

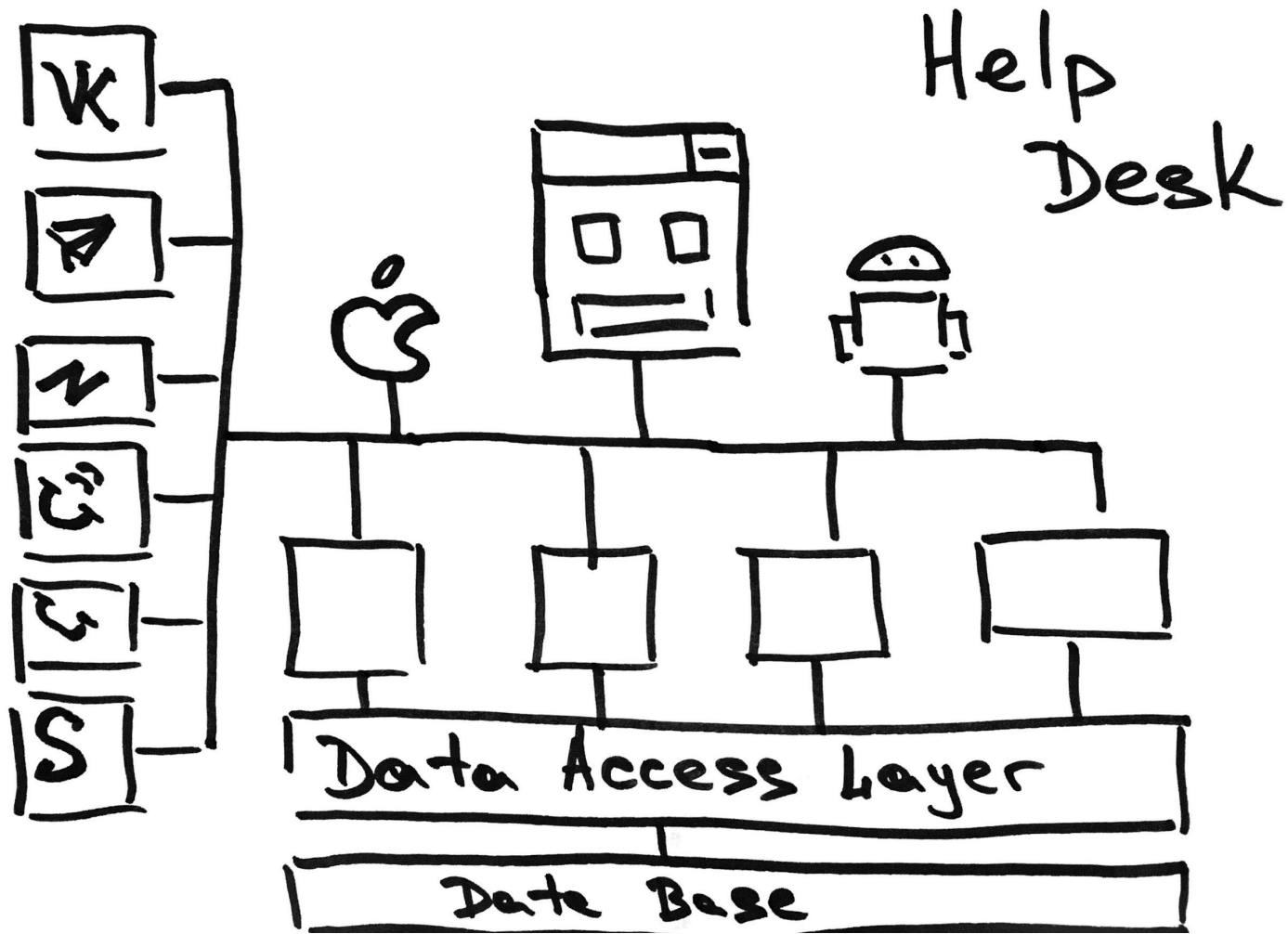
# НА ПУТИ К РАСПРЕДЕЛЕННОЙ СИСТЕМЕ



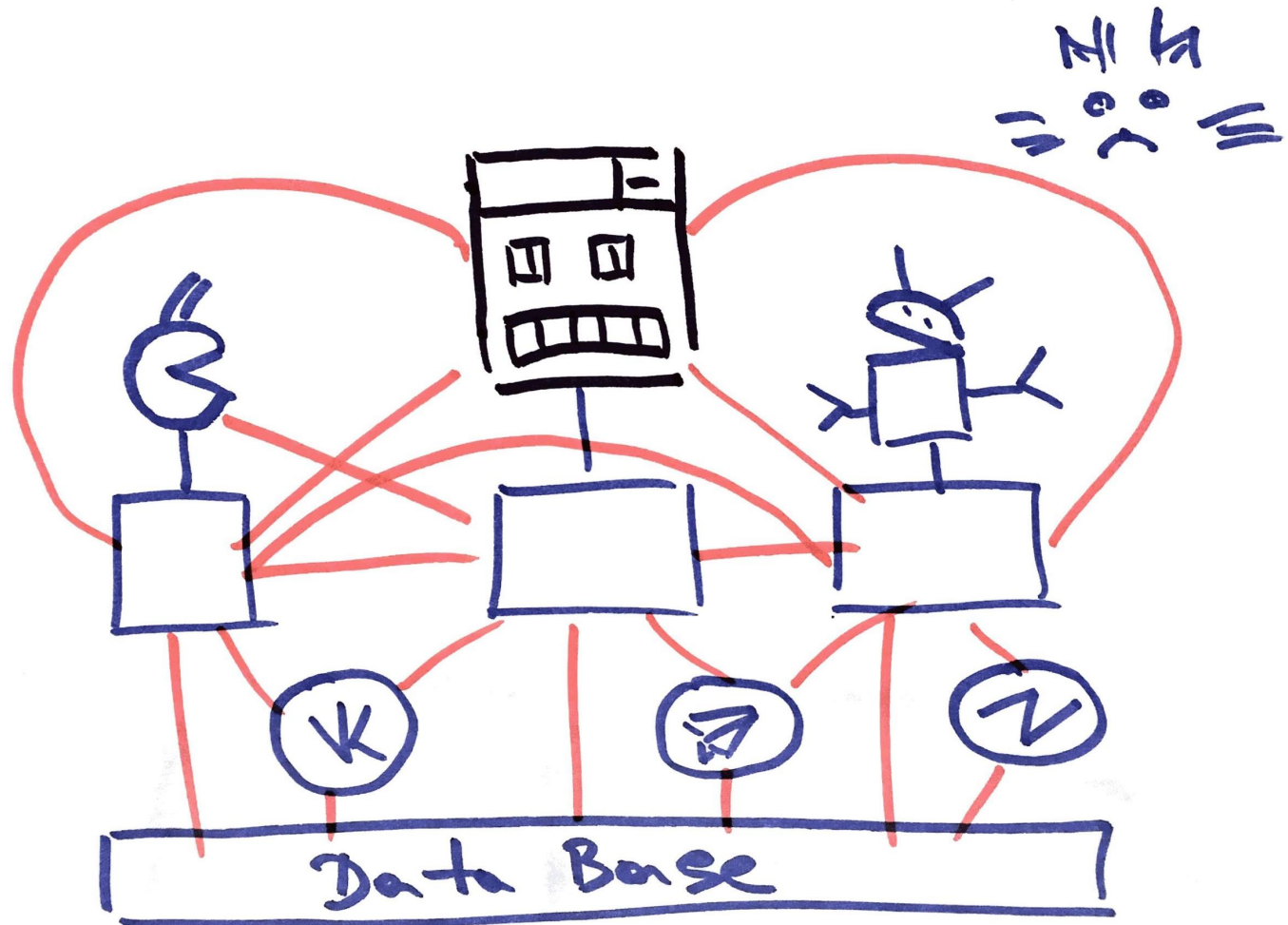
РОМАН ГОРДЕЕВ

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

“ТАКОГО ОБАЯНЬЯ ТЫ ИСПОЛНЕН...”



“НО МЫ ОТДЁРНЕМ ЗАНАВЕС И ПОКАЖЕМ ВАМ  
КАРТИНУ...”



# ЦЕЛИ:

- 1. Как работать с унаследованным кодом**
- 2. Чеклист для ORM**
- 3. Простые решения для непростых ситуаций**
- 4. Наивное проектирование**
- 5. Инструментарий**

КАК ОРГАНИЗОВАТЬ КОД?



- МОНРЕПОЗИТОРИЙ?
- РЕПОЗИТОРИЙ НА КАЖДЫЙ ПОДПРОЕКТ?

APP 1  
Repo 1



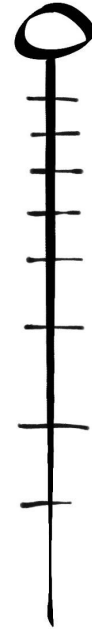
APP 2  
Repo 2



...  
...



APP N  
Repo N



History

## Artifact Repo

pom.xml +  
Dockerfile

pom.xml +  
Dockerfile

...

pom.xml +  
Dockerfile

app1.jar

app2.jar

...

appN.jar

## Docker Registry

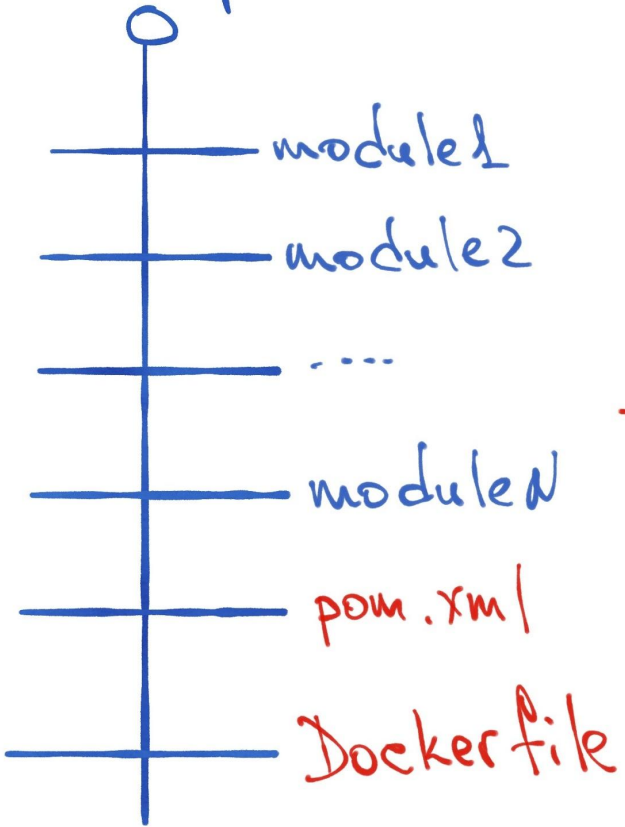
dockerImage1

dockerImage2

....

dockerImageN

Git Repo



app.jar  
dockerImage

- УДОБСТВО ЭКСПЛУАТАЦИИ
  - ПРОСТОТА
- ВОЗМОЖНОСТЬ РАЗВИТИЯ

ПРОСТЫЕ ТРУДНОСТИ...

ЧЕК-ЛИСТ ПОЛЬЗОВАТЕЛЯ

ORM

ПО УМОЛЧАНИЮ ИСПОЛЬЗУЙТЕ

LAZY FETCHING



| 71 | Association type | Default fetching policy |
|----|------------------|-------------------------|
| 72 | =====            | =====                   |
| 73 | @OneToMany       | LAZY                    |
| 74 | @ManyToMany      | LAZY                    |
| 75 | @ManyToOne       | EAGER                   |
| 76 | @OneToOne        | EAGER                   |
| 77 |                  |                         |

НИКОГДА НЕ ИСПОЛЬЗУЙТЕ

TABLE GENERATOR!

```
15 @Entity
16 @Table(name = "messages")
17 public class Message implements Serializable {
18
19     @Id
20     @GeneratedValue(strategy=GenerationType.TABLE)
21     private Long id;
22
23
```



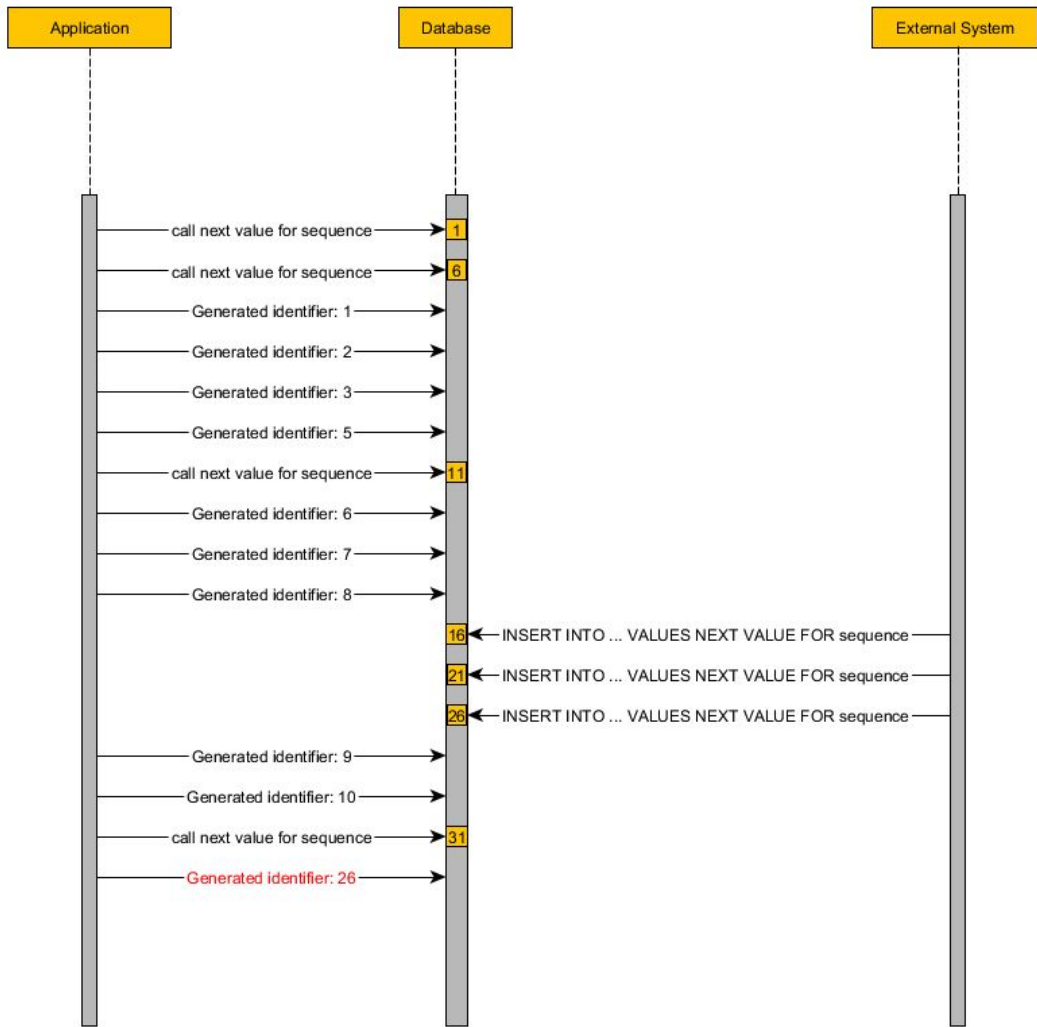
```
1 SELECT tbl.next_val
2 FROM hibernate_sequences tbl
3 WHERE tbl.sequence_name=default
4 FOR UPDATE
5
6 INSERT INTO hibernate_sequences (sequence_name, next_val)
7 VALUES (default, 1)
8
9 UPDATE hibernate_sequences SET next_val=2
10 WHERE next_val=1 AND sequence_name=default
11
12 SELECT tbl.next_val
13 FROM hibernate_sequences tbl
14 WHERE tbl.sequence_name=default
15 FOR UPDATE
16
17 UPDATE hibernate_sequences SET next_val=3
18 WHERE next_val=2 AND sequence_name=default
19
20 INSERT INTO messages (id) values (1, 2)
```

ИСПОЛЬЗУЙТЕ ГЕНЕРАТОРЫ  
"ПОСЛЕДНЕГО ПОКОЛЕНИЯ"

```
92  
93 properties.put("hibernate.id.new_generator_mappings", "true");  
94
```

```
@Entity
@Table(name = "messages")
public class Message implements Serializable {

    @Id
    @GenericGenerator(
        name = "sequenceGenerator",
        strategy = "enhanced-sequence",
        parameters = {
            @org.hibernate.annotations.Parameter(
                name = "optimizer",
                value = "pooled" ←
            ),
            @org.hibernate.annotations.Parameter(
                name = "initial_value",
                value = "1"
            ),
            @org.hibernate.annotations.Parameter(
                name = "increment_size",
                value = "5" ←
            )
        }
    )
    @GeneratedValue(
        strategy = GenerationType.SEQUENCE,
        generator = "sequenceGenerator"
    )
    private Long id;
```

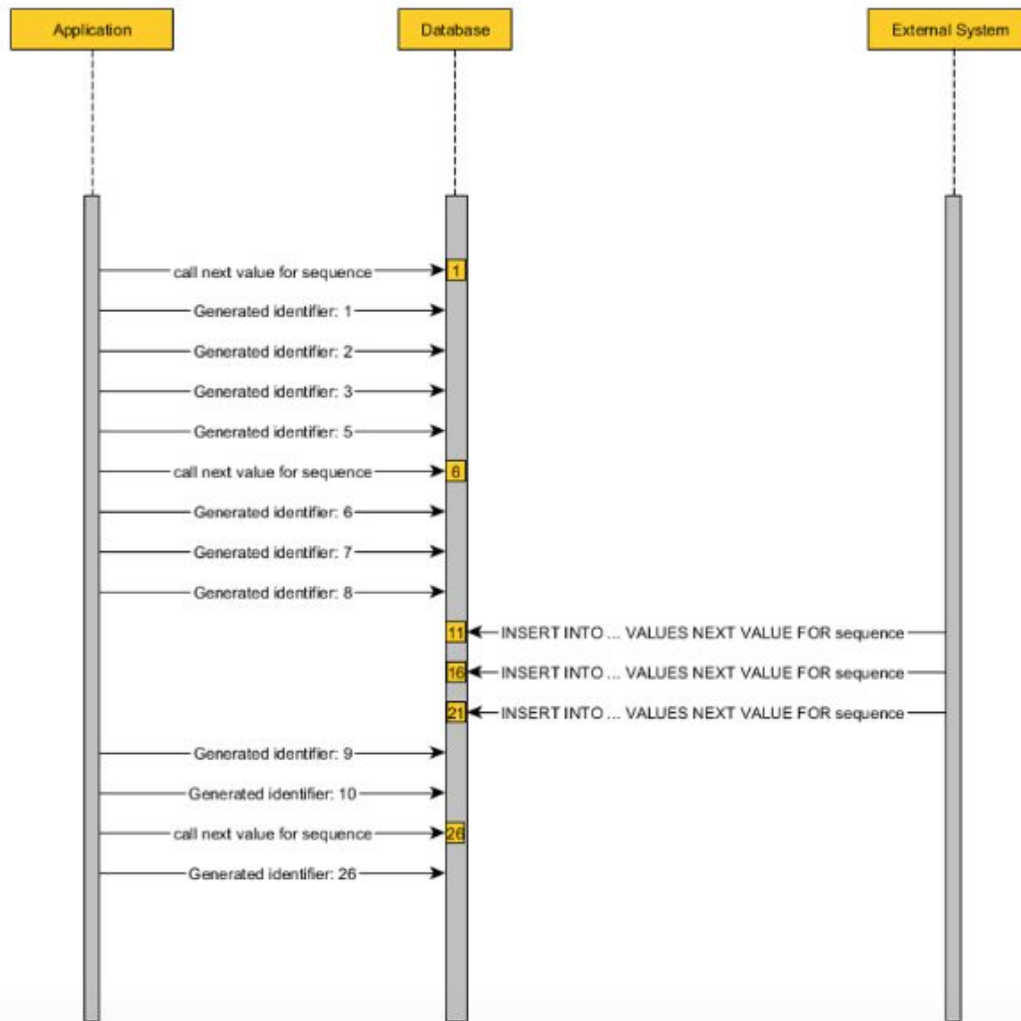




```
@Entity
@Table(name = "messages")
public class Message implements Serializable {

    @Id
    @GenericGenerator(
        name = "sequenceGenerator",
        strategy = "enhanced-sequence",
        parameters = {
            @org.hibernate.annotations.Parameter(
                name = "optimizer",
                value = "pooled-lo"
            ),
            @org.hibernate.annotations.Parameter(
                name = "initial_value",
                value = "1"
            ),
            @org.hibernate.annotations.Parameter(
                name = "increment_size",
                value = "5"
            )
        }
    )

    @GeneratedValue(
        strategy = GenerationType.SEQUENCE,
        generator = "sequenceGenerator"
    )
    private Long id;
```



СКОНФИГУРИРУЙТЕ

BATCHING

- HIBERNATE.JDBC.BATCH\_SIZE=30
- HIBERNATE.ORDER\_INSERTS=TRUE
- HIBERNATE.ORDER\_UPDATES=TRUE

Для отладки используйте  
механизмы логирования ORM и  
PROXY DATASOURCE/DRIVER

78

79

```
<logger name="org.hibernate.SQL" level="debug"/>
```

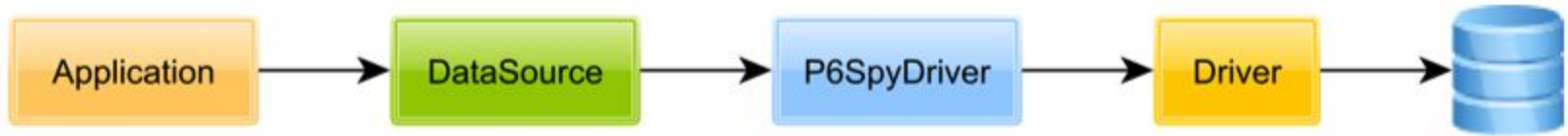
80

81

```
INSERT INTO messages (id, payload, type) VALUES (?, ?, ?)
```

82

# P6SPY



ЧТО ДЕЛАТЬ, ЕСЛИ ПРИЛОЖЕНИЕ  
"ТОРМОЗИТ"?



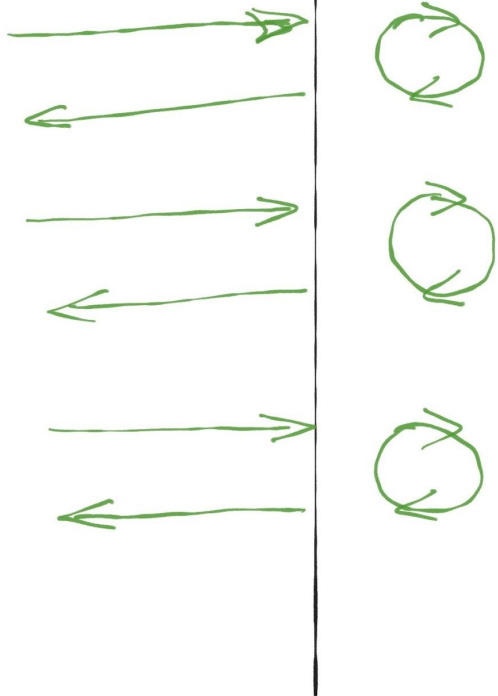
ИСПОЛЬЗУЙТЕ JMX!

```
83  
84 -Dcom.sun.management.jmxremote  
85  
86 jvisualvm  
87  
88 kill -s QUIT <pid>  
89  
90 docker kill -s QUIT <container-id>
```

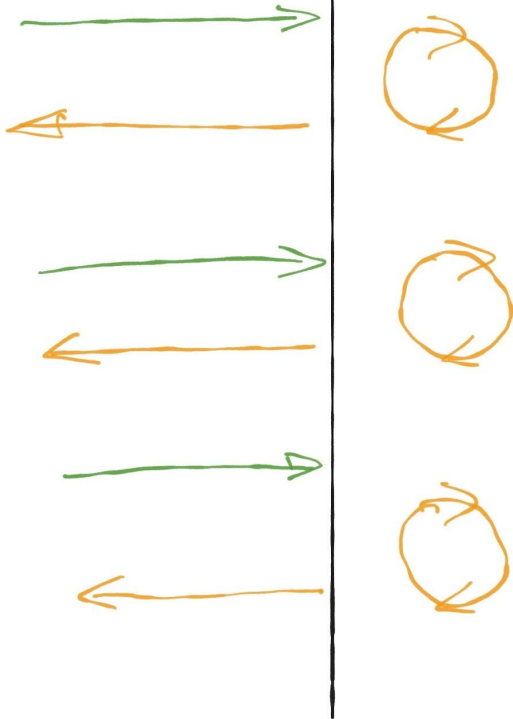
```
1 "http-nio-8080-exec-5051" #31419 daemon prio=5 os_prio=0 tid=0x00007f7548bee000 nid=0x2aed waiting for monitor entry [0x00007f
2   java.lang.Thread.State: BLOCKED (on object monitor)
3   → at ru.multicon.core.arm.service.NetworkStatusService.refreshNetworkStatus_aroundBody28(NetworkStatusService.java:258)
4   → - waiting to lock <0x00000000e8f42318> (a java.lang.Class for ru.multicon.core.arm.util.Blocking$AgentLock)
5   → at ru.multicon.core.arm.service.NetworkStatusService$AjcClosure29.run(NetworkStatusService.java:1)
6   → at org.aspectj.runtime.reflect.JoinPointImpl.proceed(JoinPointImpl.java:149)
7   → at ru.multicon.logging.LogEventAspect.logDependsOnLogType(LogEventAspect.java:73)
8   → at ru.multicon.logging.LogEventAspect.wrap(LogEventAspect.java:50)
9   → at ru.multicon.logging.LogEventAspect.ajc$inlineAccessMethod$workspace_multicon_logging_LogEventAspect$workspace_multicon_lo
10  → at ru.multicon.logging.LogEventAspect.logEventWrapMethod(LogEventAspect.java:45)
11  → at ru.multicon.core.arm.service.NetworkStatusService.refreshNetworkStatus(NetworkStatusService.java:256)
12  → at ru.multicon.core.arm.controller.ArmController.refreshNetworkStatus_aroundBody8(ArmController.java:132)
13  → at ru.multicon.core.arm.controller.ArmController$AjcClosure9.run(ArmController.java:1)
14  → at org.aspectj.runtime.reflect.JoinPointImpl.proceed(JoinPointImpl.java:149)
15  → at ru.multicon.logging.LogEventAspect.logDependsOnLogType(LogEventAspect.java:73)
16  → at ru.multicon.logging.LogEventAspect.wrap(LogEventAspect.java:50)
17  → at ru.multicon.logging.LogEventAspect.ajc$inlineAccessMethod$workspace_multicon_logging_LogEventAspect$workspace_multicon_lo
18  → at ru.multicon.logging.LogEventAspect.logEventWrapMethod(LogEventAspect.java:45)
19  → at ru.multicon.core.arm.controller.ArmController.refreshNetworkStatus(ArmController.java:131)
20  → at sun.reflect.GeneratedMethodAccessor2863.invoke(Unknown Source)
21  → at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
22  → at java.lang.reflect.Method.invoke(Method.java:498)
23  → at org.springframework.web.method.support.InvocableHandlerMethod.doInvoke(InvocableHandlerMethod.java:205)
24  → at org.springframework.web.method.support.InvocableHandlerMethod.invokeForRequest(InvocableHandlerMethod.java:133)
25  → at org.springframework.web.servlet.mvc.method.annotation.ServletInvocableHandlerMethod.invokeAndHandle(ServletInvocableHandl
26  → at org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.invokeHandlerMethod(RequestMappingHand
```

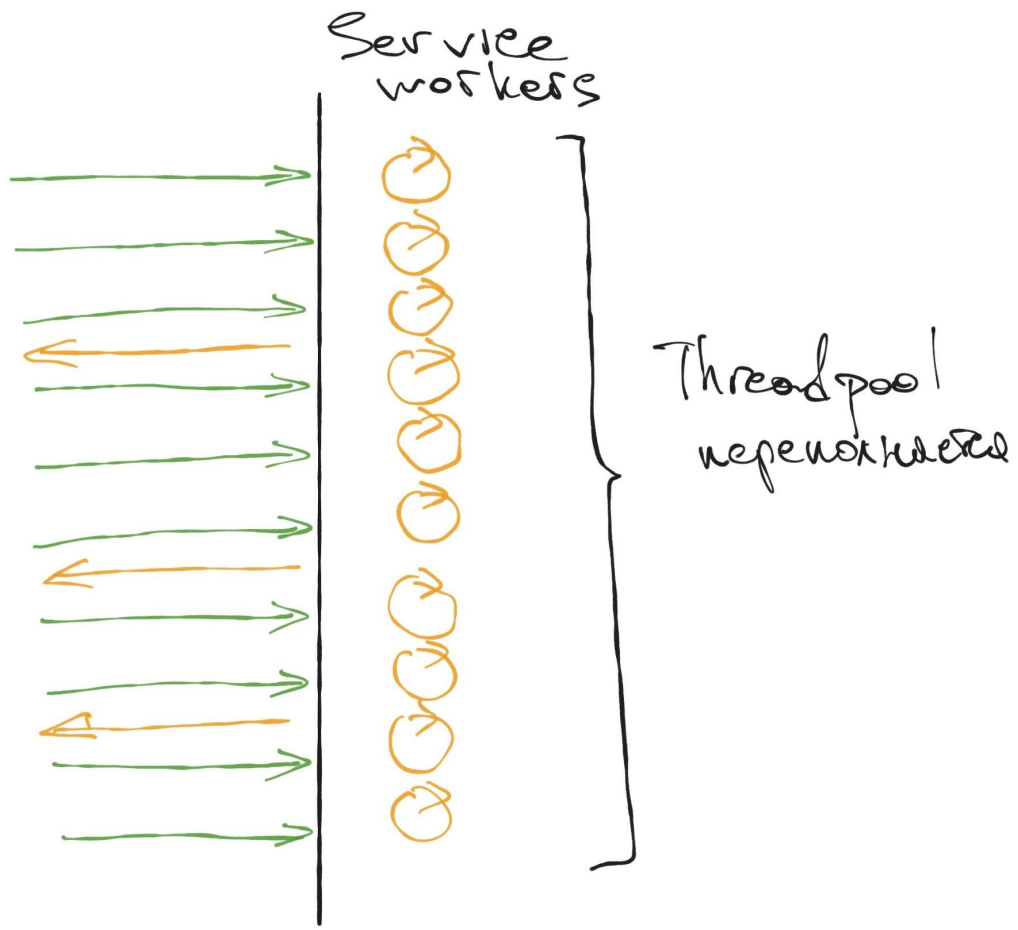
МАЙКЛ НЕЙГАРД. RELEASE IT! ПРОЕКТИРОВАНИЕ И  
ДИЗАЙН ПО ДЛЯ ТЕХ, КОМУ НЕ ВСЕ РАВНО

Service  
workers



Service  
workers

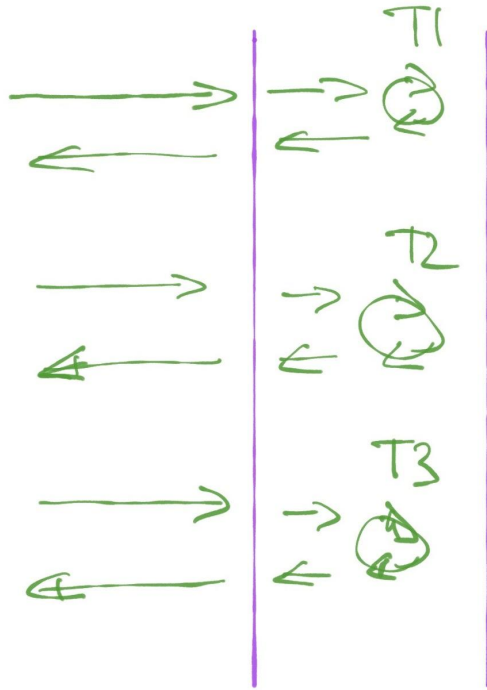




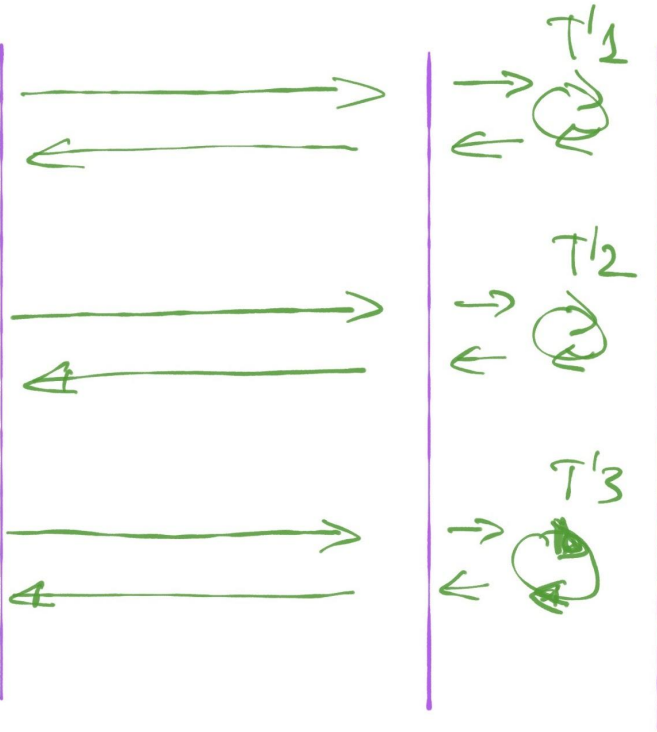
TIMEOUTS



# Service A

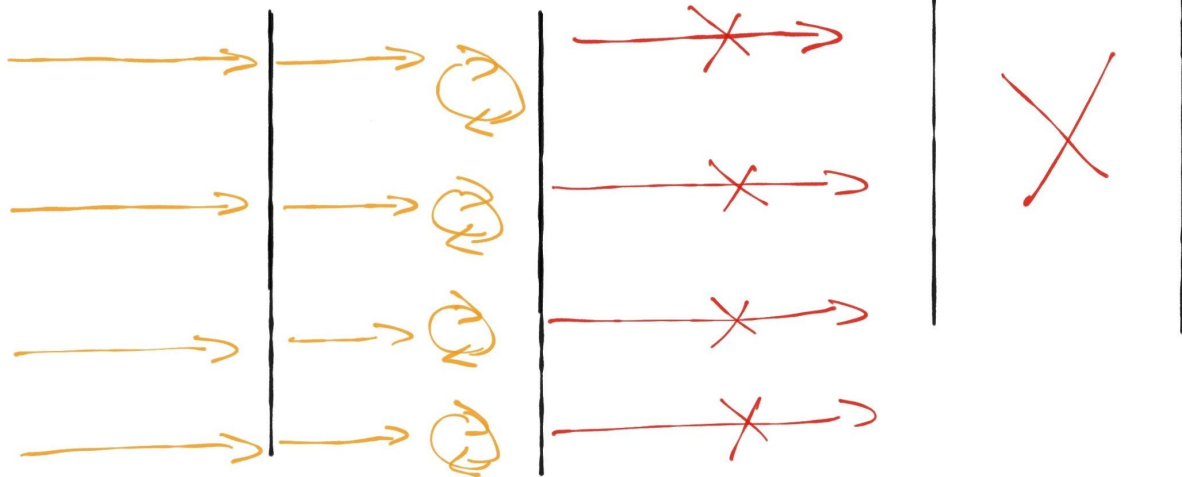


# Service B



Service A

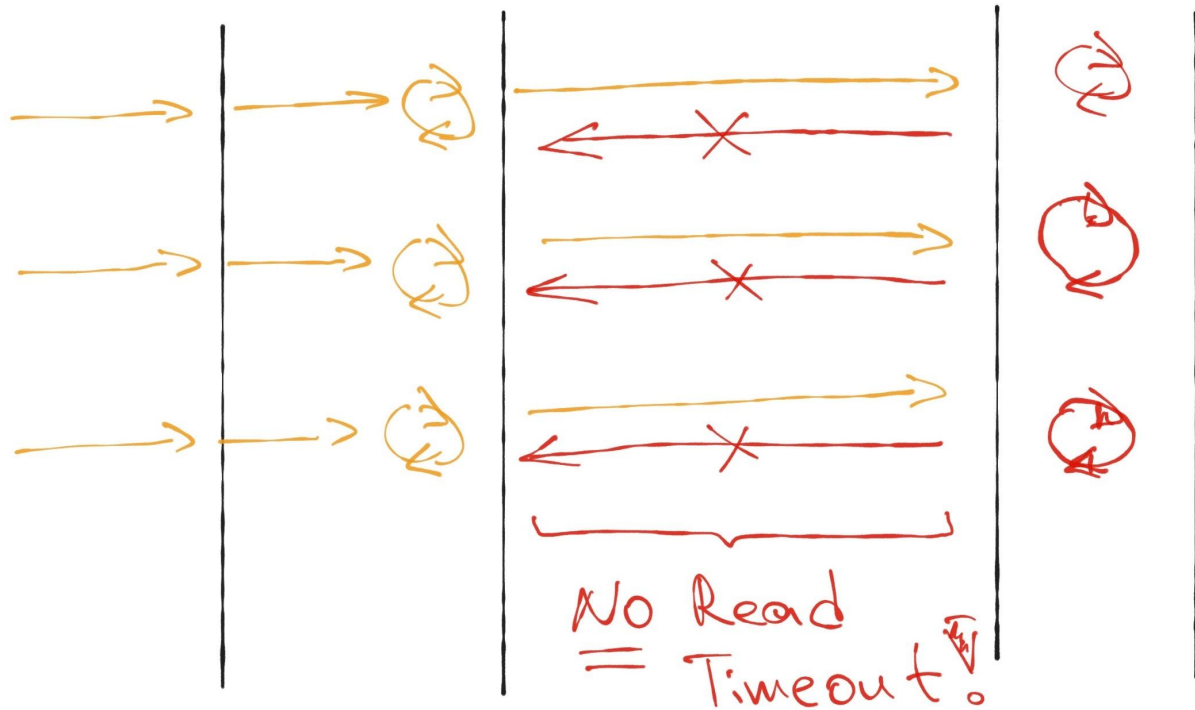
Service B

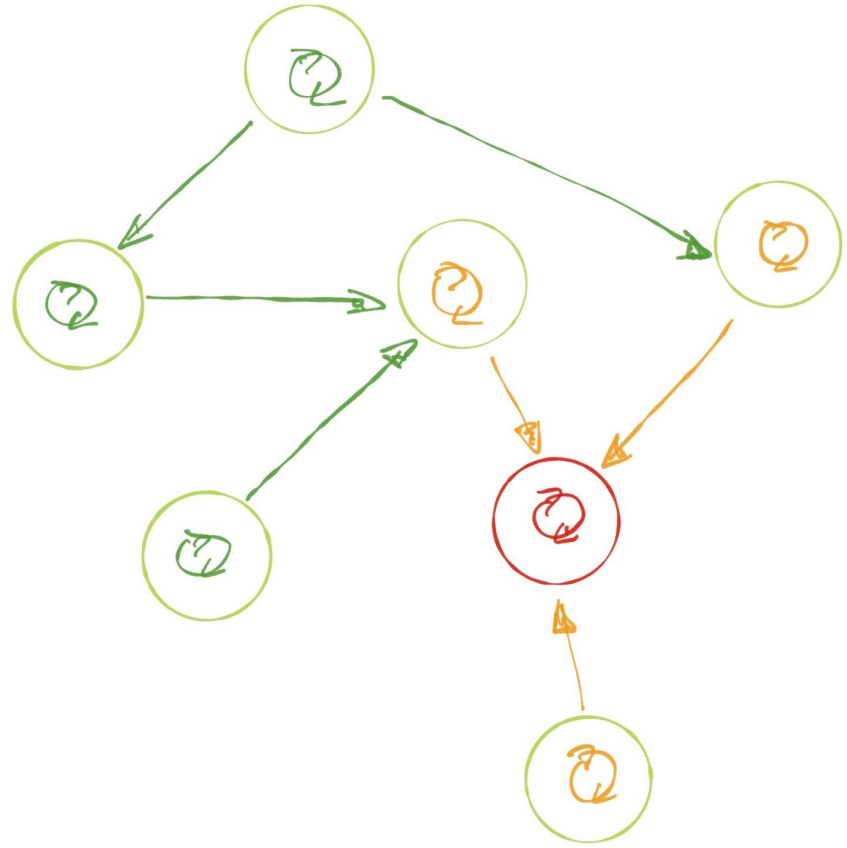


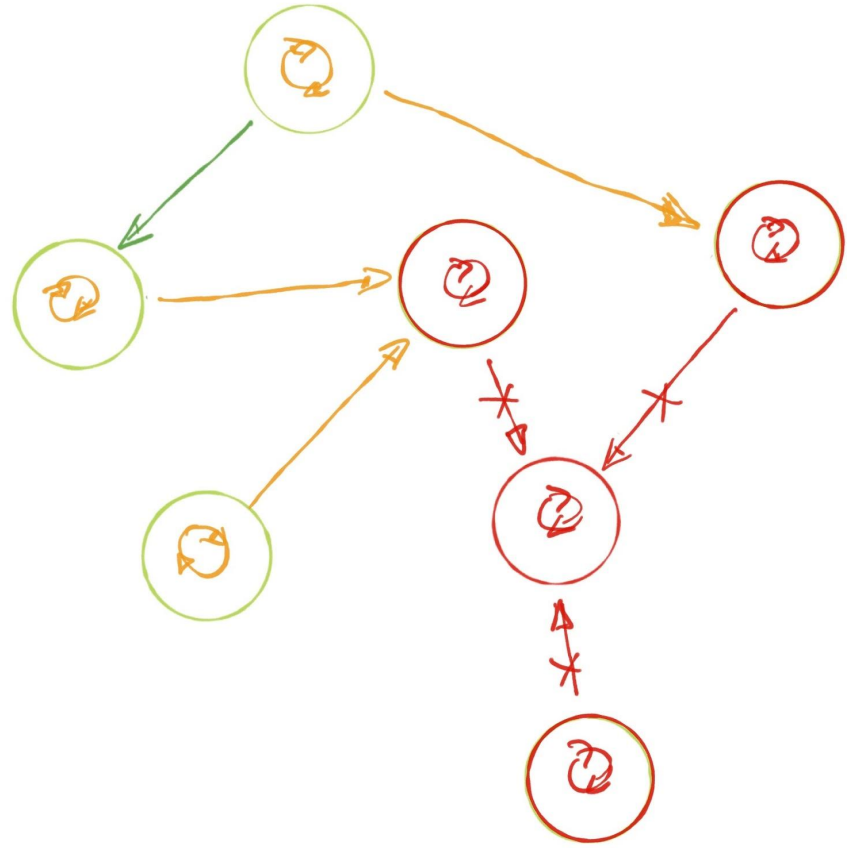
No Connection  
Timeout

Service A

Service B

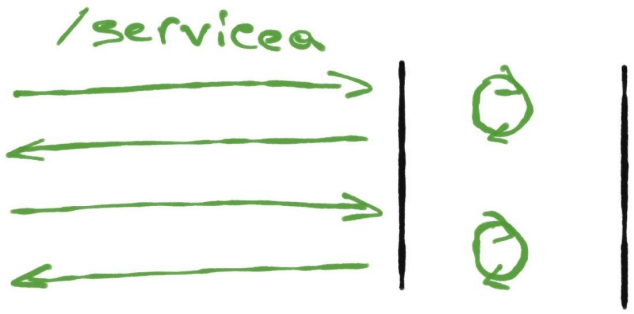
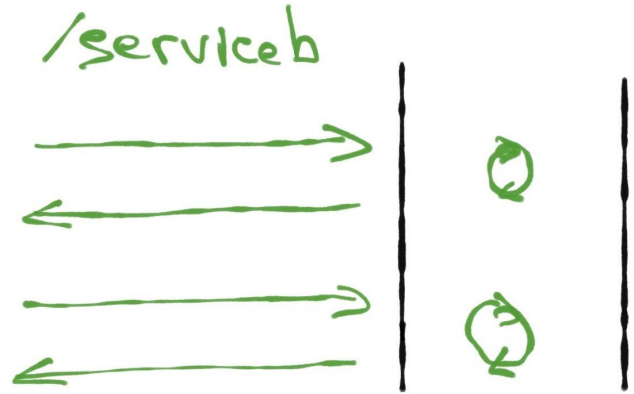
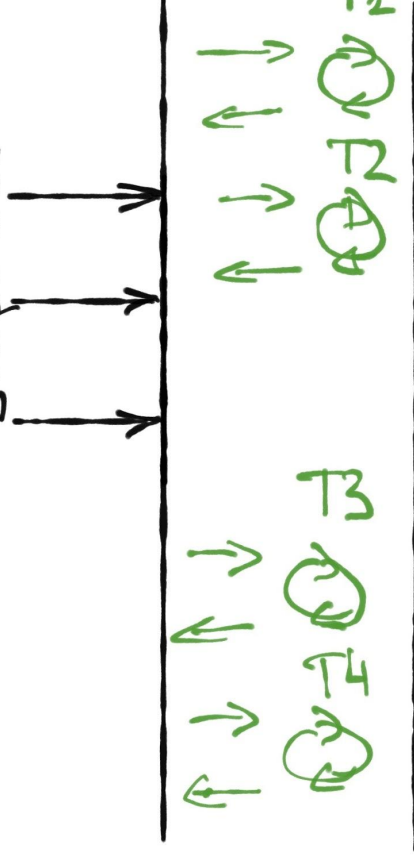


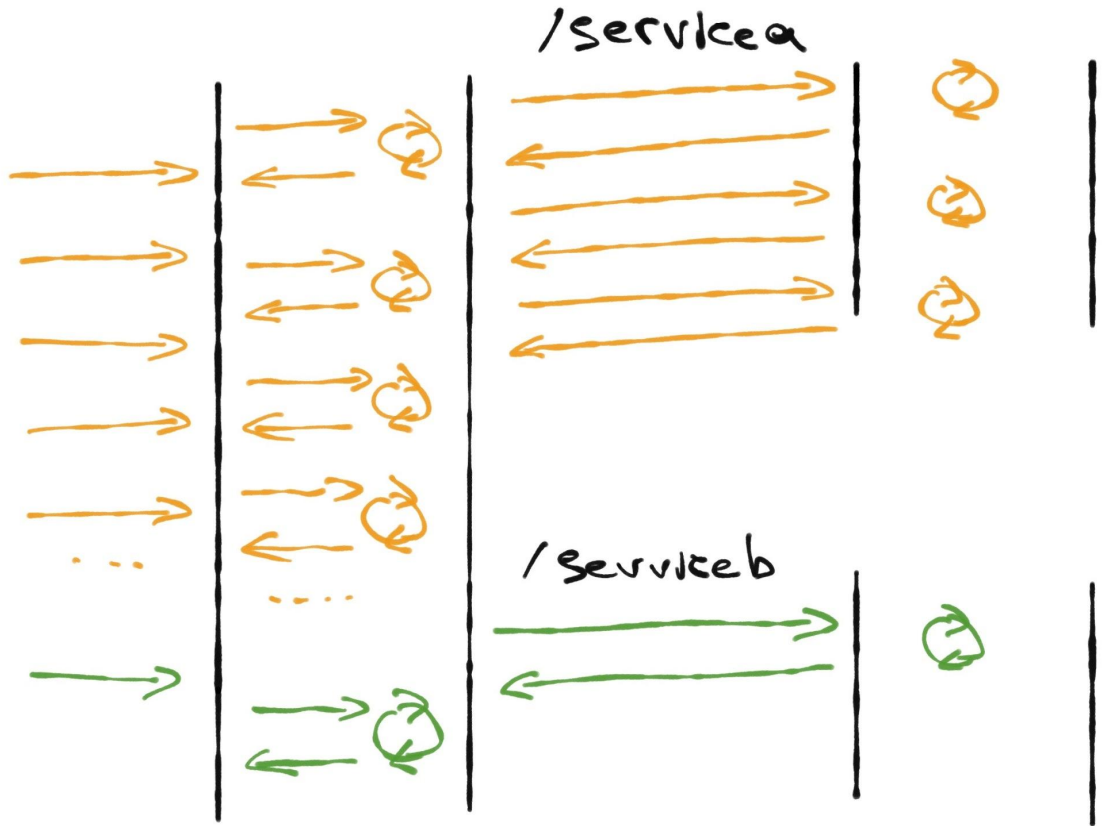




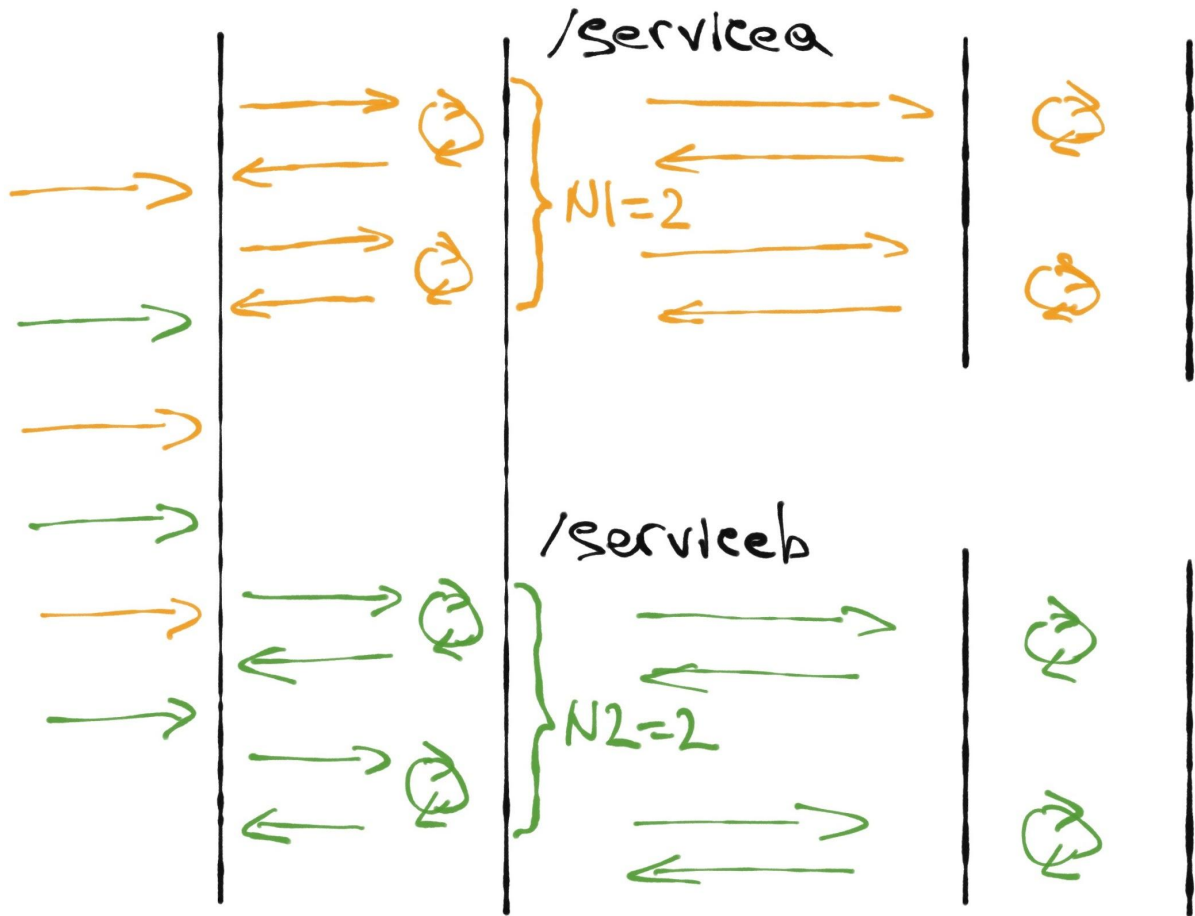
BULKHEAD

Brooklyn  
Zentrum

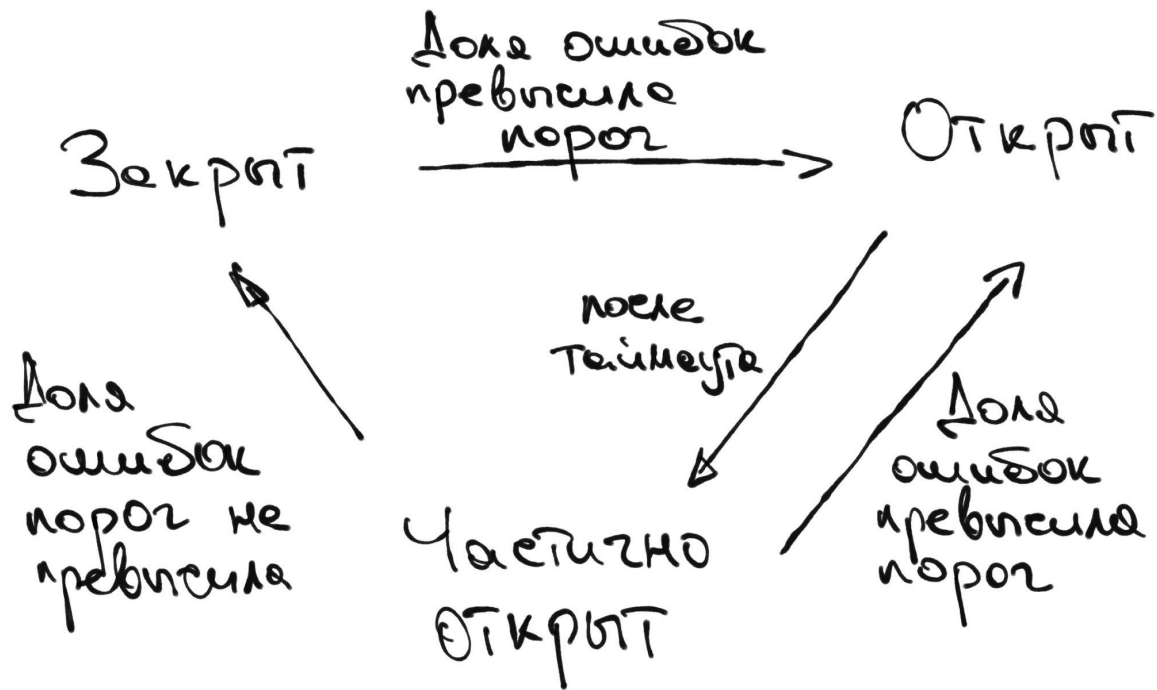






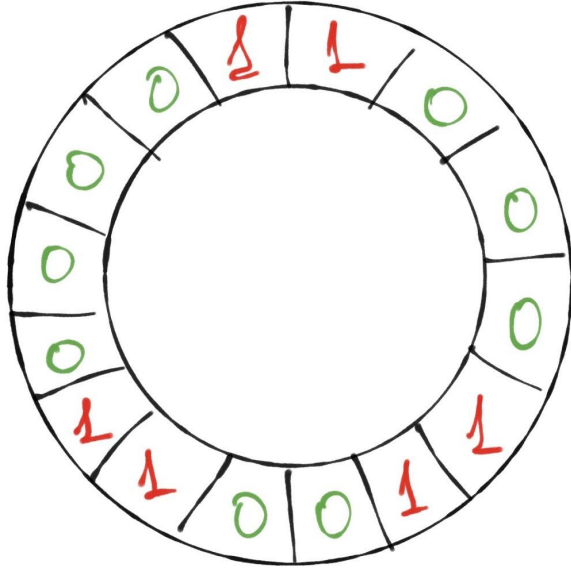


CIRCUITBREAKER



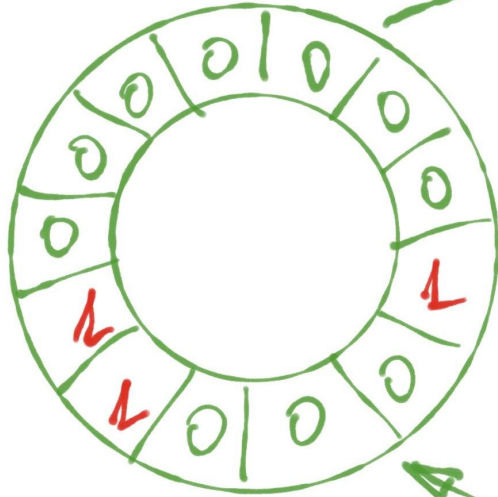
Threshold = 50%

Fract =  $\frac{6}{15}$



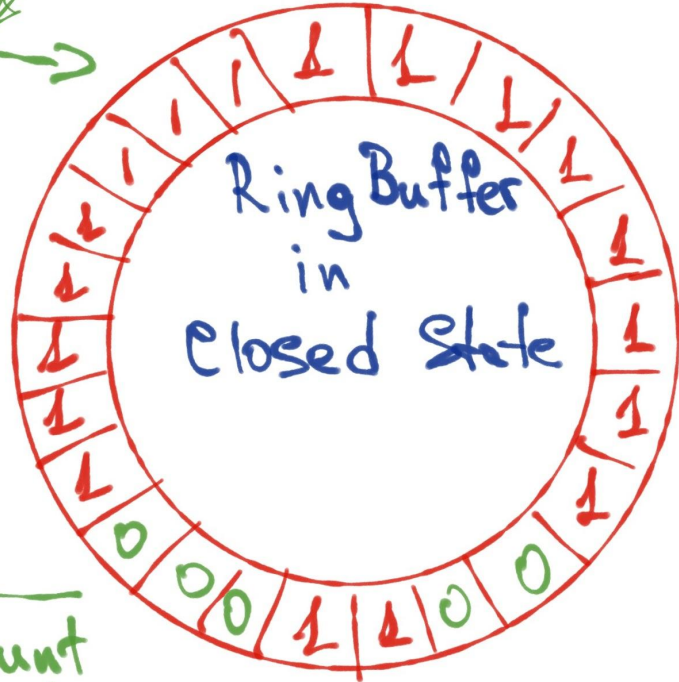
$\frac{6}{15} < \frac{1}{2} \Rightarrow \text{Closed}$

Ring Buffer  
in Half Open State



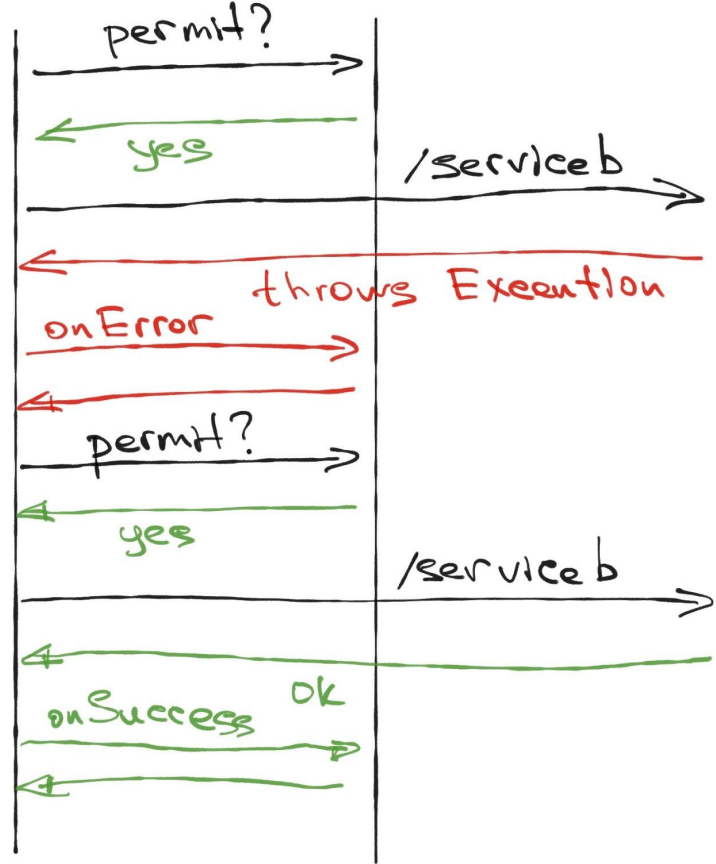
Transit to Closed State

Ring Buffer  
in  
Closed State



Start count  
after wait time  
duration

Service A      Circuit Breaker      Service B



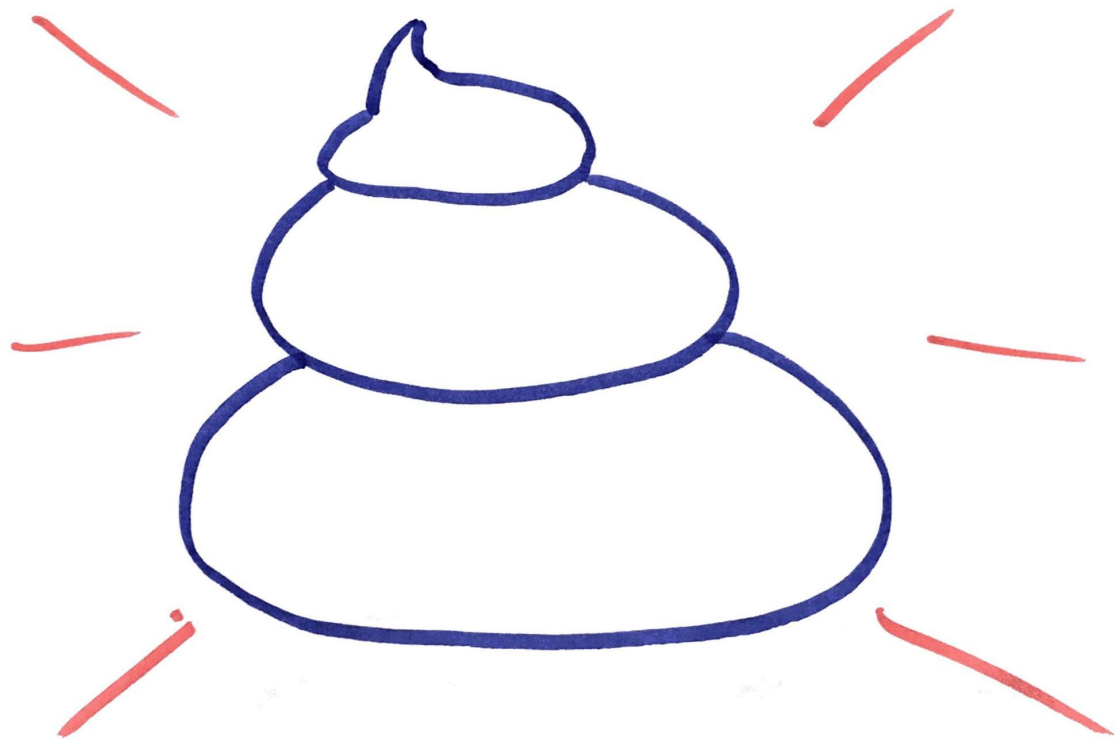
RESILIENCE 4J

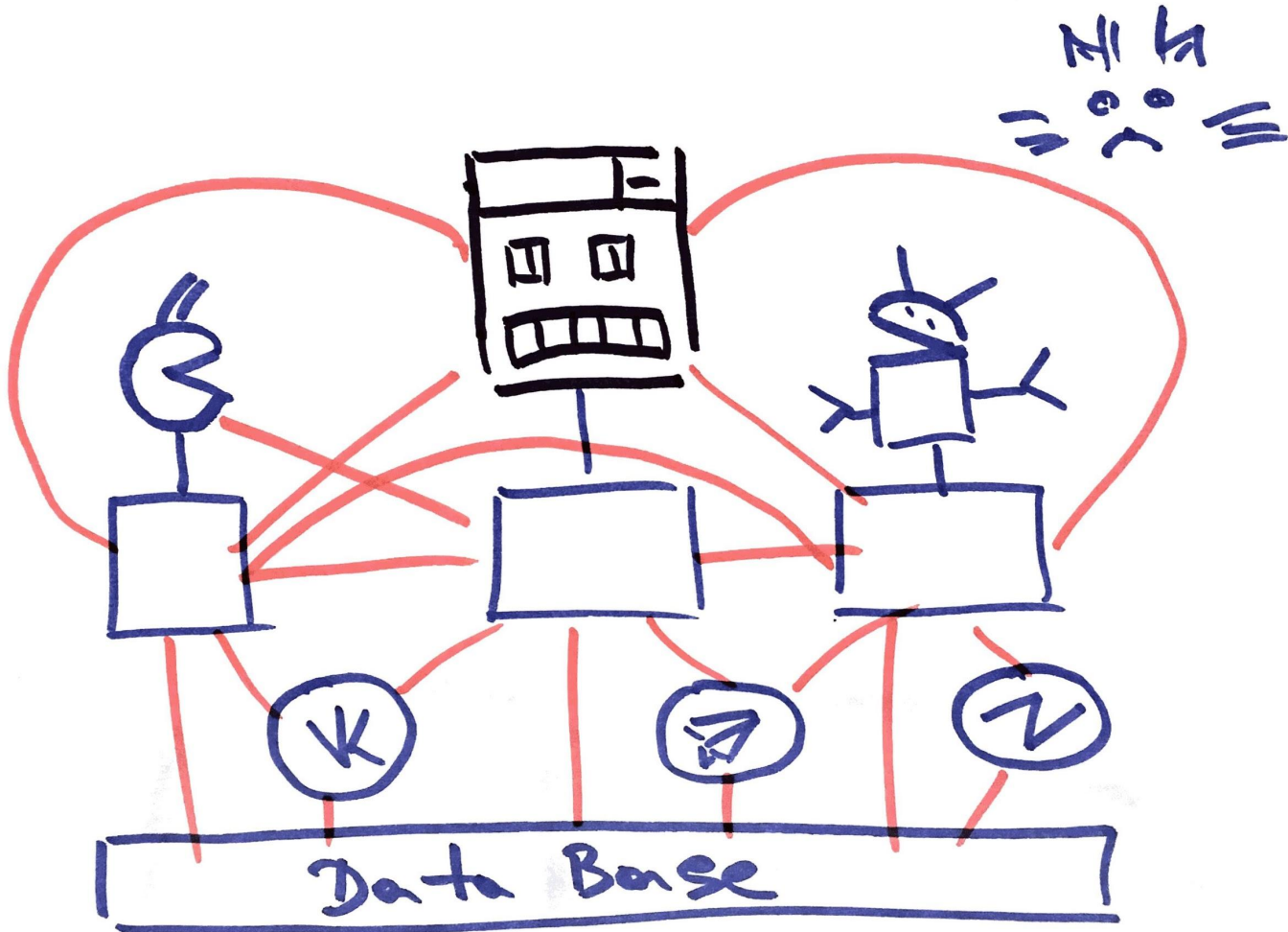
НАИВНОЕ ПРОЕКТИРОВАНИЕ...



ОСНОВНАЯ ГИПОТЕЗА

SHARED DB + RPC =

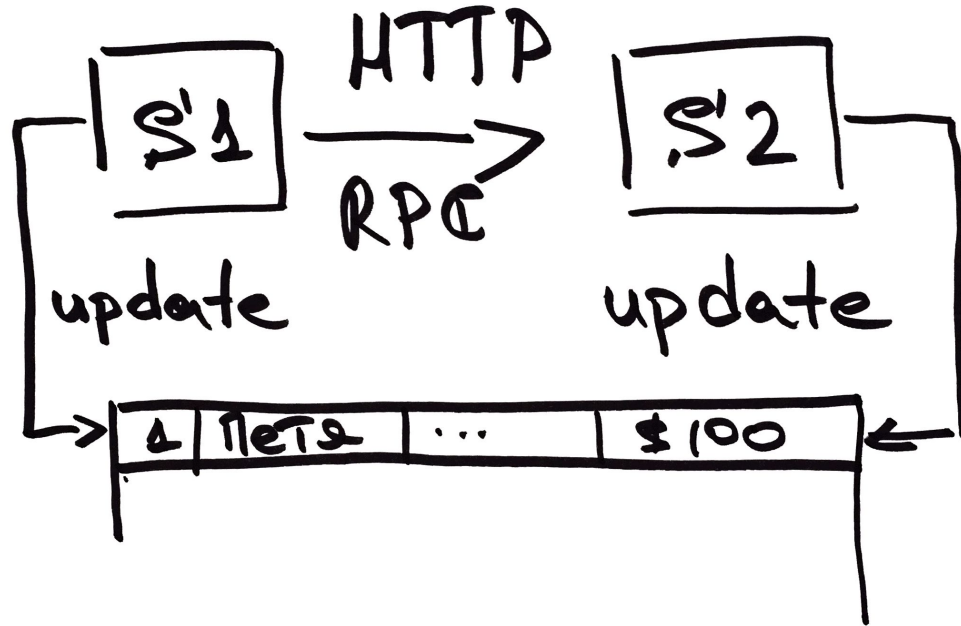




# ХЬЮСТОН, У НАС ПРОБЛЕМЫ?!

- БЛОКИРОВКИ И ДАЖЕ ... DEADLOCKS
- СВЯЗАННОСТЬ ВО ВРЕМЯ РАЗРАБОТКИ
- НЕТ ВЫБОРА В ОРГАНИЗАЦИИ ДАННЫХ
- ОГРАНИЧЕННЫЕ ВАРИАНТЫ ВЗАИМОДЕЙСТВИЯ (REQUEST/REPLY)

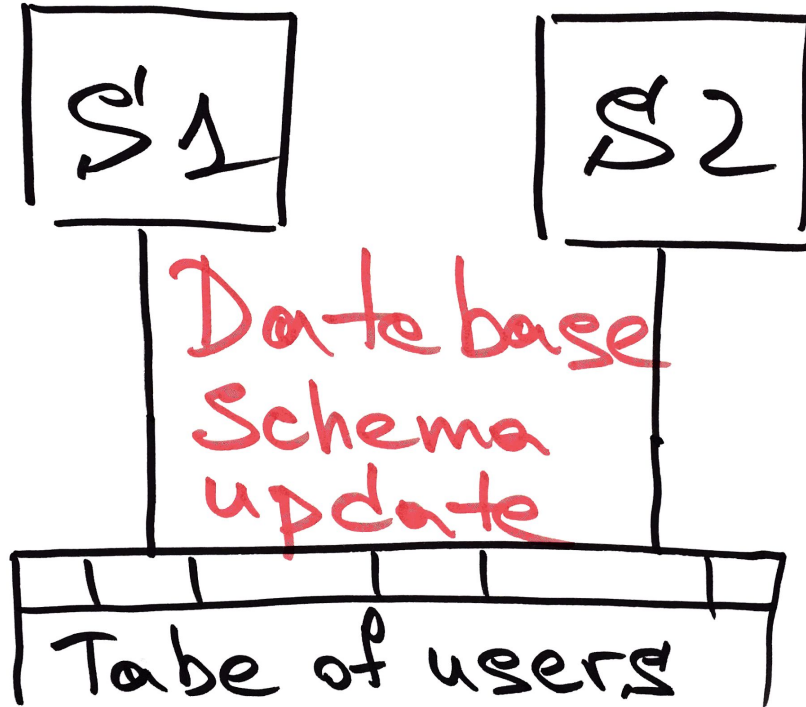
# БЛОКИРОВКИ И ДЕДЛОКИ



# ХЬЮСТОН, У НАС ПРОБЛЕМЫ?!

- БЛОКИРОВКИ И ДАЖЕ ... DEADLOCKS
- СВЯЗАННОСТЬ ВО ВРЕМЯ РАЗРАБОТКИ
- НЕТ ВЫБОРА В ОРГАНИЗАЦИИ ДАННЫХ
- ОГРАНИЧЕННЫЕ ВАРИАНТЫ ВЗАИМОДЕЙСТВИЯ (REQUEST/REPLY)

# КОНФЛИКТЫ ОБНОВЛЕНИЯ СХЕМЫ ДАННЫХ

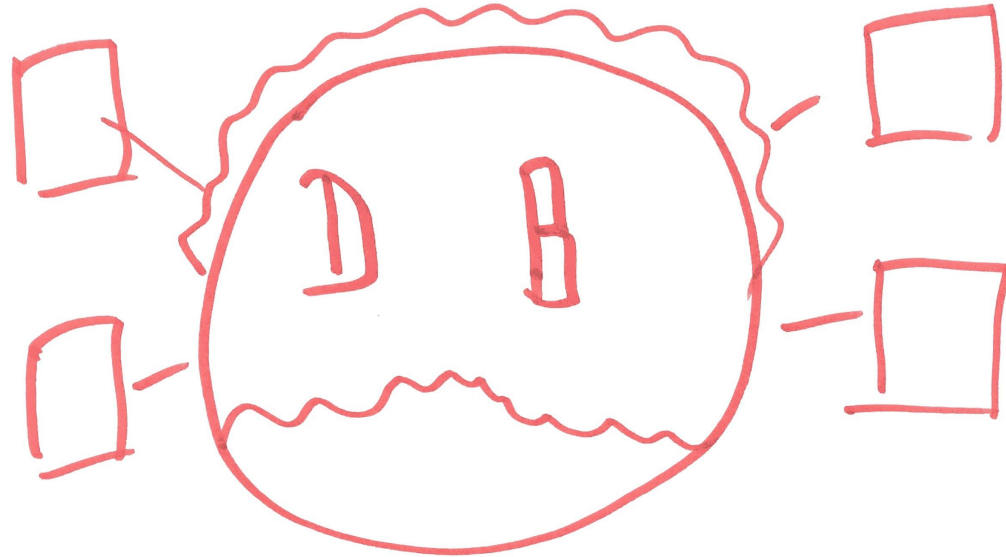




# ХЬЮСТОН, У НАС ПРОБЛЕМЫ?!

- БЛОКИРОВКИ И ДАЖЕ ... DEADLOCKS
- СВЯЗАННОСТЬ ВО ВРЕМЯ РАЗРАБОТКИ
- **НЕТ ВЫБОРА В ОРГАНИЗАЦИИ ДАННЫХ**
- ОГРАНИЧЕННЫЕ ВАРИАНТЫ ВЗАИМОДЕЙСТВИЯ (REQUEST/REPLY)

ЕДИНАЯ БД НА ВСЕ



One DB To Rule Them All...

# ХЬЮСТОН, У НАС ПРОБЛЕМЫ?!

- БЛОКИРОВКИ И ДАЖЕ ... DEADLOCKS
- СВЯЗАННОСТЬ ВО ВРЕМЯ РАЗРАБОТКИ
- НЕТ ВЫБОРА В ОРГАНИЗАЦИИ ДАННЫХ
- ОГРАНИЧЕННЫЕ ВАРИАНТЫ ВЗАИМОДЕЙСТВИЯ (REQUEST/REPLY)

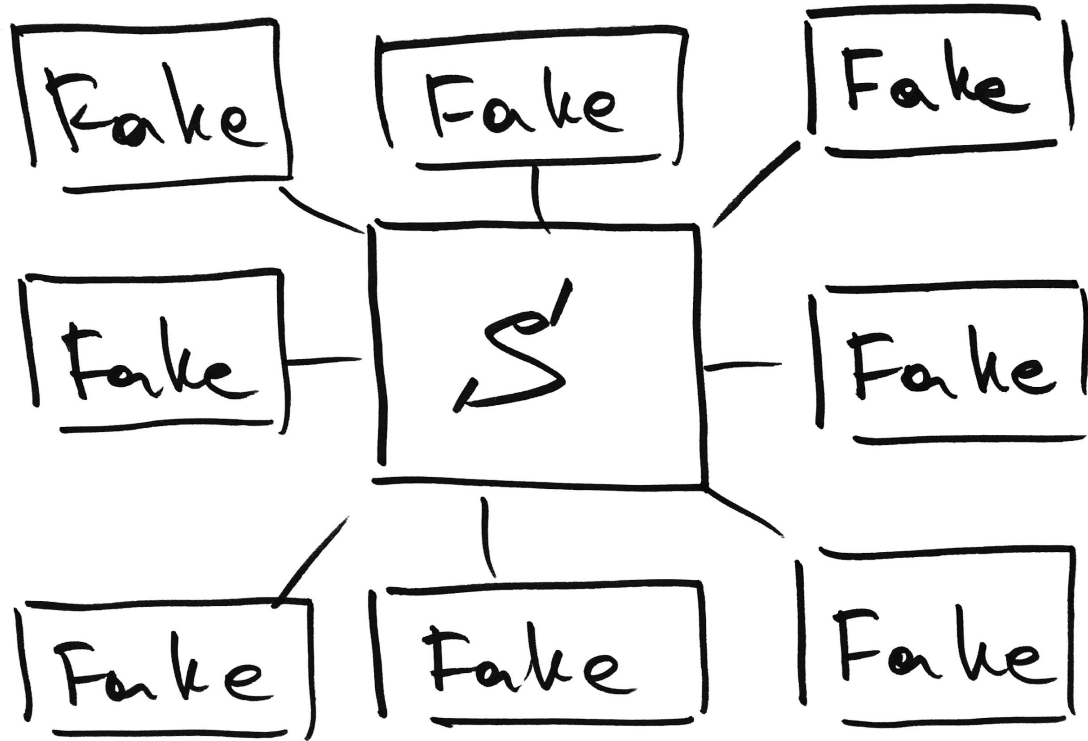
# СЛОЖНЫЕ СЕРВИСЫ

✧ Фильтрация

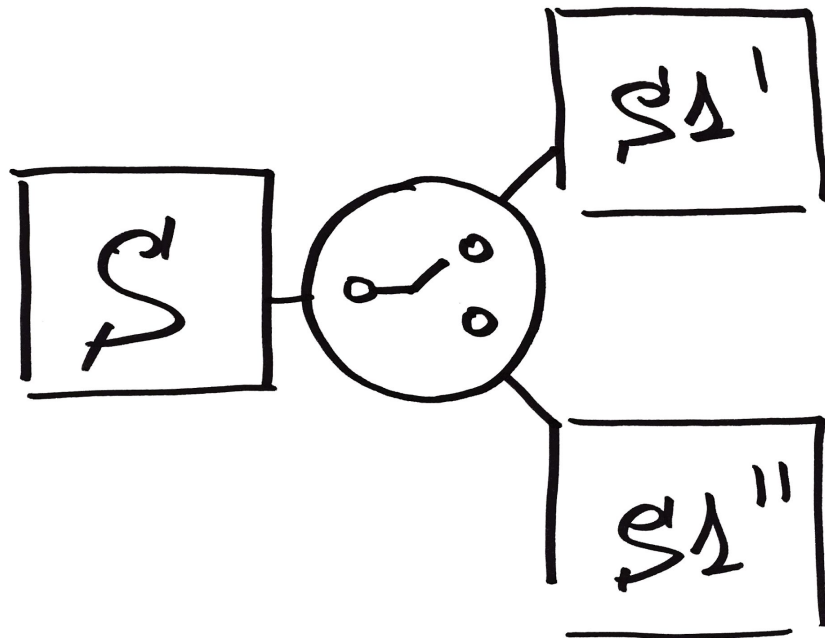
✧ Агрегирование

✧ Упорядочение

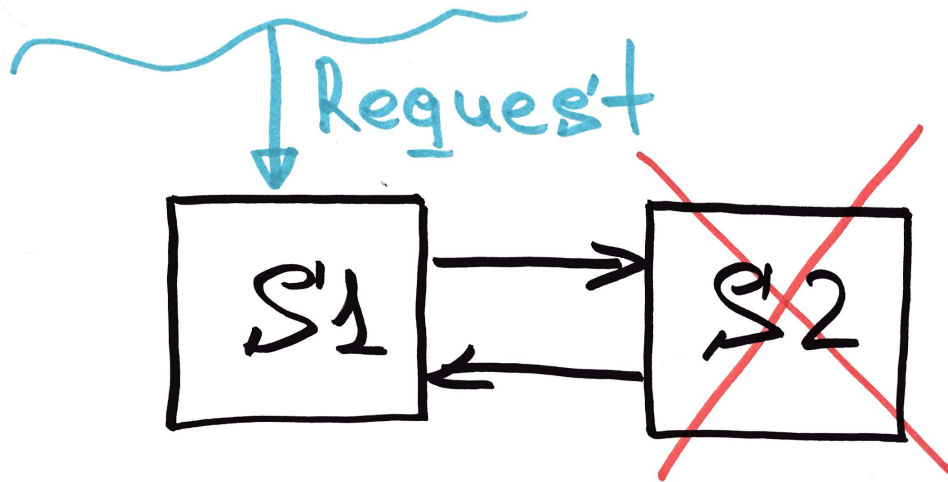
# ПЛОХО ПОДДАЮЩИЕСЯ ТЕСТИРОВАНИЮ



... И МАСШТАБИРОВАНИЮ



... И НЕНАДЕЖНЫЕ

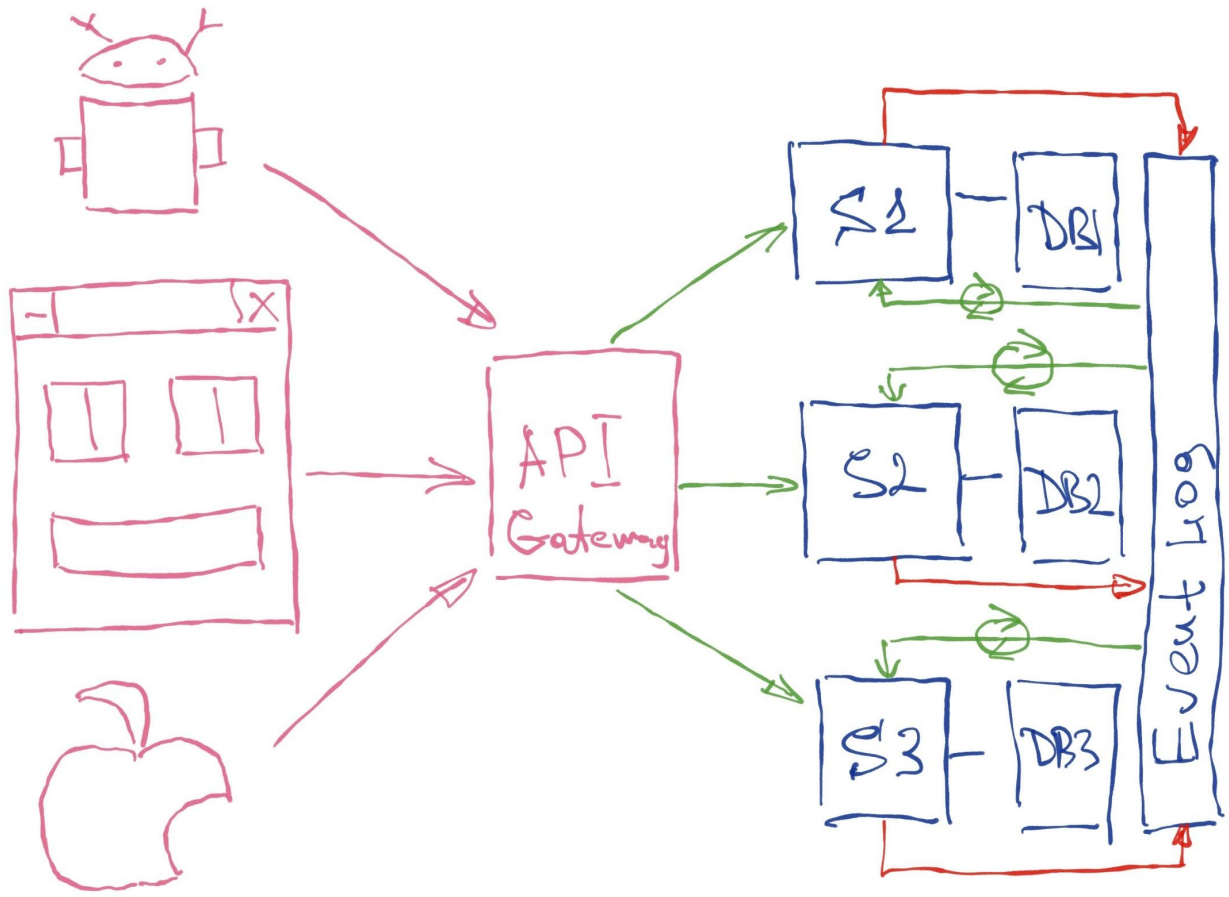


ЧТО ДЕЛАТЬ?



НЕ УСЛОЖНЯТЬ!

И ПЕРЕЕХАТЬ НА МИКРОСЕРВИСЫ ;)...



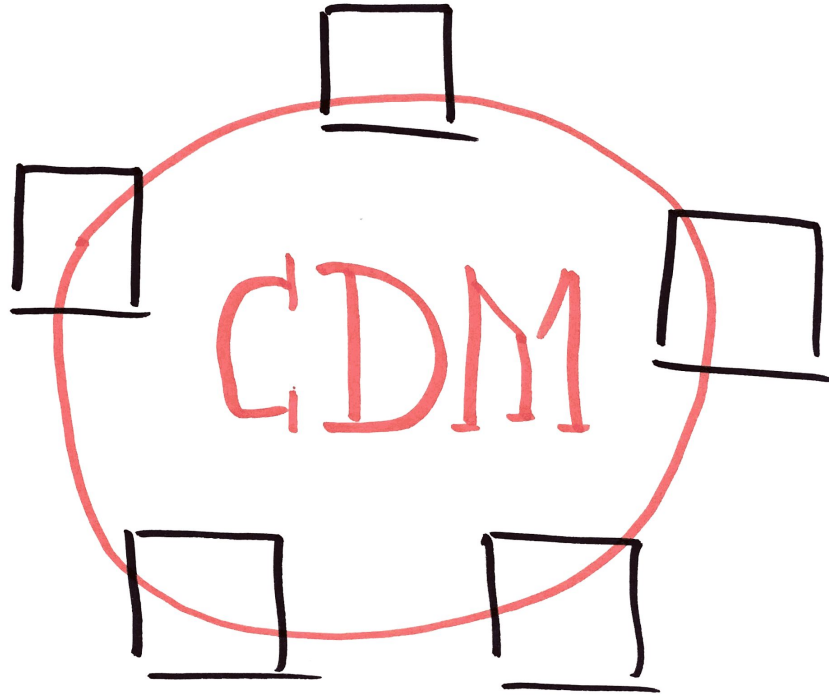
# РЕЦЕПТ

"ЭТО ЗАВИСИТ ОТ УДАЧИ. ВСЕ ЗАВИСИТ ОТ УДАЧИ..."

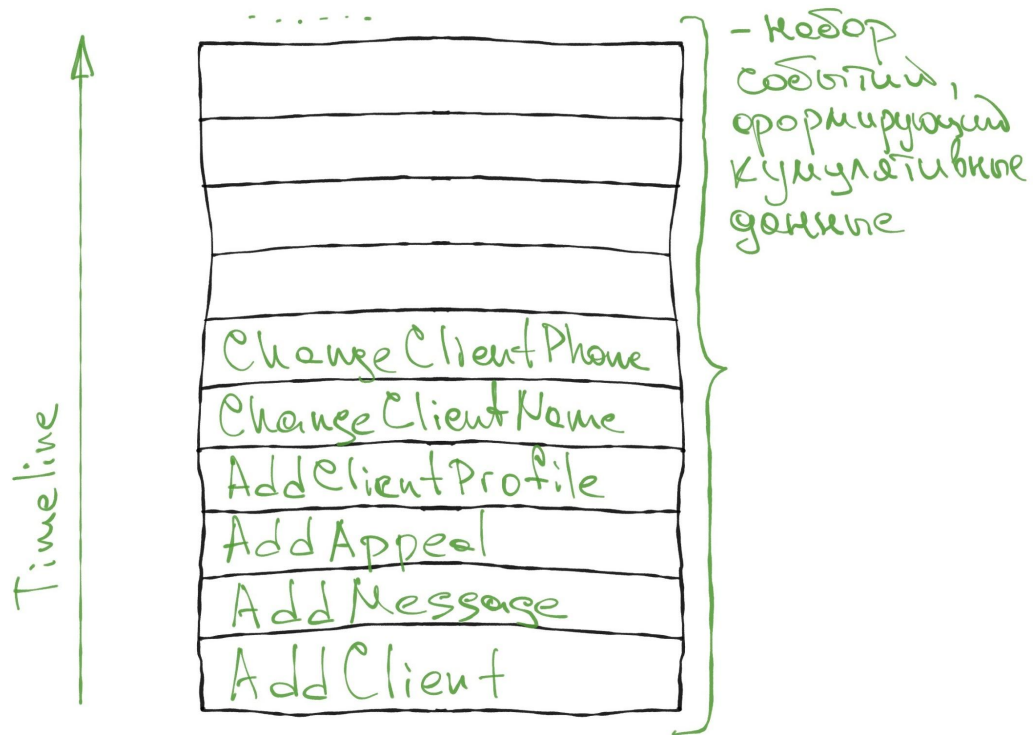
# ИНГРЕДИЕНТЫ...

- КАНОНИЧЕСКАЯ МОДЕЛЬ?!
- EVENTSOURCING?!
- CQRS?!
- ТЕСТЫ?!
- МОНИТОРИНГ, ТРАСИРОВЩИКИ, ЕТС..?!

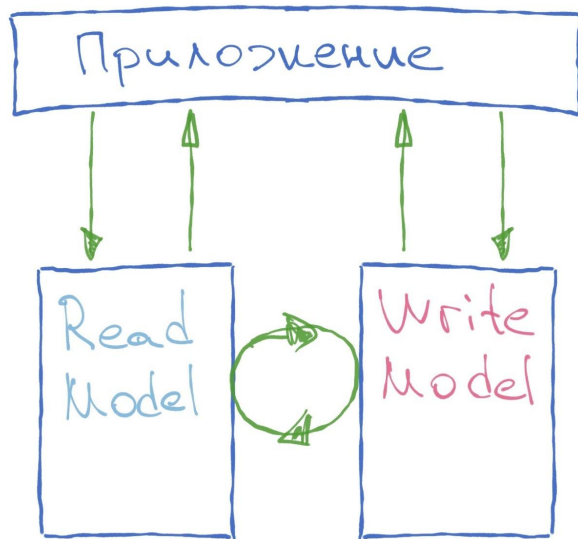
# КАНОНИЧЕСКАЯ МОДЕЛЬ ДАННЫХ

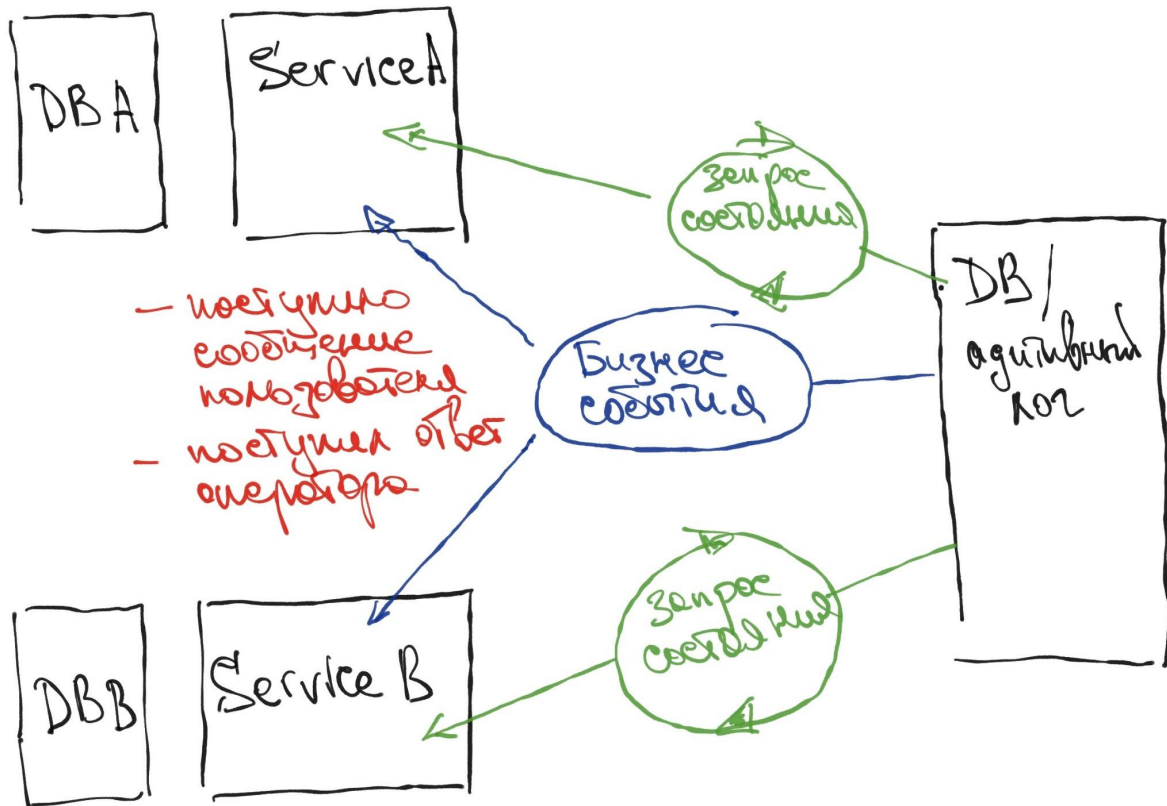


# EVENTSOURCING

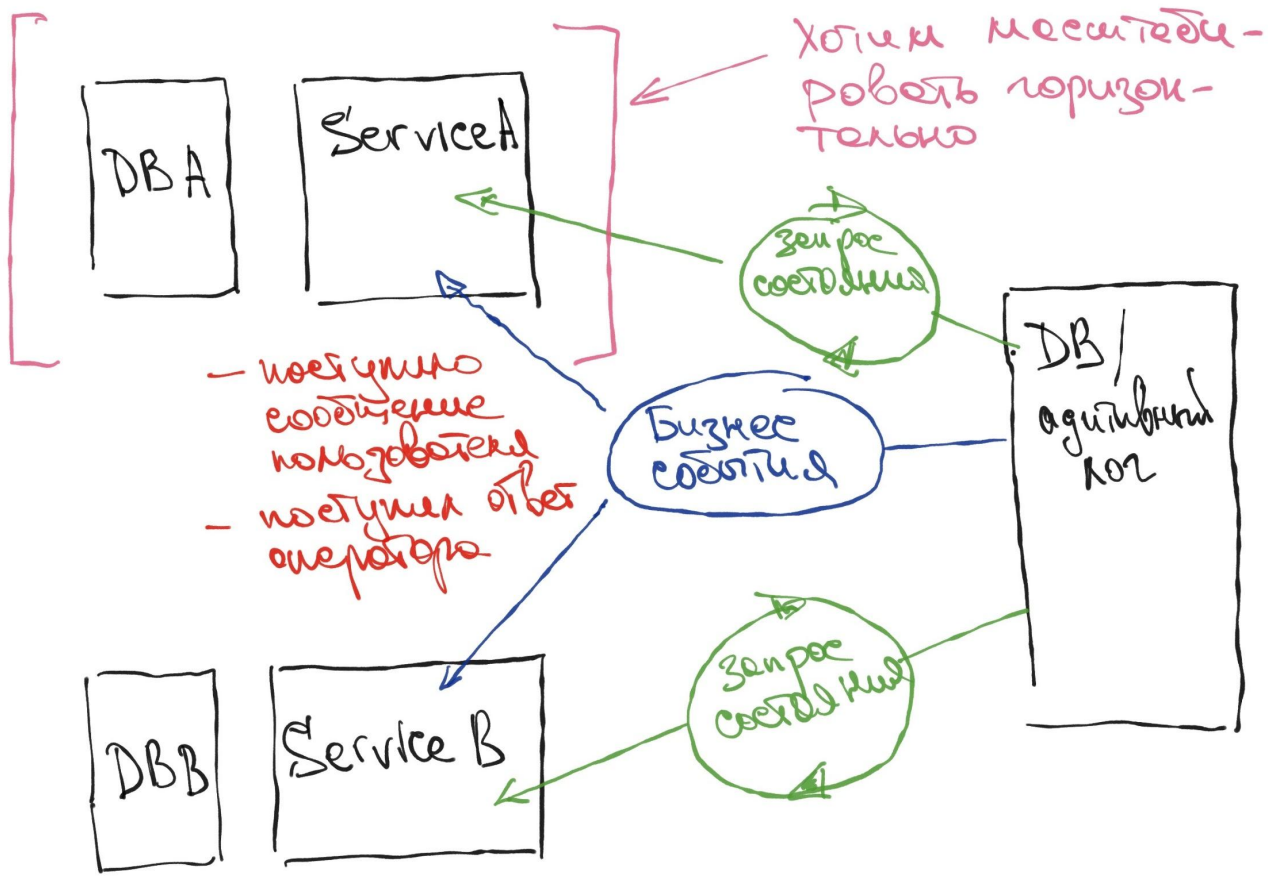


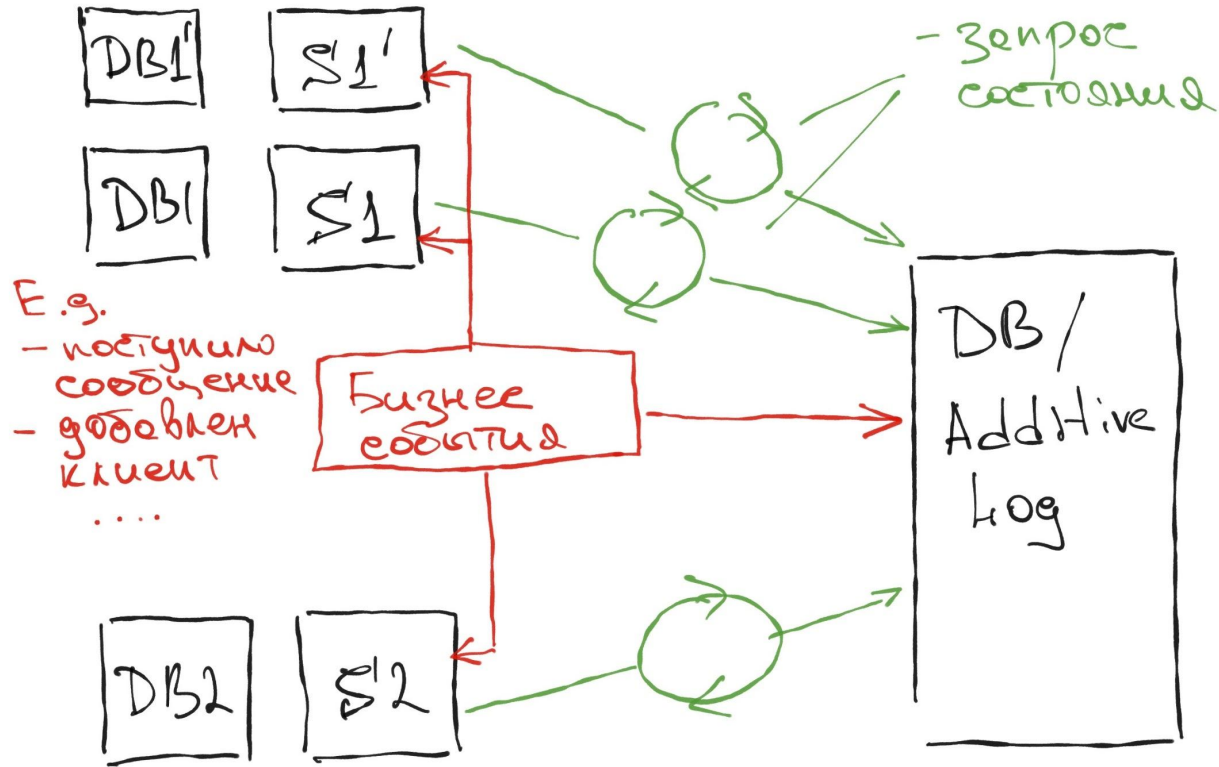
# CQRS











# ТЕСТИРОВАНИЕ

- UNIT TESTS
- INTEGRATION TESTS
- TESTCONTAINERS.ORG

# СПЕЦИАЛИЗИРОВАННЫЙ ИНСТРУМЕНТАРИЙ...

- PG\_STATS\_STATEMENTS
- AUTO\_EXPLAIN
- POSTGRES.AI

# ИТОГО

- УДОБНЫЕ ДЛЯ ВАС РЕШЕНИЯ - ОБЫЧНО ПРАВИЛЬНЫЕ
- DATA FIRST
- СЛЕДИТЕ ЗА ПУЛАМИ СОЕДИНЕНИЙ И ПОТОКОВ
- ТЕСТИРУЙТЕ САМЫЕ ХУДШИЕ СЦЕНАРИИ
- НЕ ПРЕНЕБРЕГАЙТЕ СПЕЦИАЛИЗИРОВАННЫМИ ИНСТРУМЕНТАМИ

# REFERENCES

- GREGOR HOHPE. ENTERPRISE INTEGRATION PATTERNS
- CHRIS RICHARDSON. MICROSERVICES PATTERNS
- SEBASTIAN DASCHNER. ARCHITECTING MODERN JAVA EE APPLICATIONS
- MARTIN KLEPPMANN. DESIGNING DATA-INTENSIVE APPLICATIONS
- НИКОЛАЙ ГОЛОВ. ЦЕЛОСТНОСТЬ ДАННЫХ В МИКРОСЕРВИСНОЙ АРХИТЕКТУРЕ
- МАКСИМ ГОРЕЛИКОВ. ДИЗАЙН РЕАКТИВНОЙ СИСТЕМЫ НА SPRING 5

КРИТИКА ПРИВЕТСТВУЕТСЯ, НО НЕ ЯВЛЯЕТСЯ ОБЯЗАТЕЛЬНОЙ ;)

