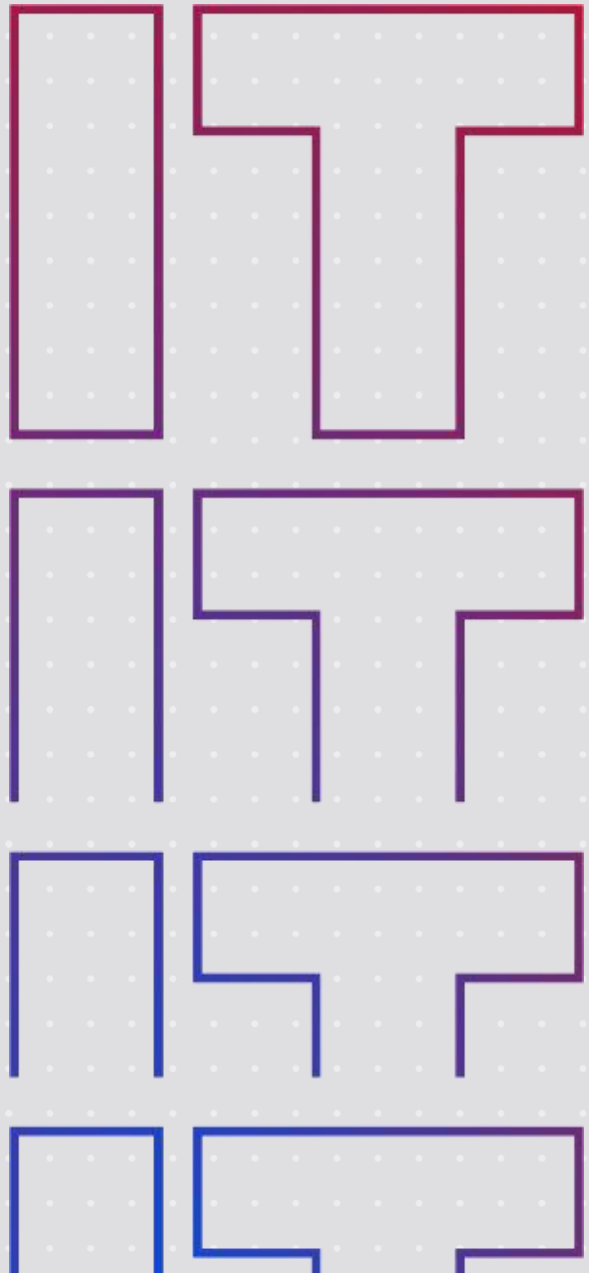
Распил монолита на микросервисы

Создание команды, выбор архитектуры и технологий



Сергей Бондарчук
Team Lead, Магнит

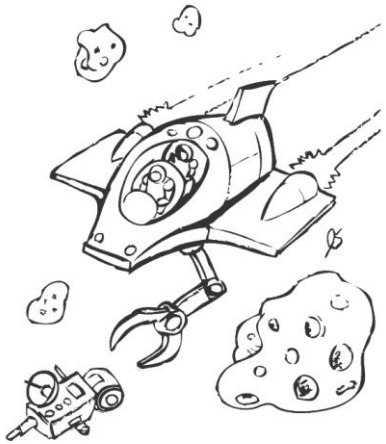
- Руководитель отдела системного анализа.
- Более 10 лет занимается внедрением IT продуктов в разных компаниях.
- Последние 6 лет работает в команде IT Магнит.
- Тимлид команды, занимающейся создание м сервисов по автоматизации транспортно й логистики.

О монолите и его проблемах

О подходе к созданию команд

О подходе к созданию сервисов

Какой был план



Что пошло не так

О монолите

Система управления транспортом - осуществляет планирование и контроль более 10 тысяч транспортных средств ONLINE

Основные функции

- **Подготовка и планирование транспорта и водителей на рейсы**
- **Агрегирование заказов и заявок на перевозку**
- **Формирование маршрутов движения ТС (Транспортная задача)**
- **Сервисы картографии, навигации и маршрутизации для грузового транспорта**
- **Алгоритм маршрутизации прямой доставки**
- **Расчет расстояний между объектами**
- **Контроль транспорта компании (Датчики телеметрии)**
- **Контроль отклонений фактического маршрута от планового**
- **Система контроля отклонений** (мониторинг более 80 событий)

Каким был монолит

Монолит – дата вывода в ПРОД 2008 год

Не виртуальный сервер
144 Ядра
1 536 Гб RAM (1.5 TB)
БД pgSQL

> 500 000
Строк кода в БД

>3ТБ+3ТБ
Размер БД

>25 000
Активных транзакций в сек.

>400 000 000
Чтений строк в сек.

>1 000 000
Записывается строк в сек.

Проблемы монолита

Почему нельзя было идти по старому

- **Только вертикальное масштабирование**
- **Блокировки в БД**
  - **слишком много пересекающихся функций**
  - **большое количество триггеров**
  - **операции чтения/записи занимают много времени**
- **Тяжелое сопровождение**
- **Большое количество легаси кода**
- **Зависимость функционала от версии платформы** (СУБД, библиотеки процедурного кода)
- **Долгое восстановление после простоя** (1 час простоя равен 3 часам восстановления)

Проблемы подхода

Как работали до начала распила - Водопад

Смена оргструктуры

Выделение команды

Анализ

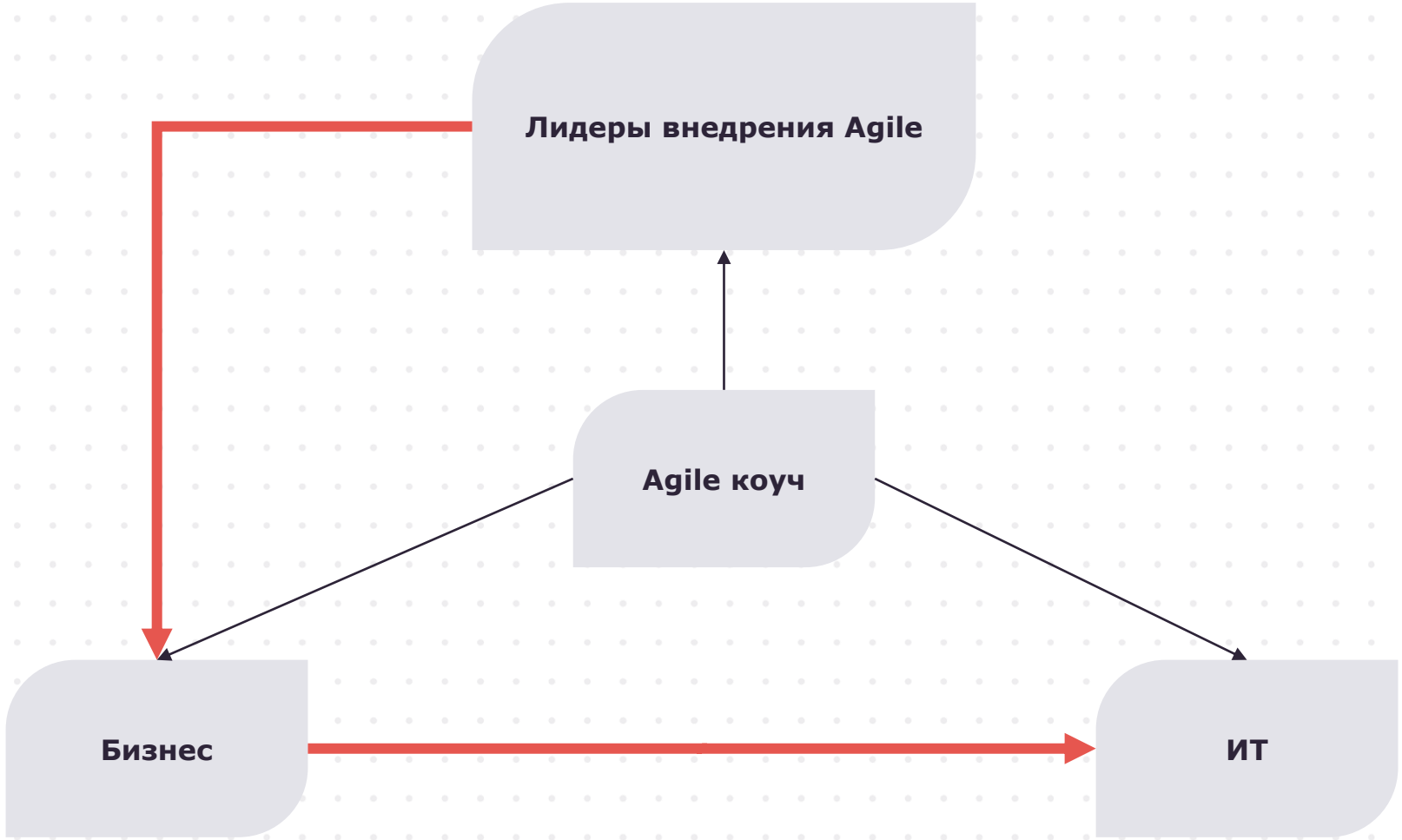
Тестирование

Разработка

DevOps

Смена методологии

Переход на гибкие подходы

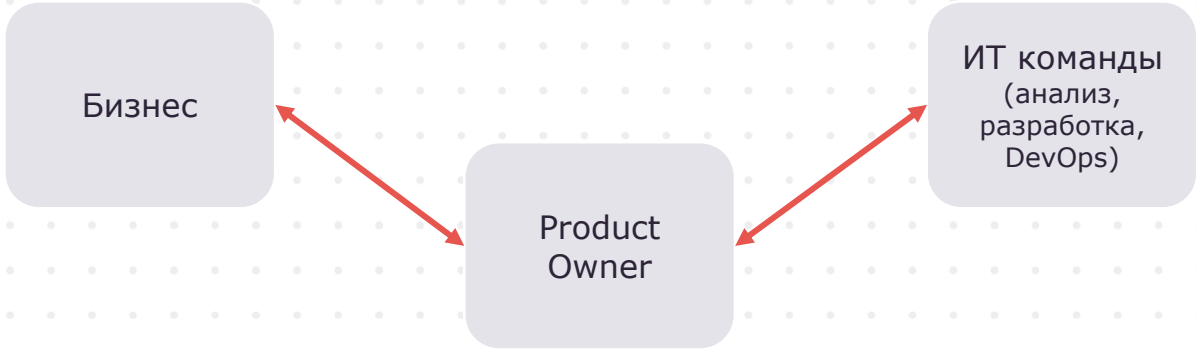


Создание команды

Новый подход

Полная очистка старой очереди

Экономические обоснования всех задач



Confluence

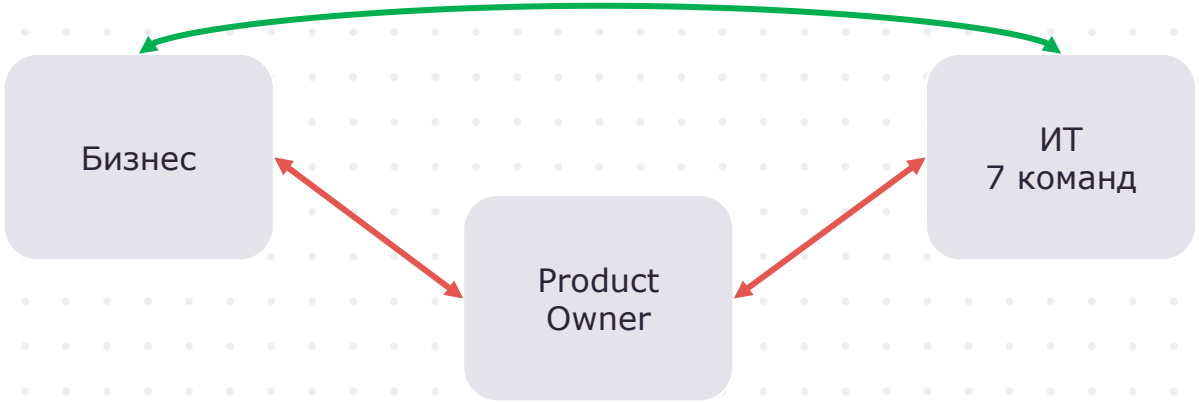
Jira Software

GitLab

Archi
archimate modelling

Создание команды

Как мы работаем сейчас



Upstream – управление входящим потоком

Downstream - заботимся о сокращении времени цикла, повышении эффективности нашего процесса и постоянном повышении качества продукта или сервиса

План распила монолита

Разбиения монолита на бизнес функции и обеспечивающих их приложений

15 функций
2 интерфейса
3 приложения обеспечивающих функции

Деление сервисов на команды

Создание команд по бизнес функциям ИС

1. СУТ (Ядро)

1. Инструмент управления ресурсами транспорта

3. Сервис расчета расстояний между объектами

3. Планшет водителя экспедитора

5. Система контроля отклонений

6. Рабочее место маршрутизатора

6. Сервис поиска догрузов по пути следования ТС

2. Межсклад

3. География

4. Телеметрия

5. Портал торгов

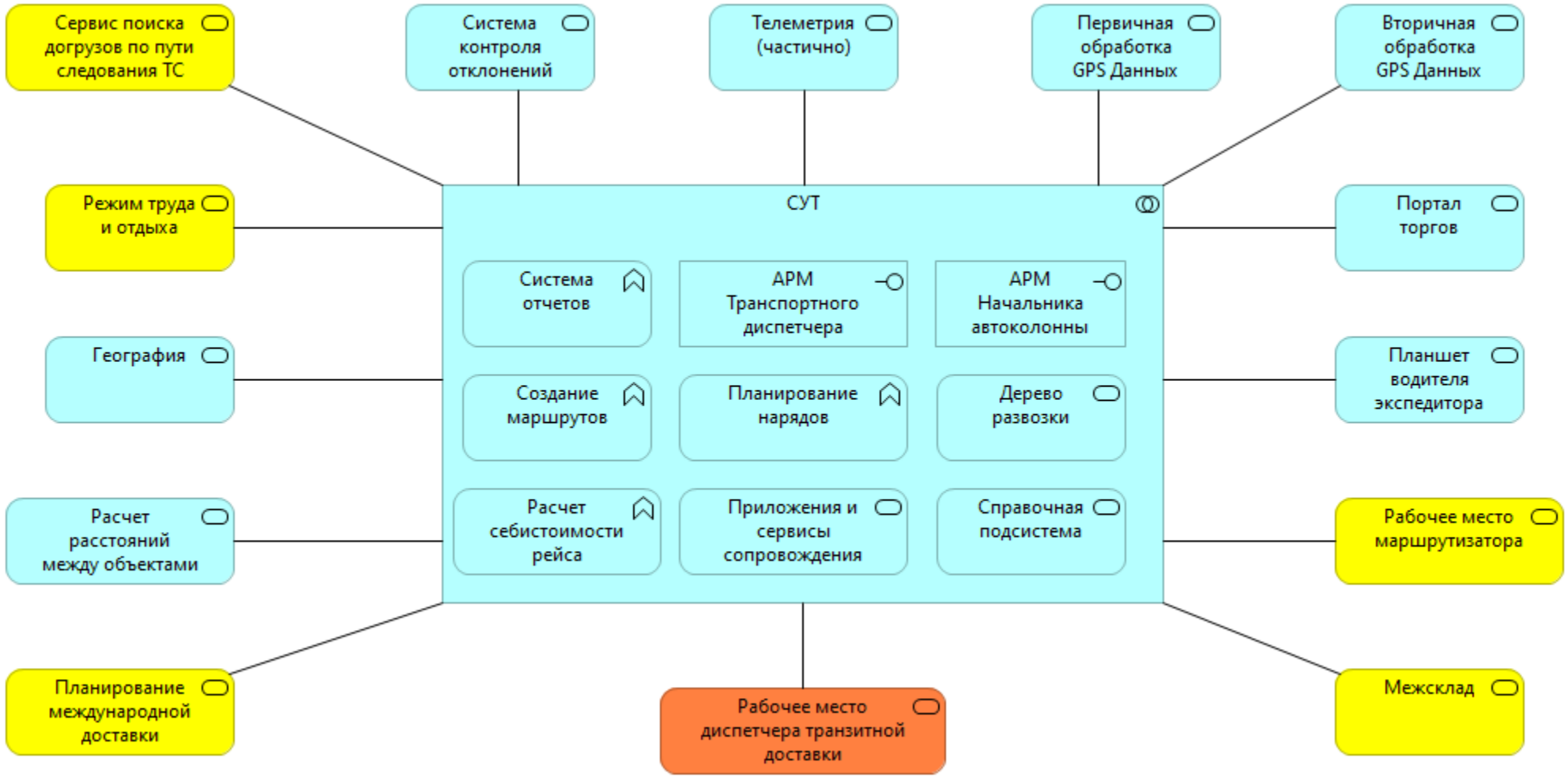
6. Сервис РТО

7. Рабочее место диспетчера транзитной доставки

Было сформировано 7 команд, выделено 14 крупноблочных сервисов

Распил монолита

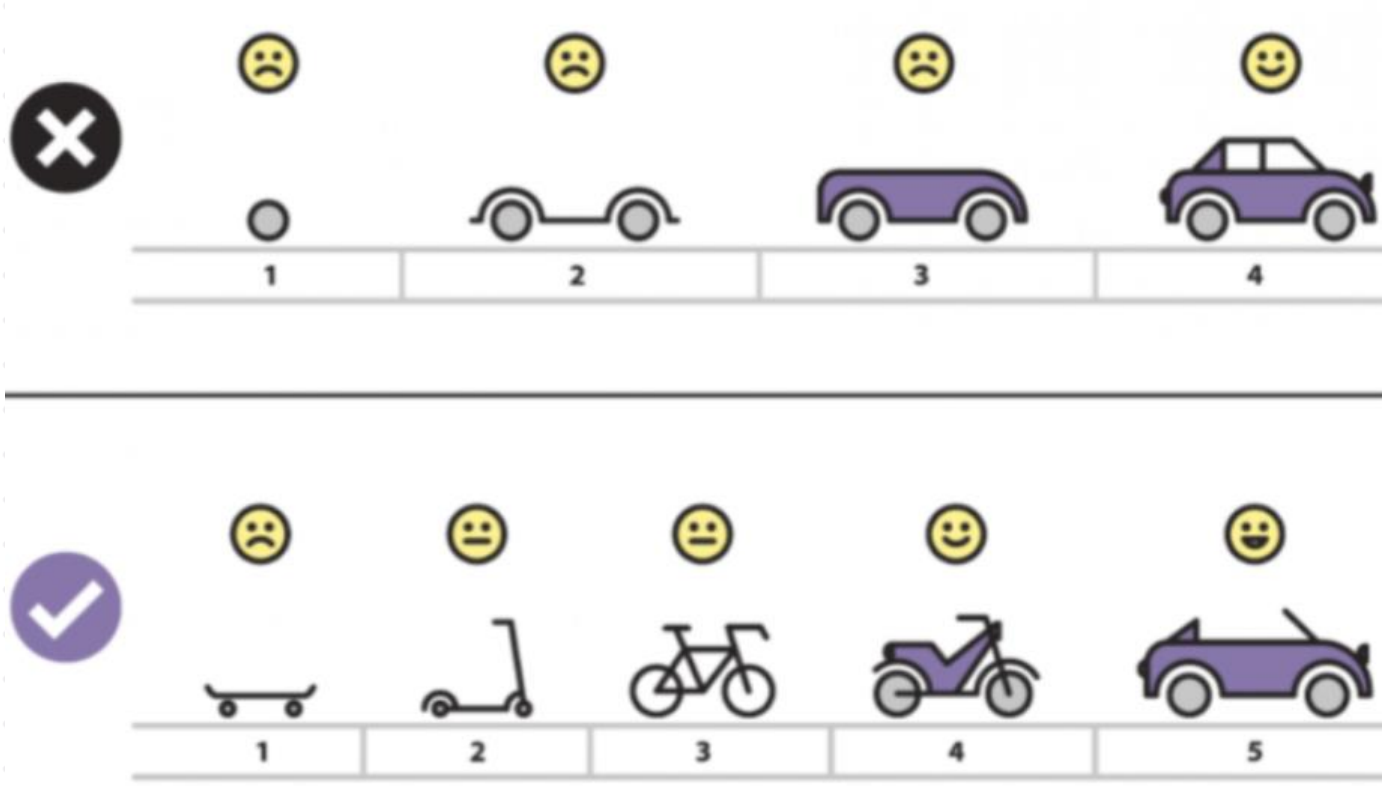
Целевые технологии и прототипы



О подходе к созданию сервисов

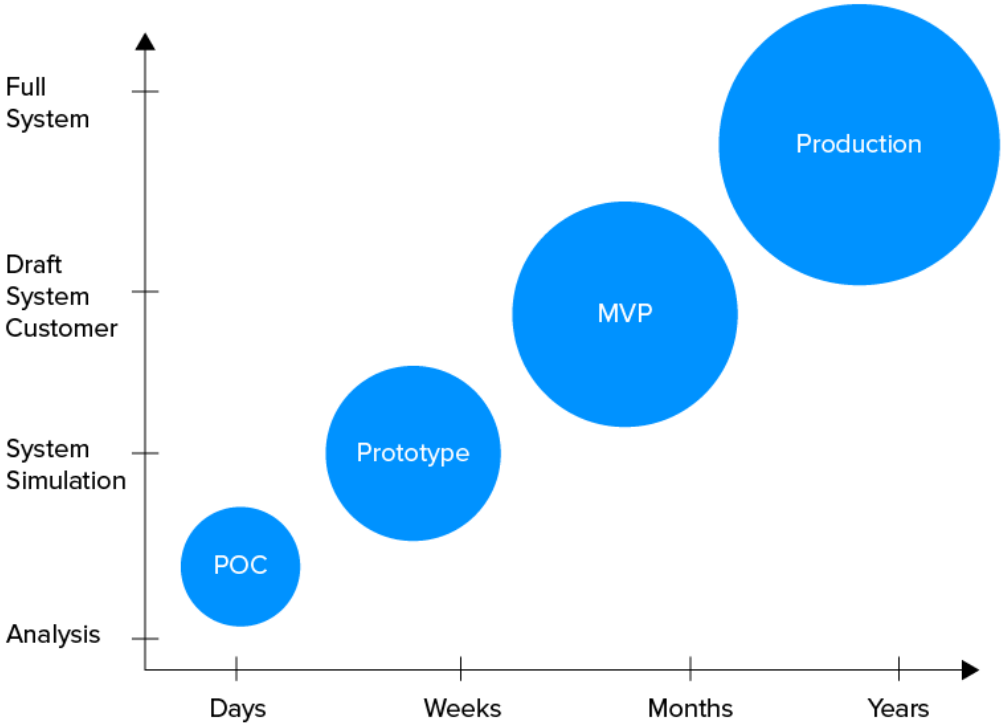
Быстрый фейл - это приемлемый результат, Создание функционала важней документирования

- **MVP -** Minimal Viable Product (минимально жизнеспособный продукт)
- **Массовое создание прототипов**



О подходе к созданию сервисов

Прототипирование и MVP



MVP — версия продукта, которая имеет минимальный набор функций исключительно для реализации бизнес-цели, обычно является первой версией продукта.
Используется целевыми пользователями

Prototype — используется для демонстрации какой-либо части системы, обнаружения ошибок в ней, опроса ключевых пользователей.
Используется внутри команды и для демонстрации стейкхолдерам

Прототипирование

Преимущества

- Можно быстро подтвердить или опровергнуть гипотезы
- Дает возможность совершить быстрый фейл
- Нет ограничений по технологиям
- Не нужно выделение большого количества ресурсов
- Сокращение времени и стоимости разработки
- Повышает уверенность в успешности результата

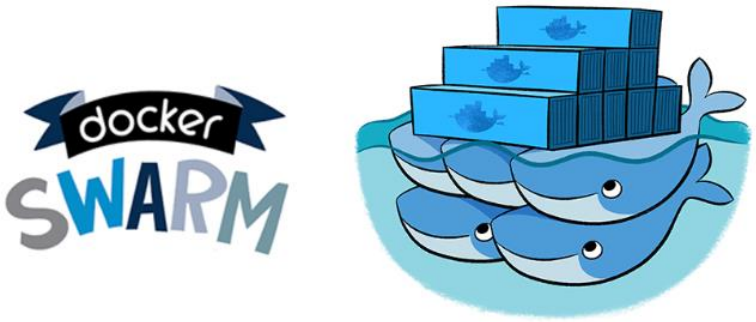
Прототипирование

Недостатки

- **Недостаточный анализ:** концентрация на ограниченном прототипе может отвлечь разработчиков от правильного анализа всего проекта
- **Путаница пользователя в отношении прототипа и результирующей системы:** пользователи могут ошибочно полагать, что прототип, который следует отбросить, является ненастроенной целевой системой
- **Разработчики цепляются за прототип:** разработчики могут слишком цепляться за прототип, для создания которого они приложили много усилий
- **Затягивание этапа прототипирования:** пользователи начинают привыкать работать в нецелевом решении

Прототипирование

Инструменты (интерфейс не обсуждаем)



Удобное развертывание и управление контейнерами (не нужно привлекать админов)

BI-отчетность: интегрирует данные из различных учетных систем и быстро их обрабатывает

Имеет место быть, если есть свободные руки которые владеют только MS Access

Что пошло не так

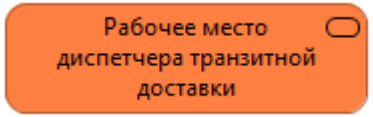
В подходе к созданию сервисов

- **При переходе на сервисную (микросервисную) архитектуру не уделили должного внимание проектировке архитектурного ландшафта:**
  - Аутентификация, Авторизация
  - Общие справочники и их синхронизация
  - Авто тесты, мониторинг и управление

- **Создание функционала важней документирования -** это было большой ошибкой

- **Массовое создание прототипов не всегда оправдано**

  Мы создали еще один монолит, от которого не можем уйти уже 4 года (из них 2 года убили на SAP TM)

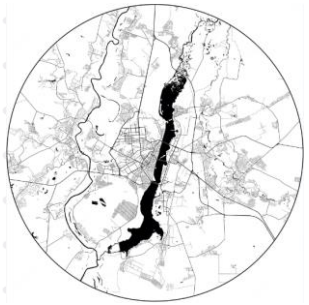


Что сейчас

Сервисы

Вынесли всю логику из монолита, он стал базой данных

Перешли на Keycloak, избавились от своего сервиса авторизации



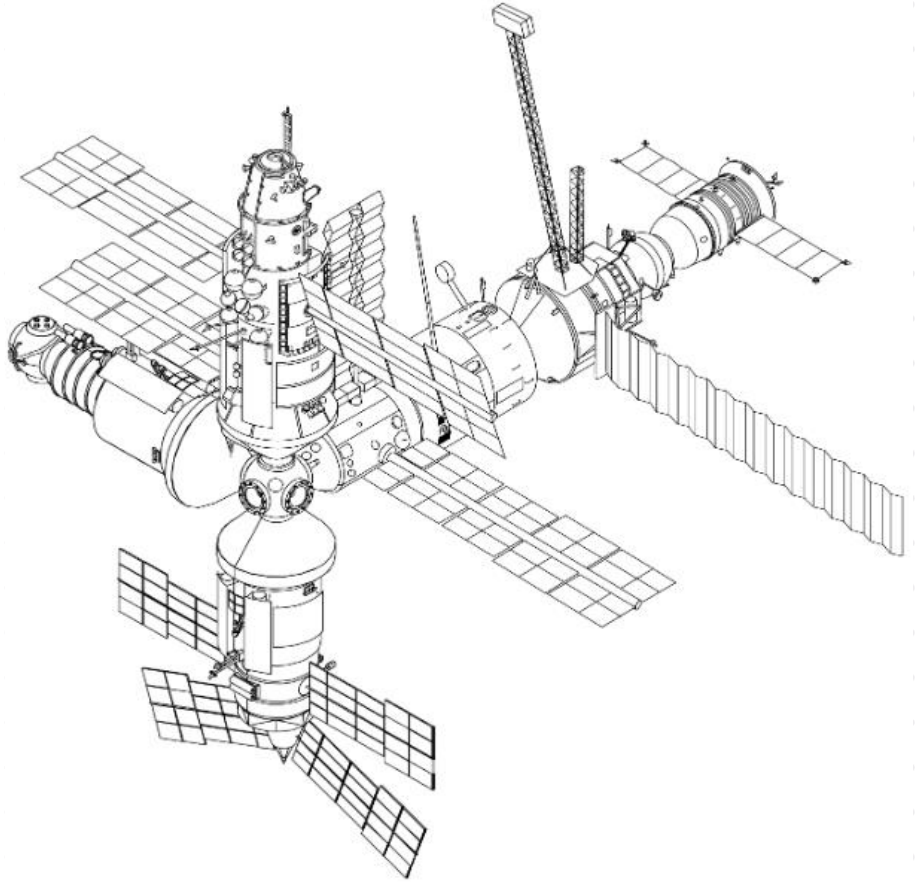
Вынесли все географические сервисы, завершили переход с OSRM на Valhalla

- Оптимизировали сервисы справочной информации
- Пересмотрели очереди (в некоторых случаях перешли с MongoDB на Reddis)
- Объединили или удалили сервисы, которые дублируют функционал

Начали распил прототипа, который стал монолитом №2

- Провели анализ функционала
- Определили перечень стейкхолдеров
- Создали команду
- Нарезали истории в Jira

Создали часть ключевых сервисов, сейчас занимаемся интеграцией



HABR

VC

Magnit

Спасибо за внимание!
Вопросы?

Контакты