СБЕРБАНК ТЕХНОЛОГИИ

От монолита к микросервисам

На примере разработки «Кредитной фабрики» Сбербанка

Николай Арчаков
Nikolay.Archakov@gmail.com

“Кредитная фабрика” это технология, реализующая сквозной процесс обработки и принятия решений по кредитным заявкам, обеспечивающая автоматизацию существующих ручных процессов обработки данных, начиная от ввода кредитной заявки во фронт офисе, сбора данных из внешних и внутренних систем и заканчивая принятием решения о выдаче или отказе в кредите.

Основные объемные показатели

- **Максимальное количество кредитных заявок в день – 125 000**

Основные объемные показатели

- **Максимальное количество кредитных заявок в день – 125 000**
- **Пиковая нагрузка кредитных заявок в час – 25 000**

Основные объемные показатели

- **Максимальное количество кредитных заявок в день – 125 000**
- **Пиковая нагрузка кредитных заявок в час – 25 000**
- **Максимальное количество работающих пользователей – 26 000**

Основные объемные показатели

- **Максимальное количество кредитных заявок в день – 125 000**
- **Пиковая нагрузка кредитных заявок в час – 25 000**
- **Максимальное количество работающих пользователей – 26 000**
- **Среднее количество запросов к внешним системам в день – 2 000 000**

Основные объемные показатели

- **Максимальное количество кредитных заявок в день – 125 000**
- **Пиковая нагрузка кредитных заявок в час – 25 000**
- **Максимальное количество работающих пользователей – 26 000**
- **Среднее количество запросов к внешним системам в день – 2 000 000**
- **Режим работы автоматизированной системы – 24 х 7 х 365**

Основные объемные показатели

- **Максимальное количество кредитных заявок в день – 125 000**
- **Пиковая нагрузка кредитных заявок в час – 25 000**
- **Максимальное количество работающих пользователей – 26 000**
- **Среднее количество запросов к внешним системам в день – 2 000 000**
- **Режим работы автоматизированной системы – 24 х 7 х 365**
- **Надежность работы системы – 99,7 %**

Monolith *VS* MicroServices

«*Не следует начинать новый проект с микросервисов, даже при полной уверенности, что будущее приложение будет достаточно большим, чтобы оправдать такой подход.*»

Мартин Фаулер

«Не взирая на все имеющиеся в средах разработки удобные инструменты для рефакторинга и переноса кода, разделить проект на части крайне непростая задача. Еще труднее будет разделить существующий монолит на части.»

Стефан Тилков

No silver bullet

«Не существует единственного универсального решения, которое увеличит производительность разработчика и простоту проекта.»

Фредерик Брукс

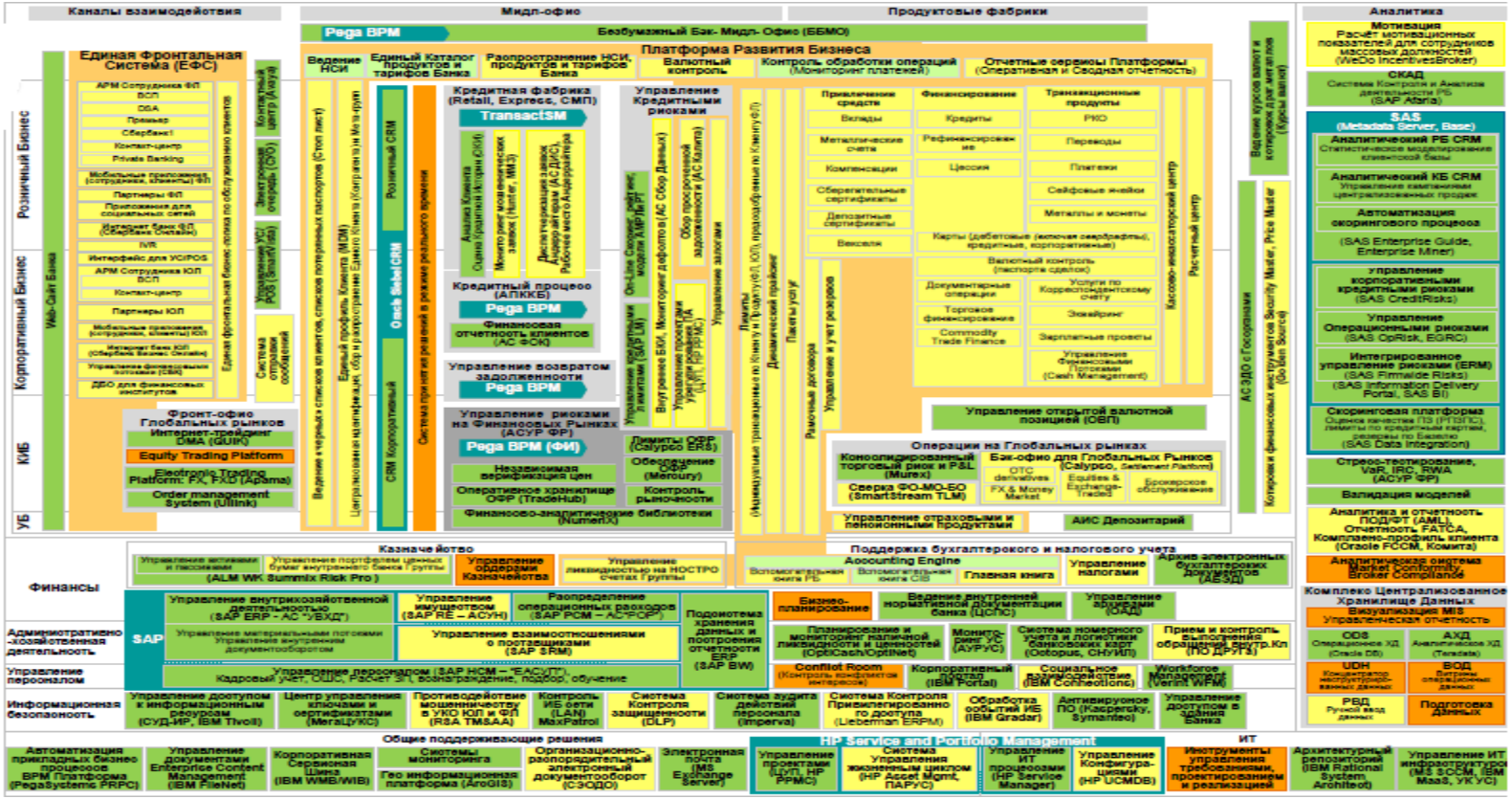
«Полагаю, что когда из инструментов имеется только молоток, то заманчиво рассматривать все как гвозди.»

Абрахам Маслоу

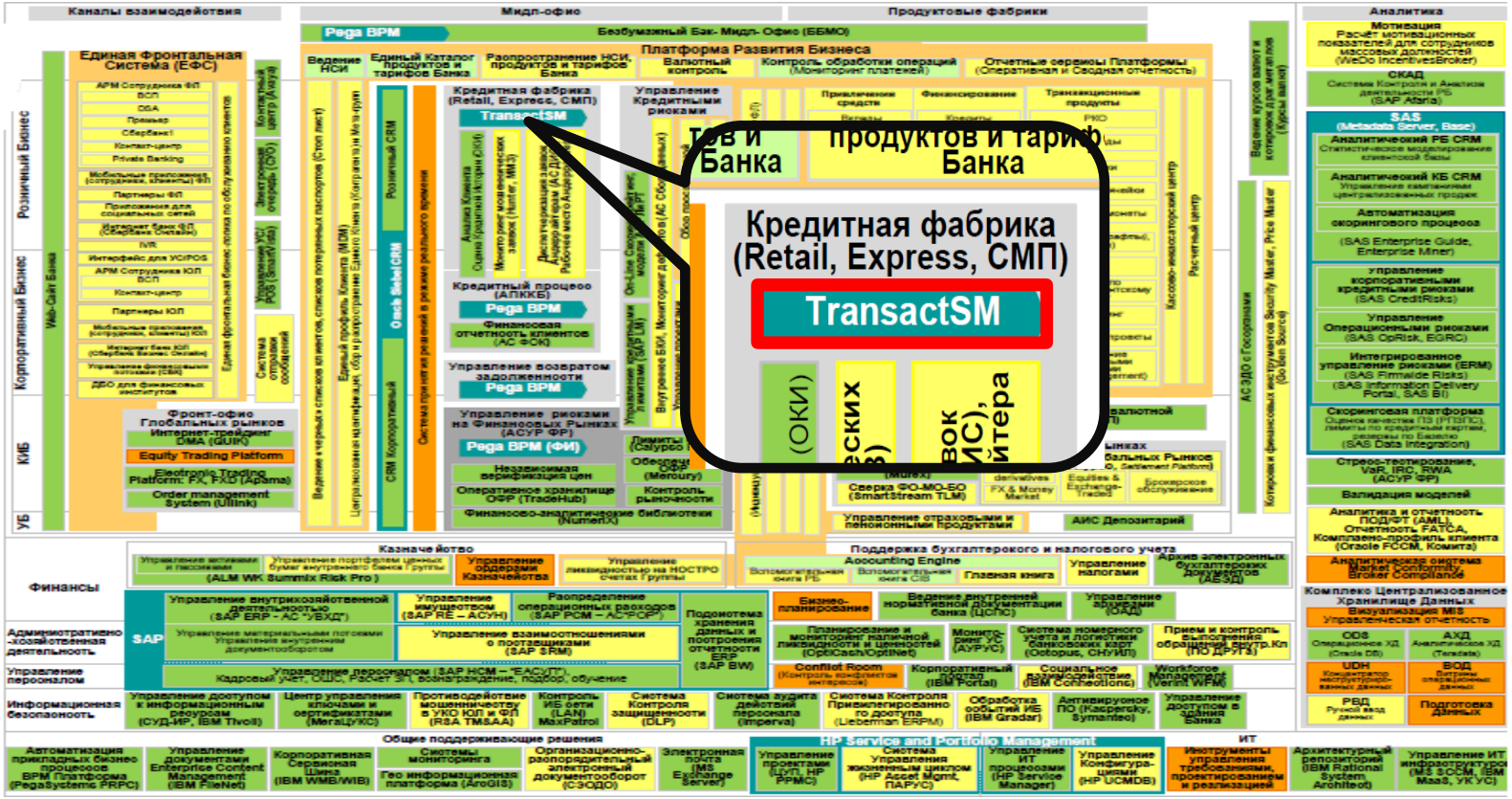
Исходные предпосылки

1. **Архитектурное видение**

Общая архитектура банка



Общая архитектура банка



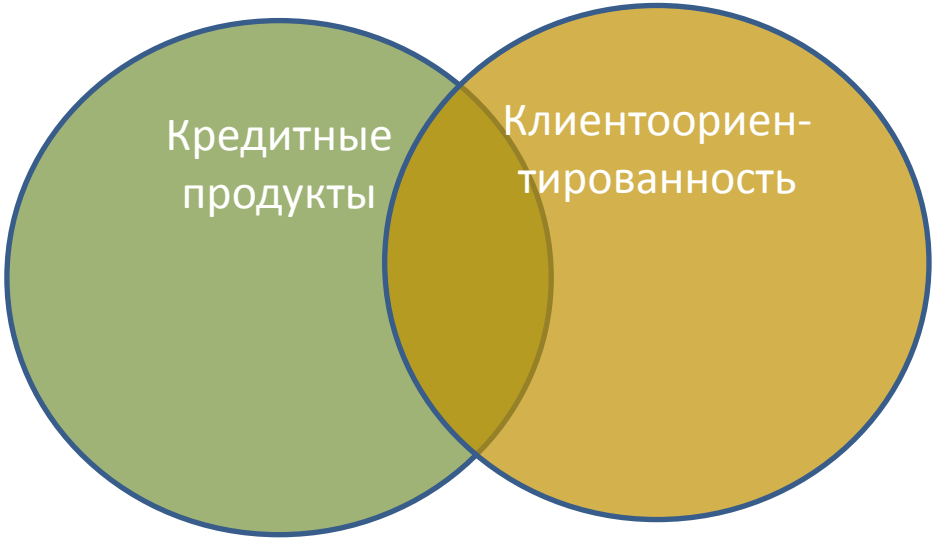
Исходные предпосылки

1. **Архитектурное видение**
2. **Предметная область**

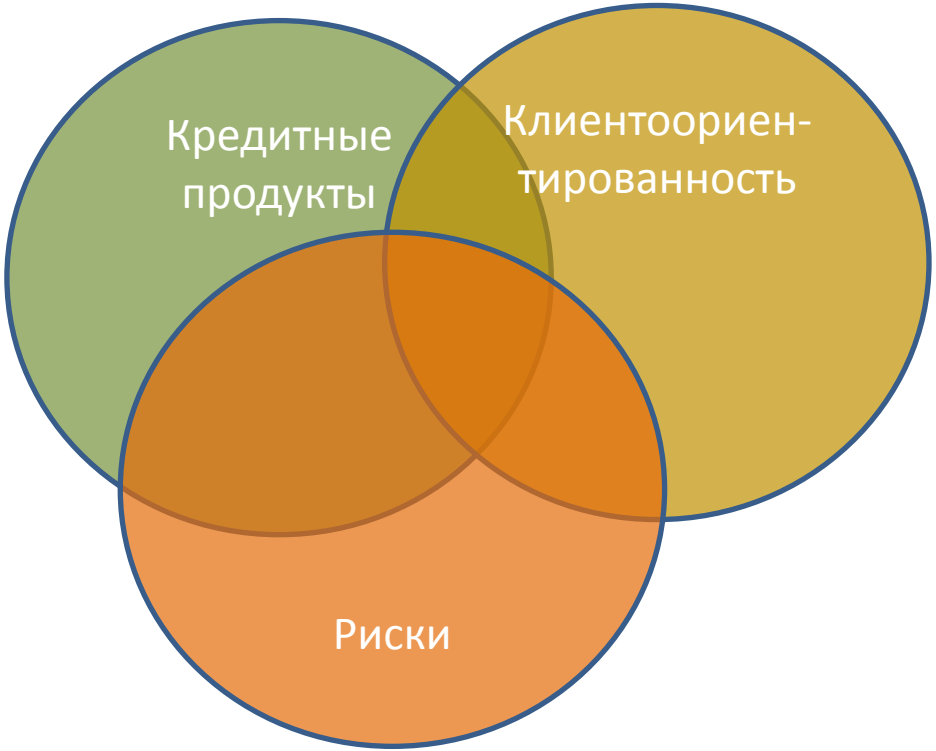
Предметная область

Предметная область

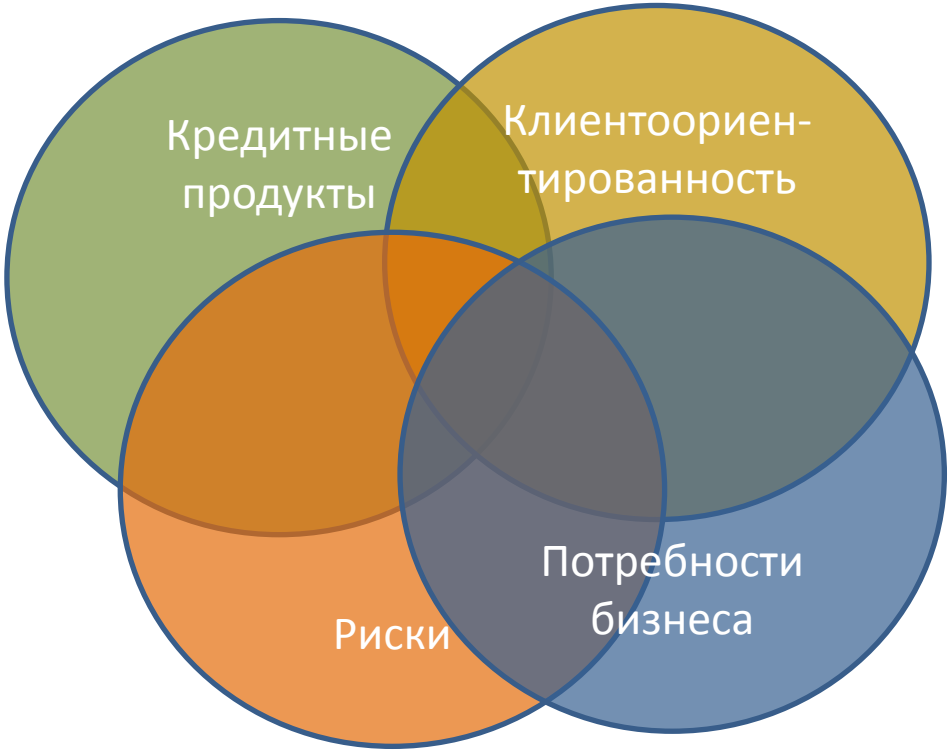
Предметная область



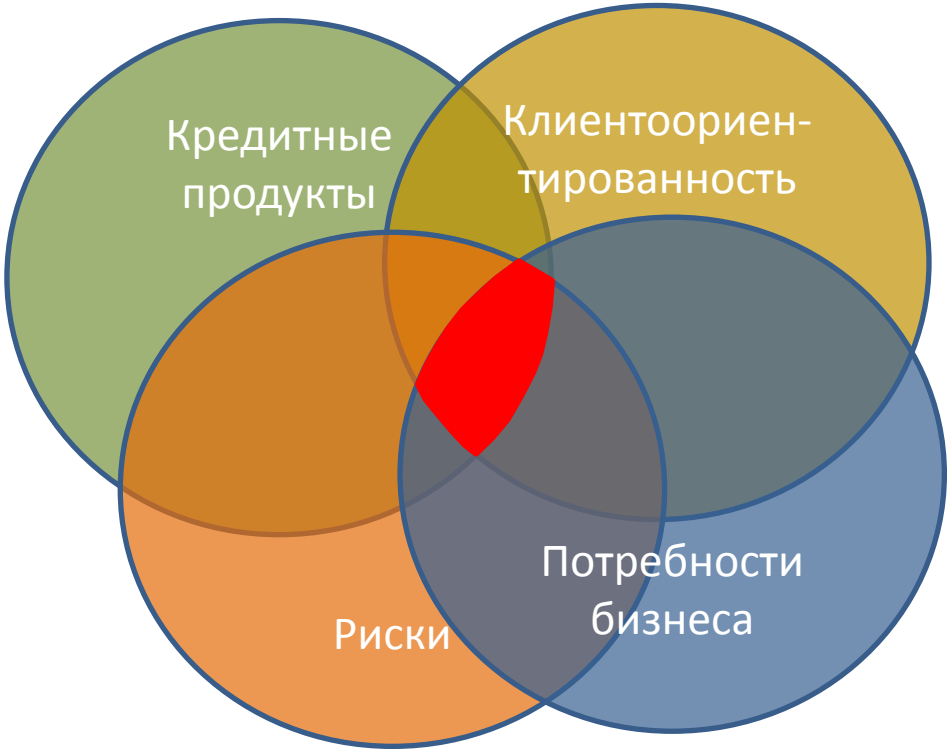
Предметная область



Предметная область



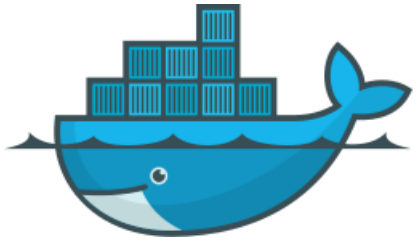
Предметная область



Исходные предпосылки

1. **Архитектурное видение**
2. **Предметная область**
3. **Инфраструктурные ограничения**

Инфраструктурная незрелость



docker

VS

Microsoft Azure

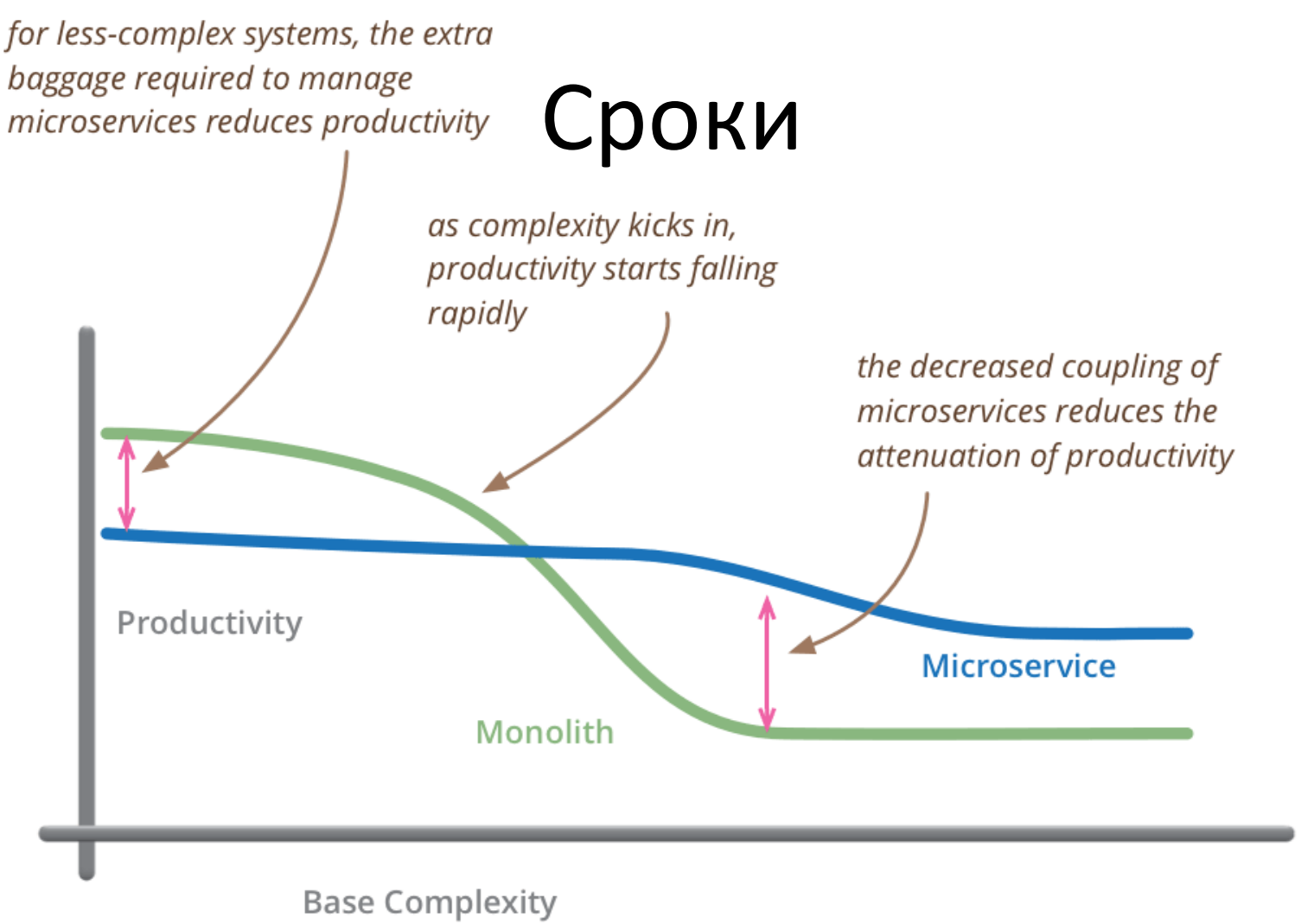
VS

CLOUDFOUNDRY

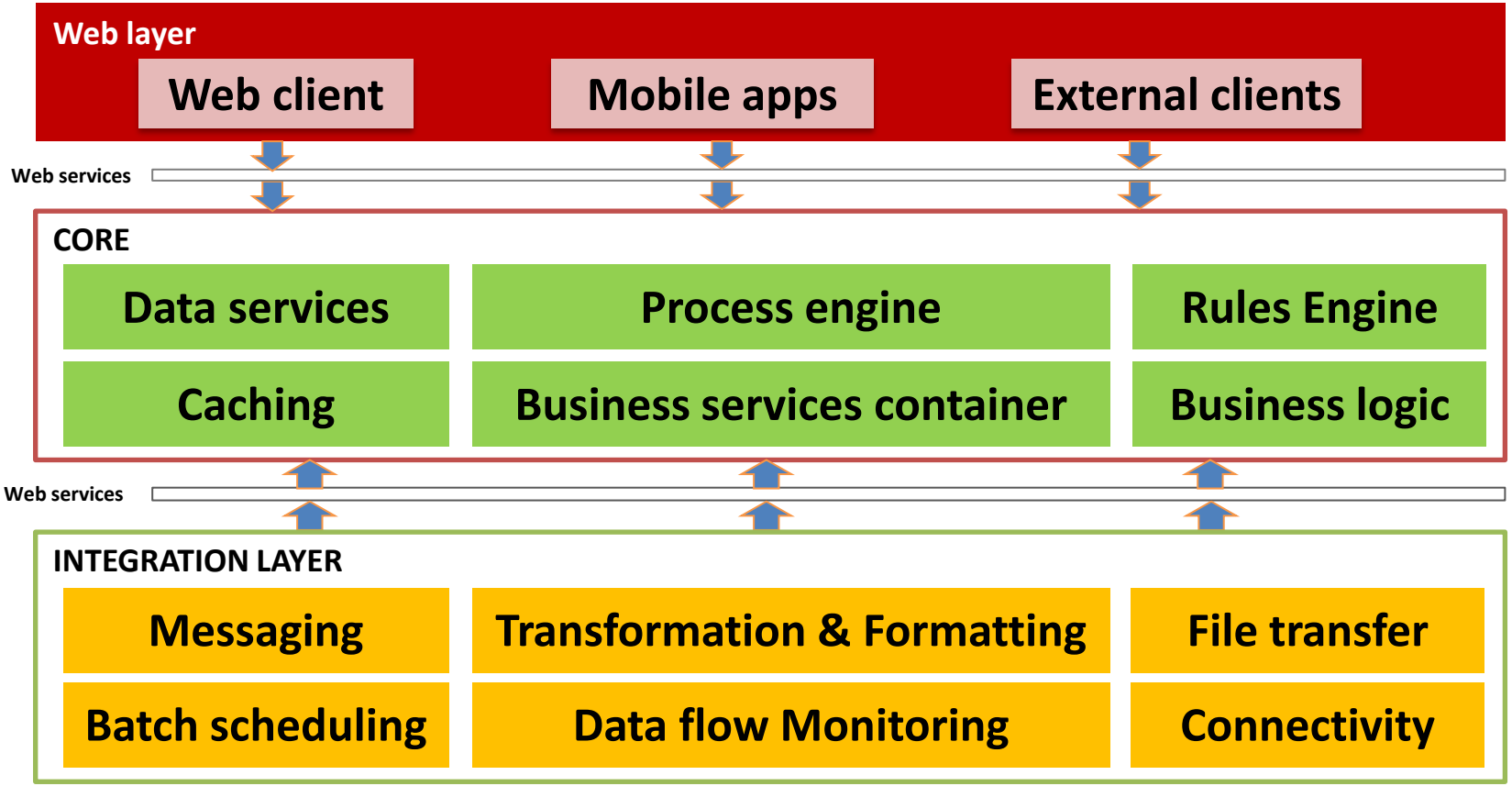
Исходные предпосылки

1. **Архитектурное видение**
2. **Предметная область**
3. **Инфраструктурные ограничения**
4. **Сроки**

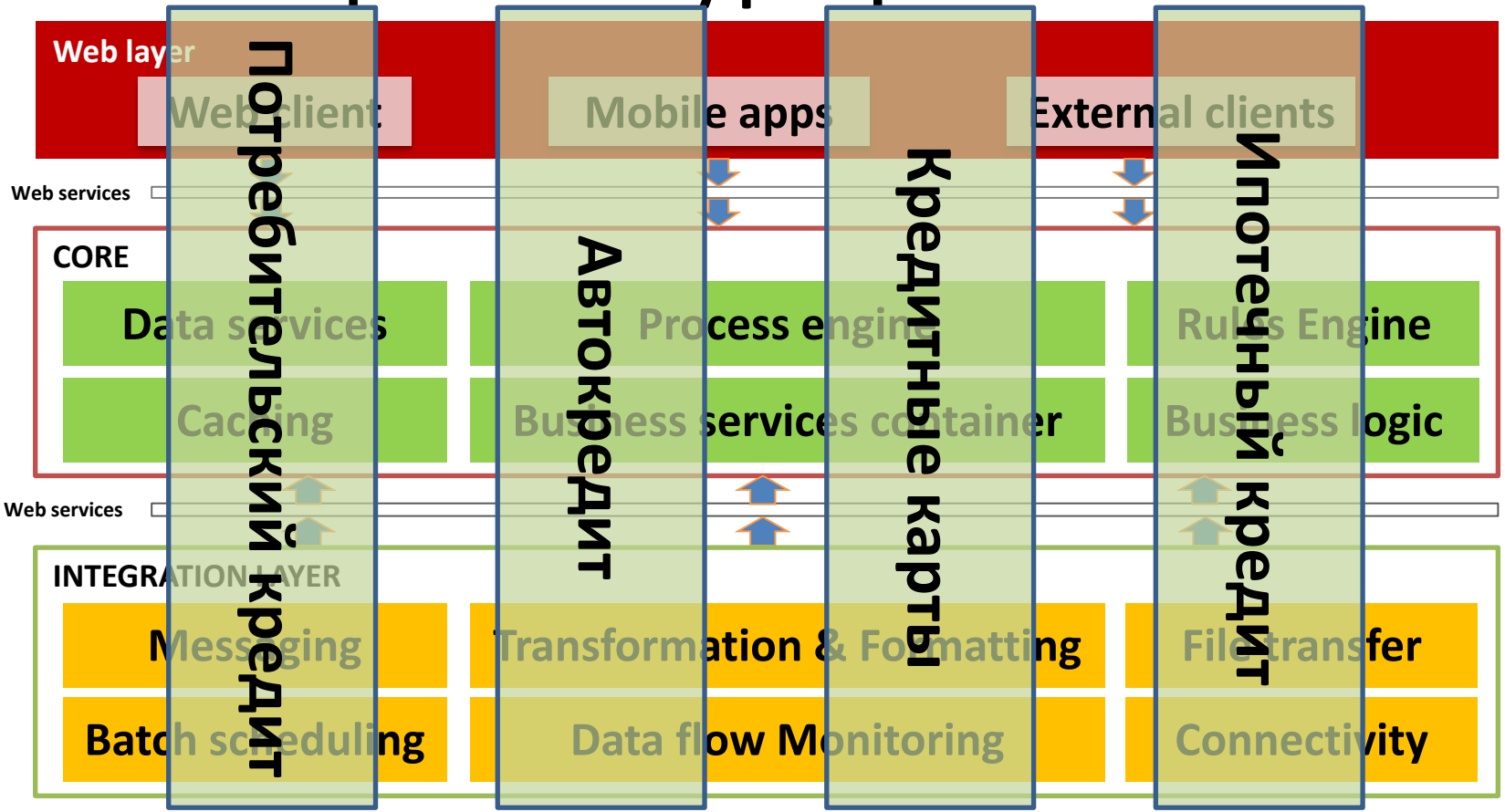
Сроки


for less-complex systems, the extra baggage required to manage microservices reduces productivity
as complexity kicks in, productivity starts falling rapidly
the decreased coupling of microservices reduces the attenuation of productivity
Productivity
Microservice
Monolith
Base Complexity

Архитектура решения



Архитектура решения



“Божественный” сервис

```
public class HelperService {
    private final static String STREET_ADDRESS_TEMPL_FULL = "улица %s, д.%s";
    private final static String STREET_ADDRESS_TEMPL_STREET = "улица %s";
    private final static String STREET_ADDRESS_TEMPL_HOME = "д. %s";

    public String makeAddressStreet(String street, String house) {
        if (street != null && house != null)
            return String.format(STREET_ADDRESS_TEMPL_FULL, street, house);

        if (street != null)
            return String.format(STREET_ADDRESS_TEMPL_STREET, street);

        if (house!=null)
            return String.format(STREET_ADDRESS_TEMPL_HOME, house);

        return EMPTY_STRING;
    }




}
```

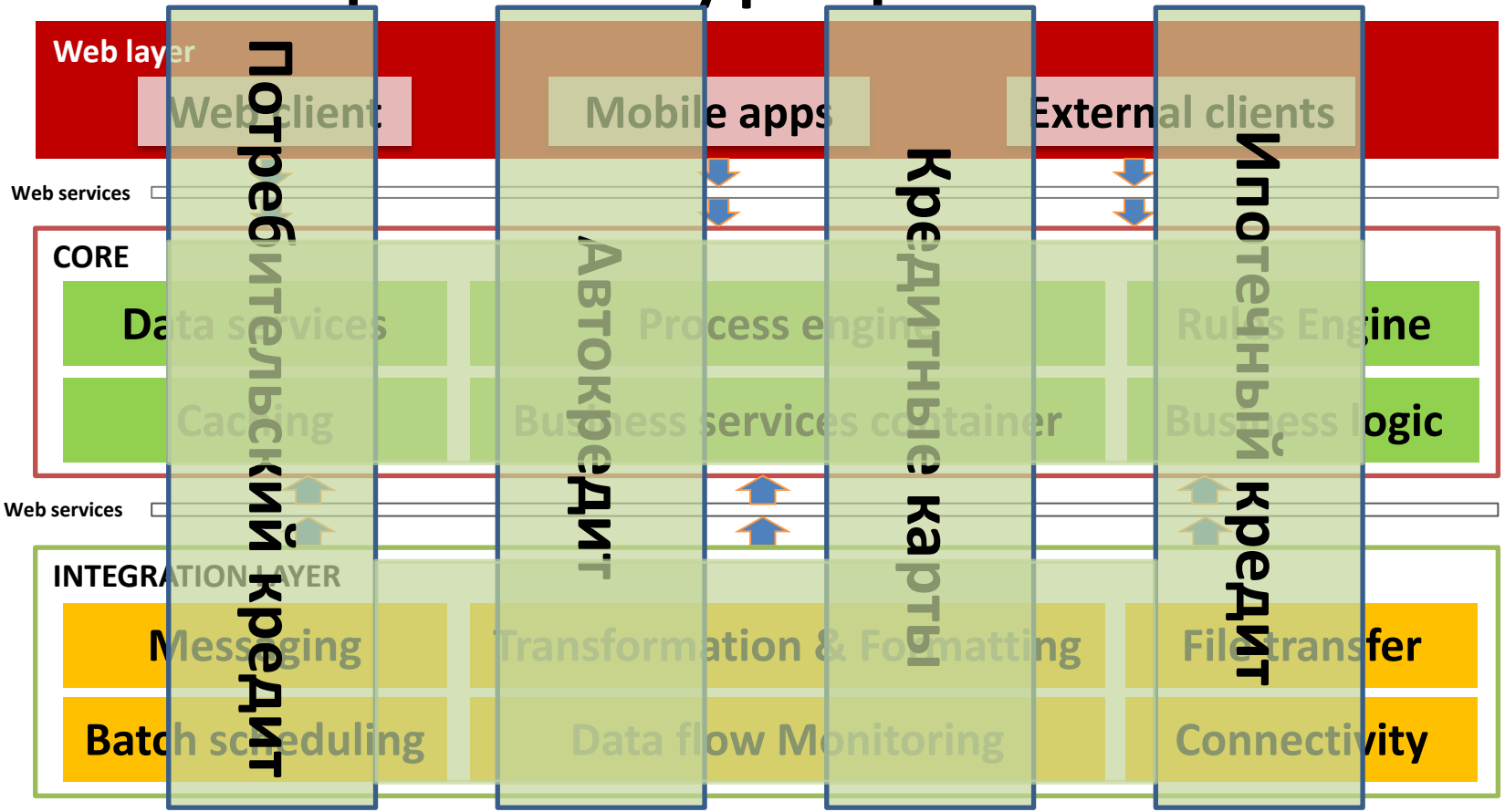
```
public class HelperService {
    . . .

    public String makeAddressStreet(String street, String house) {
        . . .
    }









}
```

```
public class HelperService {
    . . .

    public String makeAddressStreet(String street, String house) {
        . . .
    }

    // TODO Jira-886
    public Double debtBurden(List<Liability> liabilities) {
        double debt = 0.0;

        for (Liability liability : liabilities) {
            debt += liability.getDebt().doubleValue();
        }

        return new Double(debt);
    }




}
```

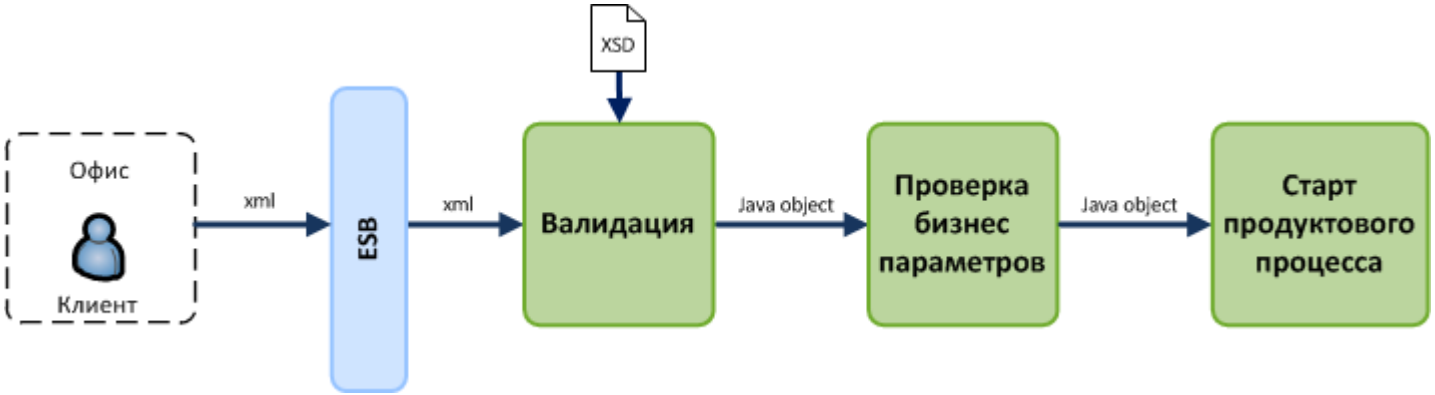
```
public class HelperService {
    . . .

    public String makeAddressStreet(String street, String house) {
        . . .
    }

    // TODO Jira-886
    public Double debtBurden(List<Liability> liabilities) {
        . . .
    }









}
```

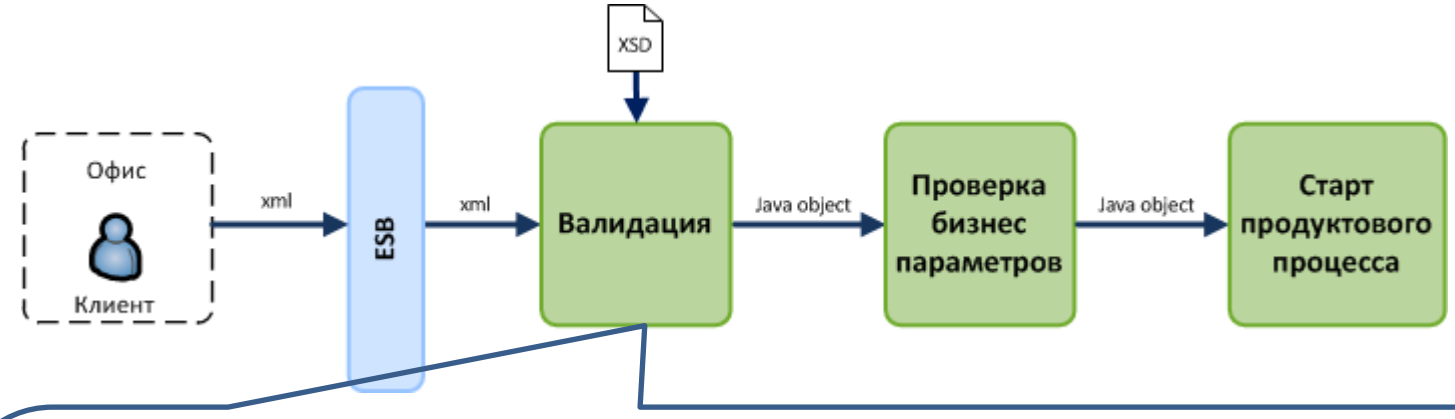
```
public class HelperService {
    private final static Mdm5 MDM5 = new Mdm5();

    public String makeAddressStreet(String street, String house) {
        . . .
    }

    // TODO Jira-886
    public Double debtBurden(List<Liability> liabilities) {
        . . .
    }

    // TODO Jira-917
    public String calculatedMD5(List<TrsCharac> inputs) throws Exception {
        String inputData = "";

        for (TrsCharac input: inputs) {//TODO
            inputData = inputData + input.getValue().replaceFirst("^0+","") + "_";
        }
        return MDM5.getMD5(inputData);
    }
}
```

Архитектура решения

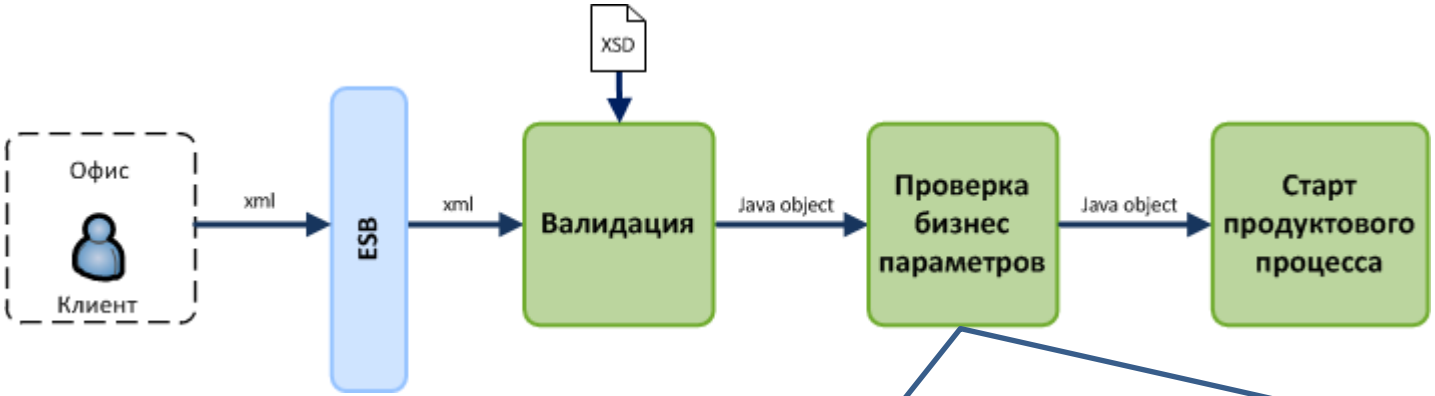


Страх дублирования кода

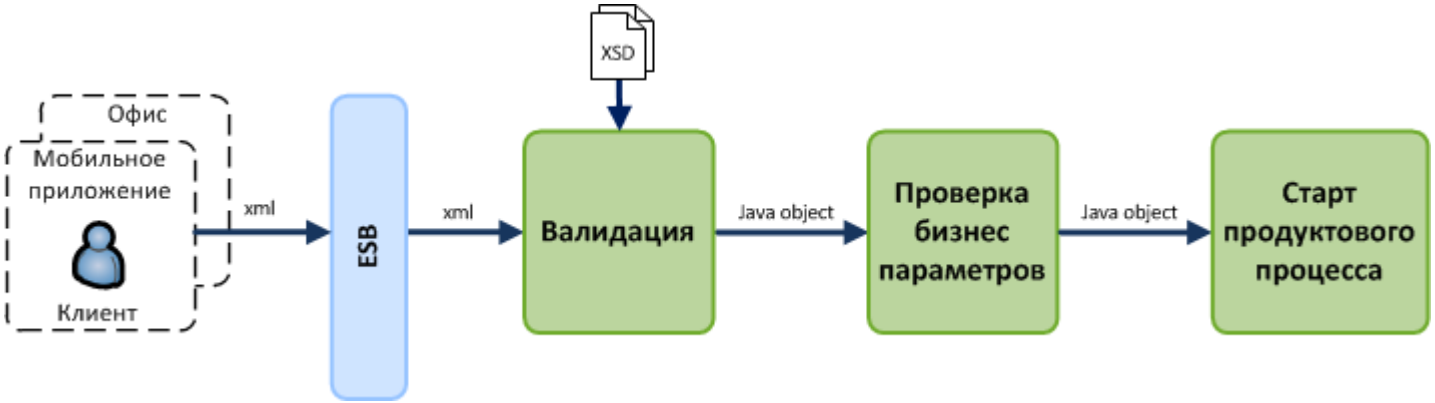

XSD
Офис
Клиент
xml
ESB
xml
Валидация
Java object
Проверка бизнес параметров
Java object
Старт продуктового процесса

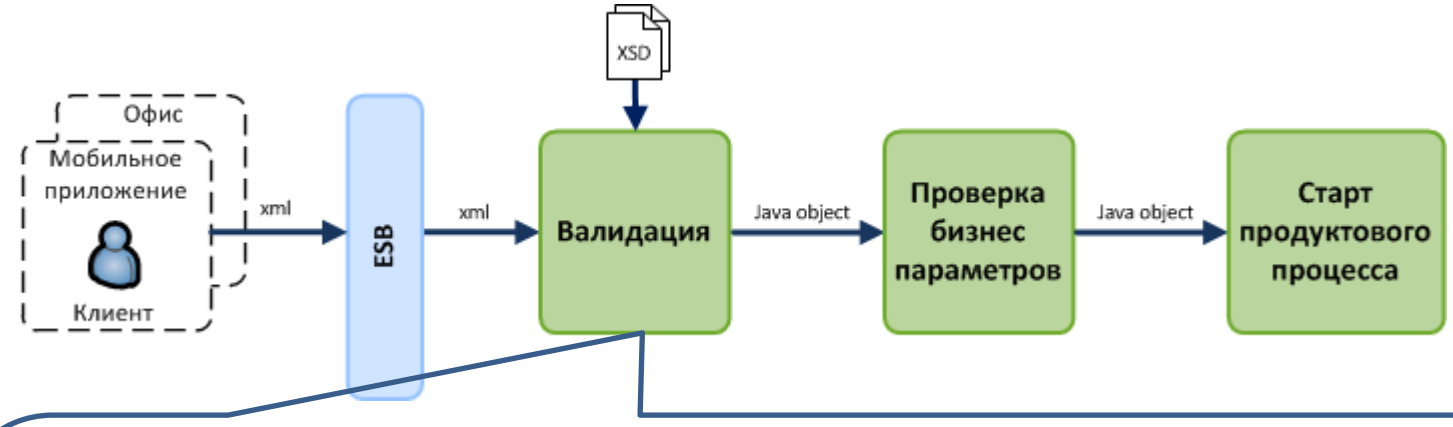



```
public ValidatorEntity<ChannelEntity> validate(InputStream xml, Schema xsd) {
    SchemaFactory schemaFactory = SchemaFactory.newInstance(...);
    ChannelEntity entity;

    try {
        jaxbContext = JAXBContext.newInstance(...);
        Unmarshaller unmarshaller = getJAXBContext().createUnmarshaller();
        unmarshaller.setSchema(xsd);
        ValidationEventCollector validationEventCollector = new ExtValidationEventCollector();
        unmarshaller.setEventHandler(validationEventCollector);
        entity = (ChannelEntity)unmarshaller.unmarshal(xml);
    } catch (JAXBException e){
        return new ValidatorEntity<>(e.getMessage(), ValidatorStatus.ERROR);
    }

    return new ValidatorEntity<>(entity, parseValidationEventCollector(validationEventCollector));
}
```

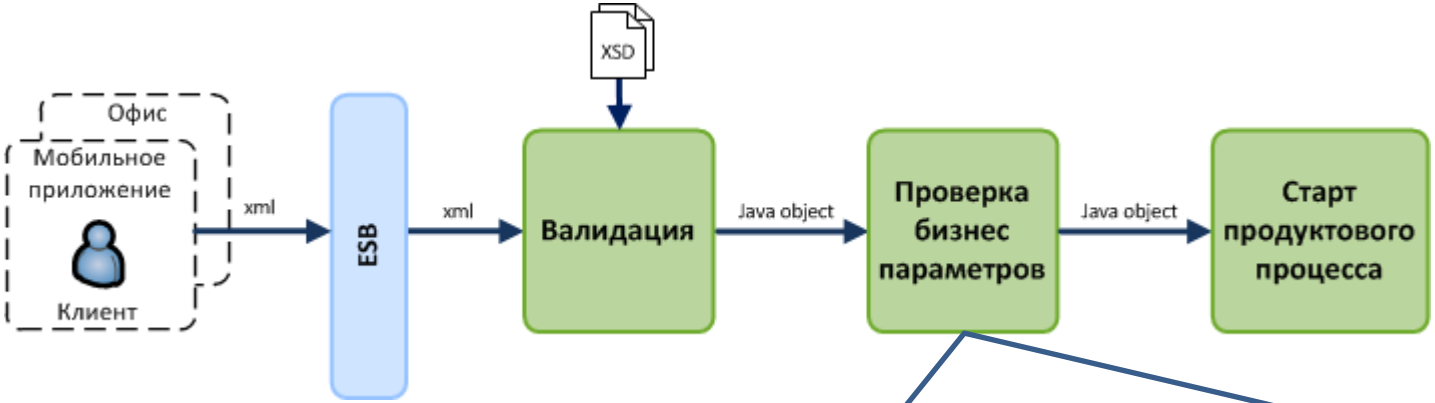



```
public CheckEntity<ChanelEntity> check(ChanelEntity entity) {
    CheckEntity<ChanelEntity> checkEntity = new CheckEntity(entity);

    if (entity.getCarInfo() != null) {
        CarInfo carInfo = entity.getCarInfo();
        if (!checkWithRegExp(carInfo.getPassportNum(), '^[0-9A-ZA-Я\Ё]{4}$')) {
            checkEntity.addCheck(CAR_PASSPORT_NUM_CHECK, carInfo.getPassportNum());
        }
        if (!checkWithRegExp(carInfo.getPassportSeria(), '^[0-9]{6}$')) {
            checkEntity.addCheck(CAR_PASSPORT_SERIA_CHECK, carInfo.getPassportSeria());
        }
    }

    return checkEntity;
}
```


XSD
Офис
Мобильное приложение
Клиент
xml
ESB
xml
Валидация
Java object
Проверка бизнес параметров
Java object
Старт продуктового процесса




```
public ValidatorEntity<ChannelEntity> validate(InputStream xml, Schema xsd) {
    SchemaFactory schemaFactory = SchemaFactory.newInstance(...);
    ChannelEntity entity;

    try {
        jaxbContext = JAXBContext.newInstance(...);
        Unmarshaller unmarshaller = getJAXBContext().createUnmarshaller();
        unmarshaller.setSchema(xsd);
        ValidationEventCollector validationEventCollector = new ExtValidationEventCollector();
        unmarshaller.setEventHandler(validationEventCollector);
        entity = (ChannelEntity)unmarshaller.unmarshal(xml);
    } catch (JAXBException e){
        return new ValidatorEntity<>(e.getMessage(), ValidatorStatus.ERROR);
    }

    return new ValidatorEntity<>(entity, parseValidationEventCollector(validationEventCollector));
}
```




```
public CheckEntity<ChanelEntity> check(ChanelEntity entity) {
    CheckEntity<ChanelEntity> checkEntity = new CheckEntity(entity);

    if (entity.getCarInfo() != null) {
        CarInfo carInfo = entity.getCarInfo();
        if (!checkWithRegExp(carInfo.getPassportNum(), '^[0-9A-ZA-Я\Ë]{4}$')) {
            checkEntity.addCheck(CAR_PASSPORT_NUM_CHECK, carInfo.getPassportNum());
        }
        if (!checkWithRegExp(carInfo.getPassportSeria(), '^[0-9]{6}$')) {
            checkEntity.addCheck(CAR_PASSPORT_SERIA_CHECK, carInfo.getPassportSeria());
        }

        if (SALE_CHANEL_MOBILE.equals(entity.getSaleChannel())) {
            if (!checkWithRegExp(carInfo.getOsagoSeria(), '^[A-ZA-Я\Ë]+$')) {
                checkEntity.addCheck(OSAGO_SERIA_CHECK, carInfo.getOsagoSeria());
        }
    }

    return checkEntity;
}
```


Сайт
Офис
Мобильное приложение
Клиент
xml
ESB
xml
Оркестровщик
xml
XSD
Валидация
Java object
Проверка бизнес параметров
Java object
Старт продуктового процесса

Бизнес-логика в библиотеках


Бюро кредитных историй «Чистая совесть»
Кредитное бюро «Сбербанка»
Объединенное кредитное бюро «Национальный хантер»
Межрегиональное кредитное бюро
Сбор данных по кредитной заявке
Профиль клиента «Сбербанка»
Национальное кредитное бюро
Сотовый оператор
Пенсионный фонд России
Столичное кредитное бюро

```
public class Liability extends AbstractLiability {
    . . .

    public boolean isSimilar(Liability theOther) {
        String clsNum = number.toString();
        String clsOther = theOther.number.toString();
        Integer days = new Integer(0);
        Integer otherDays = new Integer(0);

        if (SOURCE_TSM.equals(source)) {
            clsNum = clsNum.replaceFirst("^0*", "");
            clsOther = clsOther.replaceFirst("^0*", "");
        }
        if (SOURCE_DCAS.equals(source)) {
            days = daysRemaining(dateRedemptionScheduled);
            otherDays = daysRemaining(dateRedemptionScheduled);
        }
        return dateStart.equals(theOther.dateStart)
                && amountInitial.equals(theOther.amountInitial)
                && clsNum.equals(clsOther) && days.equals(otherDays)
                && currency.equals(theOther.currency);
    }

    . . .
}
```


Бюро кредитных историй «Чистая совесть»
Кредитное бюро «Сбербанка»
Объединенное кредитное бюро «Национальный хантер»
Конвертер
Конвертер
Конвертер
Межрегиональное кредитное бюро
Конвертер
Сбор данных по кредитной заявке
Конвертер
Профиль клиента «Сбербанка»
Конвертер
Конвертер
Конвертер
Конвертер
Национальное кредитное бюро
Сотовый оператор
Пенсионный фонд России
Столичное кредитное бюро

```
public class Liability extends AbstractLiability {
    . . .

    public boolean isChallengerBetter(Liability challenger) {
        if (SOURCE_DCAS.equals(source))
            return true;

        if (SOURCE_NBCH.equals(challenger.source) && SOURCE_GP.equals(source))
            return false;

        if (SOURCE_EI.equals(challenger.source) && SOURCE_GP.equals(source))
            return false;

        return true;
    }

    . . .



}
```

```
public class Liability extends AbstractLiability {
    . . .
    private double rate;

    . . .

    public boolean isChallengerBetter(Liability challenger) {
        return challenger.rate > rate ? true : false;

    }

    . . .









}
```

Как с этим бороться?

- **Code Review**

Как с этим бороться?

- **Code Review**
- **Design Review**

Как с этим бороться?

- **Code Review**
- **Design Review**
- **Ретроспектива релизов**

Как с этим бороться?

- **Code Review**
- **Design Review**
- **Ретроспектива релизов**
- **Обучение**

Как с этим бороться?

- **Code Review**
- **Design Review**
- **Ретроспектива релизов**
- **Обучение**
- **Формирование продуктовых команд**


TEXAS INSTRUMENTS
TI-83 Plus

Чего мы добились

- **Повышение надежности системы**

Чего мы добились

- **Повышение надежности системы**
- **Уменьшение утилизации ресурсов на 30%**

Чего мы добились

- **Повышение надежности системы**
- **Уменьшение утилизации ресурсов на 30%**
- **Сокращение времени вывода в продакшен**

Чего мы добились

- **Повышение надежности системы**
- **Уменьшение утилизации ресурсов на 30%**
- **Сокращение времени вывода в продакшен**
- **Разделение монолита на микросервисы**

Выводы

- **Не допускать появление «божественного» кода**
- **Не бояться оправданного дублирования кода**
- **Вычленять бизнес-логику из библиотек**
- **Строго придерживаться архитектуры**