИГРАЦИЮ, ПОЛНОСТЬЮ ОПИСЫВАЮЩУЮ  
АКТУАЛЬНУЮ СХЕМУ БД



```
./gradlew liquibaseGenerateChangeLog
```

Project

Project



resources

db.changeLog

data

initial

20210212001716\_changeLog.xml

db.changeLog-master.xml

static



Structure

ШАГ 3. ИНИЦИАЛИЗИРУЕМ СХЕМУ НОВОГО  
СНЕПШОТА И ПРОВЕРЯЕМ ОТСУТСТВИЕ  
РАСХОЖДЕНИЙ С ЦЕЛЕВОЙ БД

# ШАГ 4. НАСТРАИВАЕМ ИНТЕГРАЦИОННЫЕ ТЕСТЫ

ШАГ 5. СОЗДАЕМ МИГРАЦИИ СЛОВАРЕЙ

Project ▾



▾ initial

▾ data

→ countries.csv

→ languages.csv

user\_language.csv

users.csv

init.xml

20210212001716\_changelog.xml

db.changelog-master.xml

ШАГ 6. СОЗДАЕМ МИГРАЦИИ ДАННЫХ ДЛЯ  
ТЕСТОВЫХ И ЛОКАЛЬНЫХ СРЕД

Project ▾



▾ initial

▾ data

countries.csv

languages.csv

user\_language.csv

users.csv

init.xml

20210212001716\_changelog.xml

db.changelog-master.xml



# ШАГ 7. РЕФАКТОРИНГ

```
./gradlew liquibaseDiffChangeLog -PrunList=diffLog
```

# REFERENCES

- Vladimir Khorikov. Database versioning best practices  
<https://enterprisecraftsmanship.com/posts/database-versioning-best-practices/>
- Scott W. Ambler, Pramod J. Sadalage. Refactoring Databases: Evolutionary Database Design  
<https://www.ozon.ru/product/refaktoring-baz-dannyh-evolyutsionnoe-proektirovanie-162984061>
- <https://docs.liquibase.com/home.html>

THAT ALL FOLKS!

[HTTPS://GITHUB.COM/RTK-IT/MIGRATIONS](https://github.com/rtk-it/migrations)