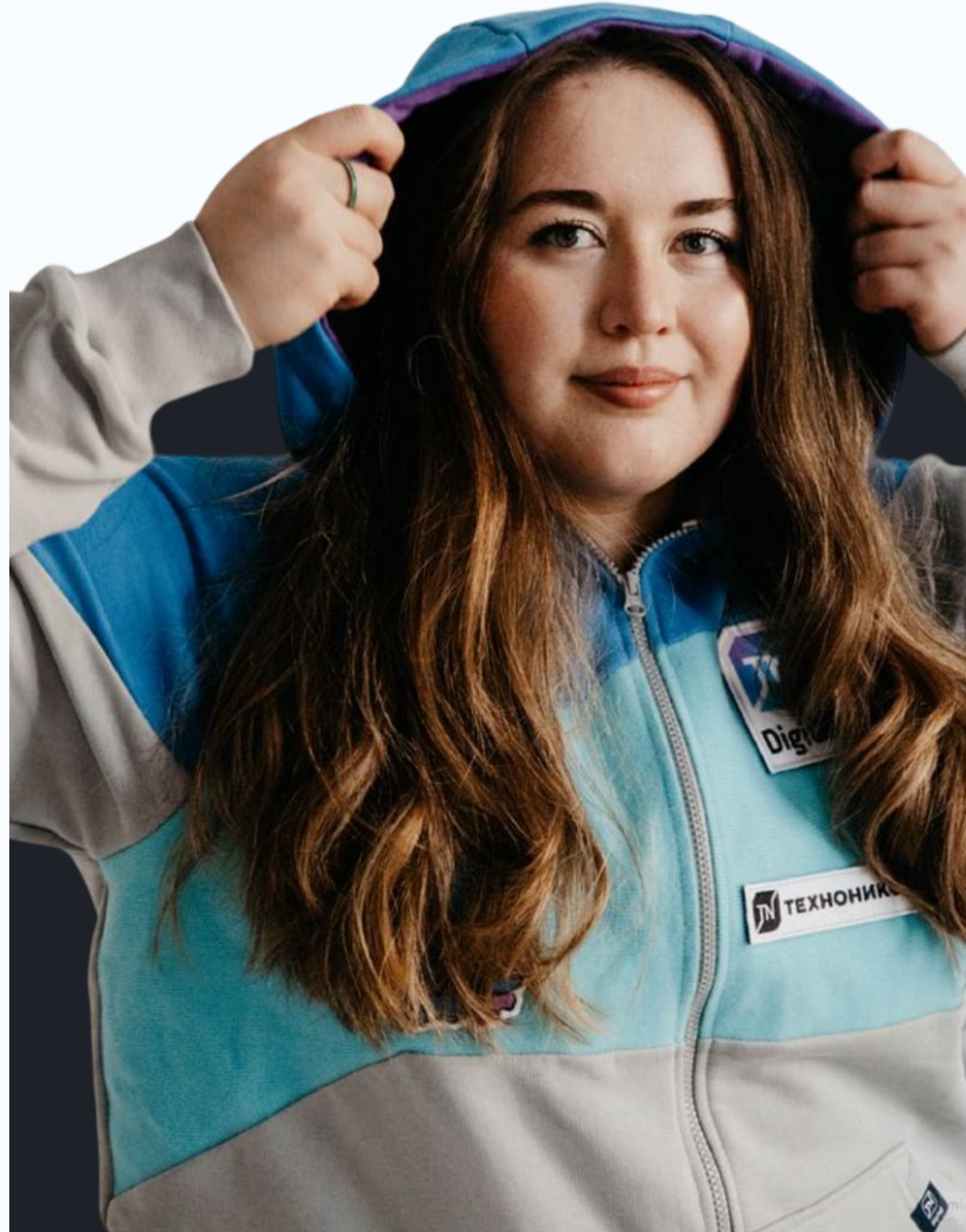




Секреты перехода к модульному монолиту

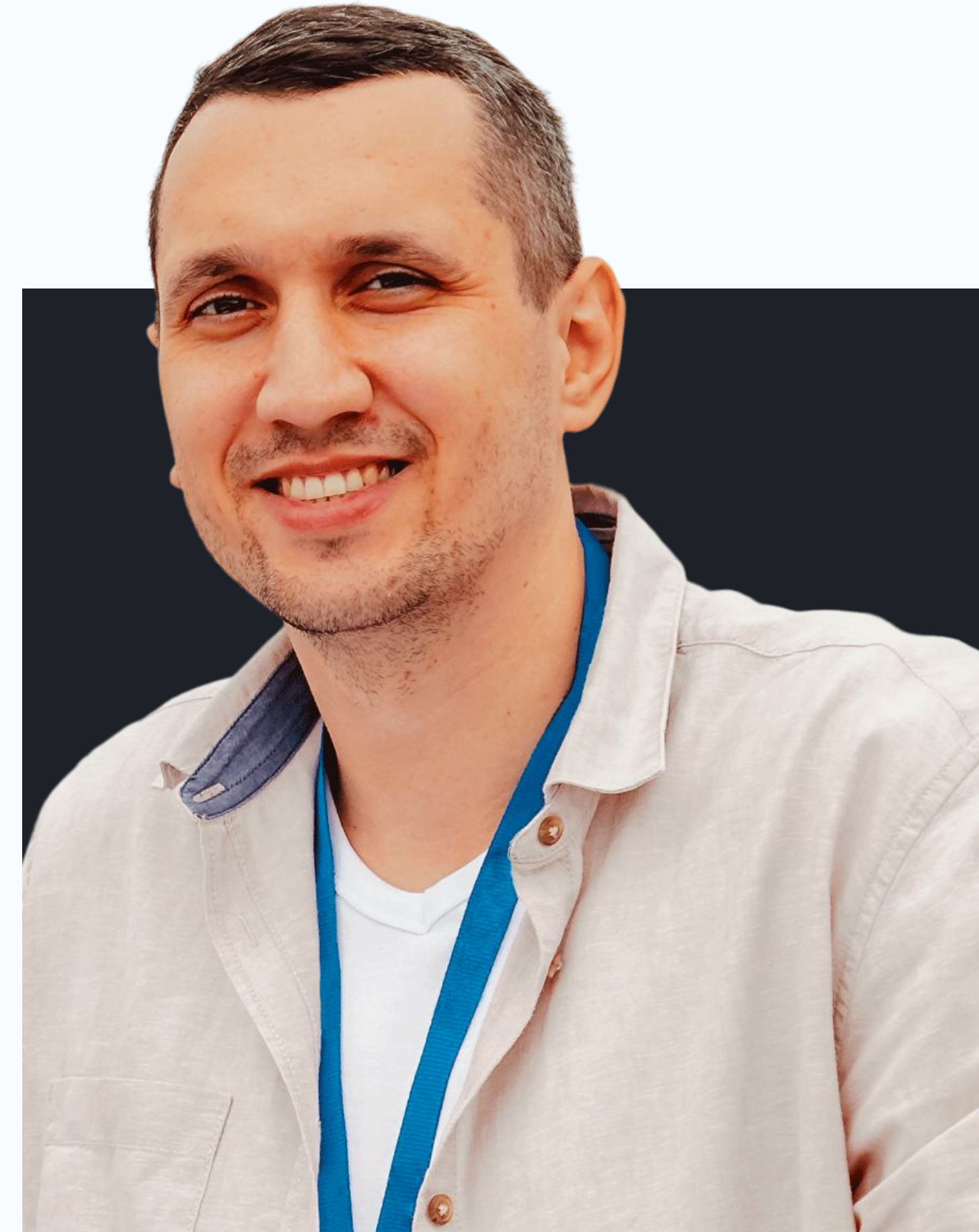
# Причем тут REST API?





## Альбина Бикбулатова

Discovery lead   Project Manager



## Станислав Балахонов

Delivery lead   BE-developer





**Бизнес процесс логистики ТЕХНОНИКОЛЬ  
и какое место мы занимаем**

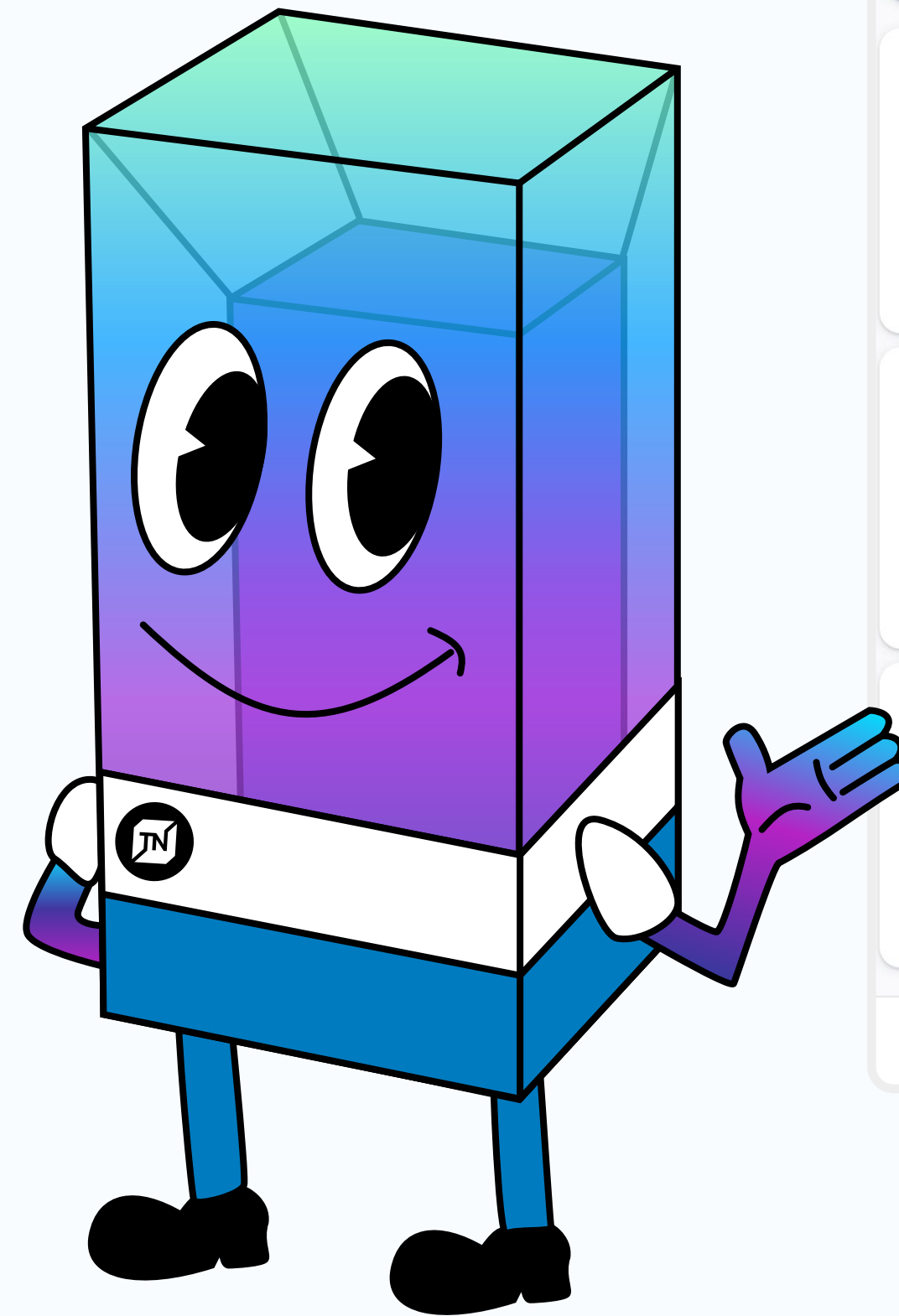


# Автоматизация бизнес-процессов





# Transportation Management System



Скриншоты веб-интерфейса системы управления транспортом (ТМ) на платформе 2С.

**Мои заявки (16)**

- Назначена** ✓  
№ 190 28 Фев - 01 Мар  
Нижний Новгород → Санкт-Петербург  
Стройматериалы / 20 000 кг / 90 м3  
₽ 55 000 / без НДС [Подробнее >](#)
- В работе** 🚚 🚚 🚚  
№ 190 28 Фев - 01 Мар  
Нижний Новгород → Санкт-Петербург  
Стройматериалы / 20 000 кг / 90 м3  
₽ 55 000 / без НДС [Подробнее >](#)
- Завершена** ✓  
№ 190 28 Фев - 01 Мар  
Нижний Новгород → Санкт-Петербург  
Лес-кругляк - 2 900 кг · 13 м3  
₽ 55 000 / без НДС [Подробнее >](#)

Вкладки: НОВЫЕ, АУКЦИОН, МОИ, ЕЩЕ, ВОДИТЕЛЬ

Статус заявки-договора	Статус погрузки	Внеш №	Время на обработку (ч:мм)	Место погрузки	Дата погрузки	Место доставки	Дата доставки
--	--	ТН00003264	00:00	Курск	08.05.2019	Курск	11.05.2022
--	--	ТН00003264	00:00	Курск	09.05.2019	Курск	11.05.2022
❌ Ошибка подписа...	--	ТН00003264	00:00	Курск	10.05.2019	Курск	11.05.2022
✅ Подписано	🟢 Выезд	ТН00003264	00:00	Курск	10.05.2019	Курск	11.05.2022
✅ Подписано	🟢 Выезд	ТН00003264	00:00	Курск	07.05.2019	Курск	11.05.2022
--	--	ТН00003264	00:00	Курск	08.05.2019	Курск	11.05.2022
🚚 Ожидает вашей...	🟡 Въезд	ТН00003264	00:00	Курск	09.05.2019	Курск	11.05.2022
❌ Ошибка подписа...	--	ТН00003264	00:00	Курск	10.05.2019	Курск	11.05.2022
🚚 Ожидает вашей...	🟡 Въезд	ТН00003264	00:00	Курск	05.05.2019	Курск	11.05.2022
--	--	ТН00003264	00:00	Курск	06.05.2019	Курск	11.05.2022
🚚 Ожидает вашей...	🟡 Въезд	ТН00003264	00:00	Курск	06.05.2019	Курск	11.05.2022
🚚 Ожидает вашей...	--	ТН00003264	00:00	Курск	08.05.2019	Курск	11.05.2022
✅ Подписано	--	ТН00003264	00:00	Курск	07.05.2019	Курск	11.05.2022
--	--	ТН00003264	00:00	Курск	07.05.2019	Курск	11.05.2022
✅ Подписано	🟡 Въезд	ТН00003264	00:00	Курск	08.05.2019	Курск	11.05.2022
✅ Подписано	🟢 Приезд	ТН00003264	00:00	Курск	09.05.2019	Курск	11.05.2022
✅ Подписано	🟡 Въезд	ТН00003264	00:00	Курск	12.05.2019	Курск	11.05.2022
✅ Подписано	🟢 Приезд	ТН00003264	00:00	Курск	13.05.2019	Курск	11.05.2022
❌ Ошибка подписа...	🟢 Приезд	ТН00003264	00:00	Курск	14.05.2019	Курск	11.05.2022
🚚 Ожидает вашей...	--	ТН00003264	00:00	Курск	16.05.2019	Курск	11.05.2022

Элементов на странице 20 0-20 из 20



# Гибридная команда





R E S T

**Почему REST API?**



# Принцип работы REST API

## 1С-TMS

Polling — паттерн асинхронного обмена по REST API.

### + Плюсы

Простота разработки

Строгое разделение логики взаимодействия между системами

### - Минусы

Большое количество запросов

Есть пирожок? 😞



Нет 😞



Прошло 30 секунд

А сейчас? 😞



Все еще нет 😞



Прошло 30 секунд

А теперь? 😞



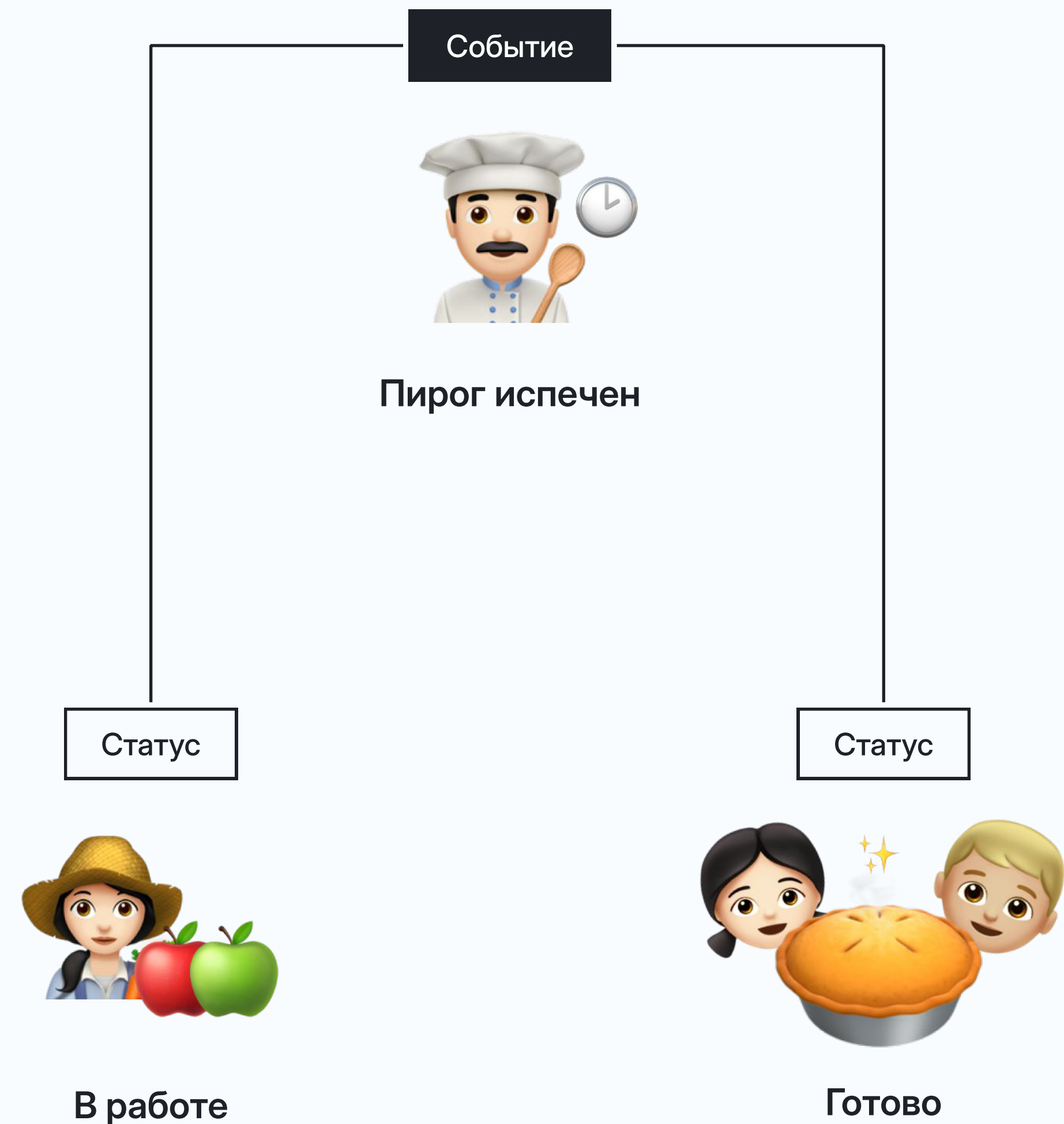
Держи, пока горячий! 🥰



# Статусно-событийная схема сущности

Статус — это состояние или условие, в котором находится объект или процесс.

Событие — это результат действия или изменение, которое происходит в системе или программе.





# Новый функционал 1С-TMS

В процессе проектирования бизнес-логики обмена 1С-TMS мы заложили много нового функционала, который в прежнем обмене не был реализован.

Версионирование заявки-договор

Предварительная заявка-договор

Претензии

Подписание соглашения по групповым заявкам

Перевозки КЗ

ЖД, контейнеры и экспорт

Новое флоу распределение заявки

Новая статусная модель маршрутного листа и заявки

Новая логика распределения заявки

Обмен с новой версией SignTN через REST API

■ Новый функционал    ■ Функционал кардинально меняющий логику





# Список Endpoints

Конечная точка (endpoint) — адрес, по которому отправляется запрос. Один и тот же эндпойнт может объединять несколько действий.

Legend:

- POST
- PUT
- PATCH
- DELETE
- GET

API Транспорт-2 <sup>v1</sup> <sup>OAS3</sup>  
 /v1/openapi.json  
 REST API для работы с логистическим сервисом ТехноНиколь

**bidding**

- PUT /public/client/v1/biddings/deadline Изменить Дедлайна Торгов На Повышение. Bidding-05
- PUT /public/client/v1/biddings/participants Изменить Видимость Заявки На Торгах. Bidding-06
- PUT /public/client/v1/biddings/status Отменить Торги На Повышение. Bidding-04
- PUT /public/client/v1/biddings/winner Назначить Победителя На Торгах. Bidding-07
- POST /public/client/v1/delivery-requests/allocation/bidding Распределить Заявку На Транспорт
- GET /web/expeditor/v1/bidding/counters Получить Счетчики Торгов

**processing**

- GET /public/client/v1/delivery-requests Просмотреть Список Заявок На Транс
- PUT /public/client/v1/delivery-requests Обновить Заявку На Транспорт (Черновик). Processing-09



## Количество изменений

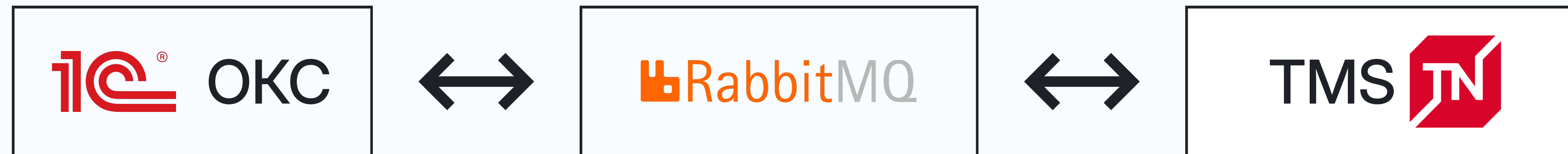
≈ 50 из 70

эндпоинтов влекли за собой либо коренное изменение логики работы, либо создание новой

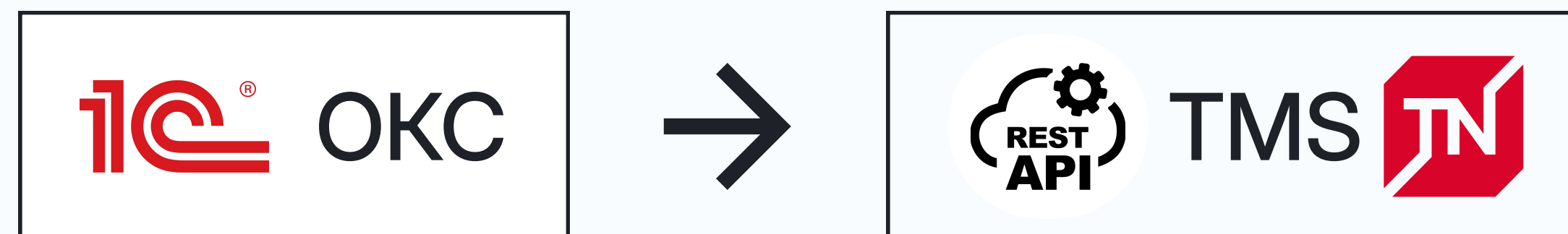


# Работа двух интеграций одновременно

## Асинхронная интеграция

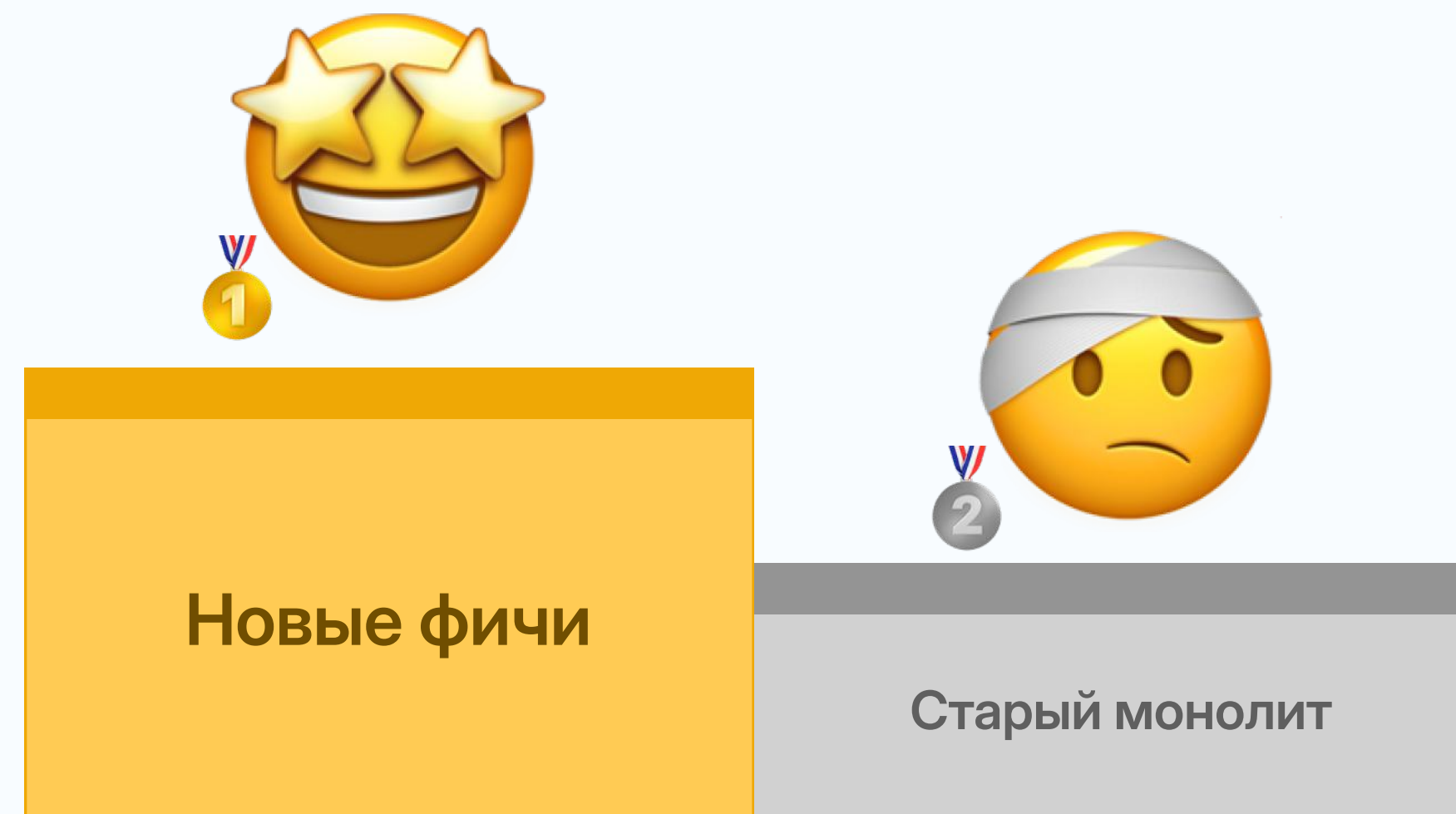


## Синхронная интеграция





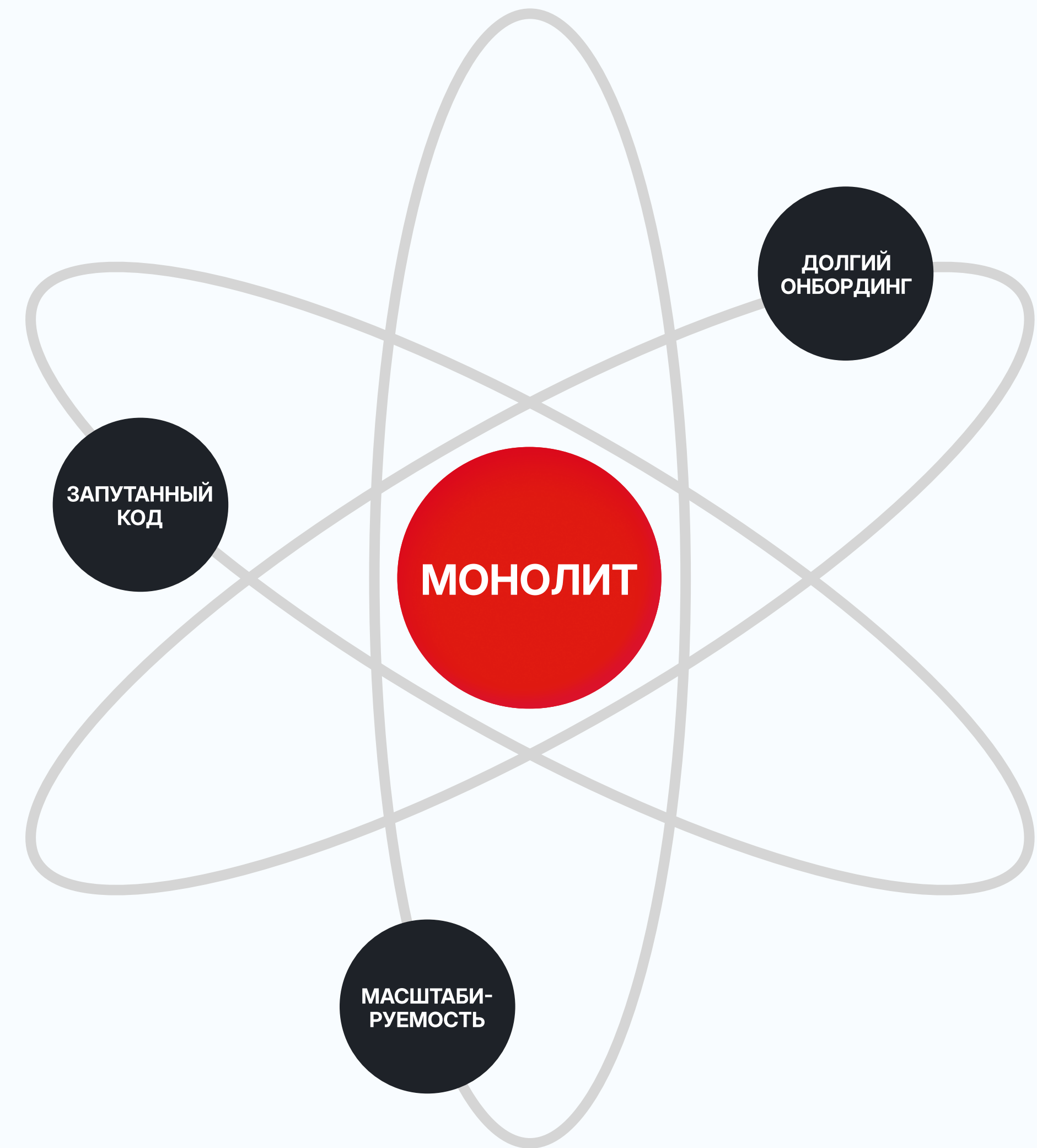
# Приоритеты разработки





## Боли монолита

Он оброс многими болезнями: большая связность кода, плохая масштабируемость, длительный онбординг новых участников и т.д.





# Хаос в бизнес-логике

И в эндпойнтах, и в функциях-обработчиках, и в ORM — всюду распределена бизнес-логика. Отсутствие порядка в структурировании кода пагубно влияло на внесение изменений, онбординг и масштабируемость.

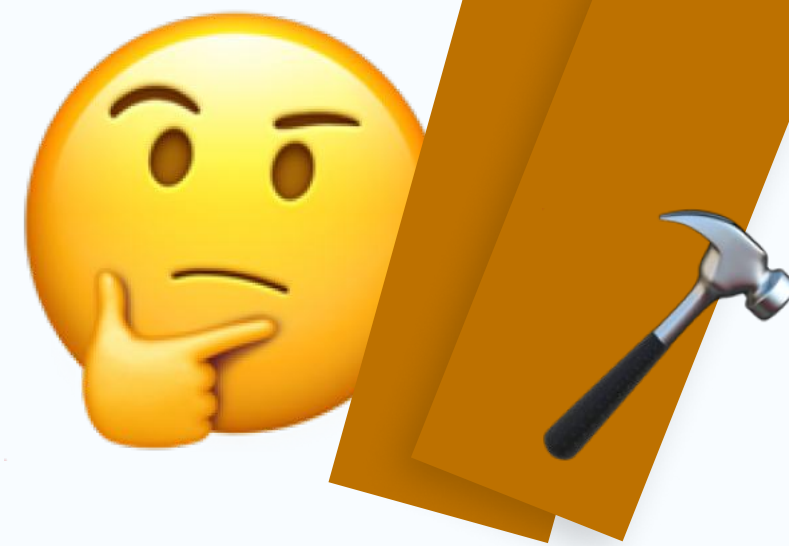




# Переход к новому монолиту

Мы приняли решение переходить на новую архитектуру приложения — построить новый монолит, но монолит модульный!

Пора обновиться





# Преимущества монолита и микросервисов

## Монолит

Проще релиз

Быстрее вызовы между модулями

Проще в управлении

Проще эксплуатация

Проще поддерживать согласованность данных

Легче проводить рефакторинг

## Микросервисы

Возможно обновление по частям

Более низкий TTM

Независимость релизов

Независимое масштабирование

Независимая разработка микросервисов

Высокая доступность отдельных модулей

Независимое тестирование микросервисов



# Monolith first

Monolith first потому что Модульный монолит включает плюсы и того и другого.

Проще релиз

Быстрее вызовы между модулями

Легче проводить рефакторинг

Проще в управлении

Проще поддерживать согласованность данных

Проще эксплуатация

Независимая разработка модулей

Независимое тестирование модулей

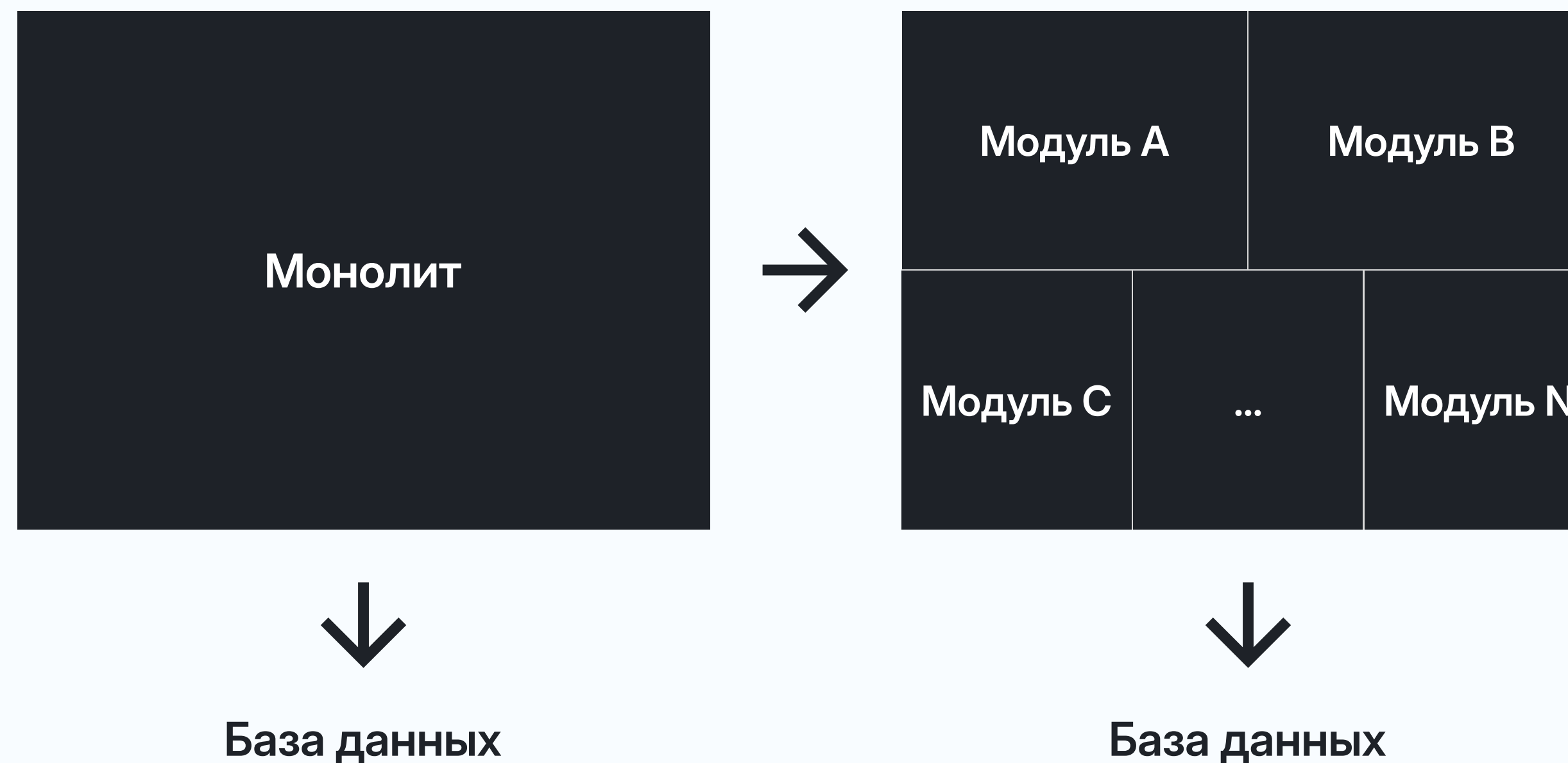
Более низкий ТТМ

\*Если вы смогли написать хороший модульный монолит, то с высокой вероятностью сделаете хорошую микросервисную архитектуру.



# Модульный монолит

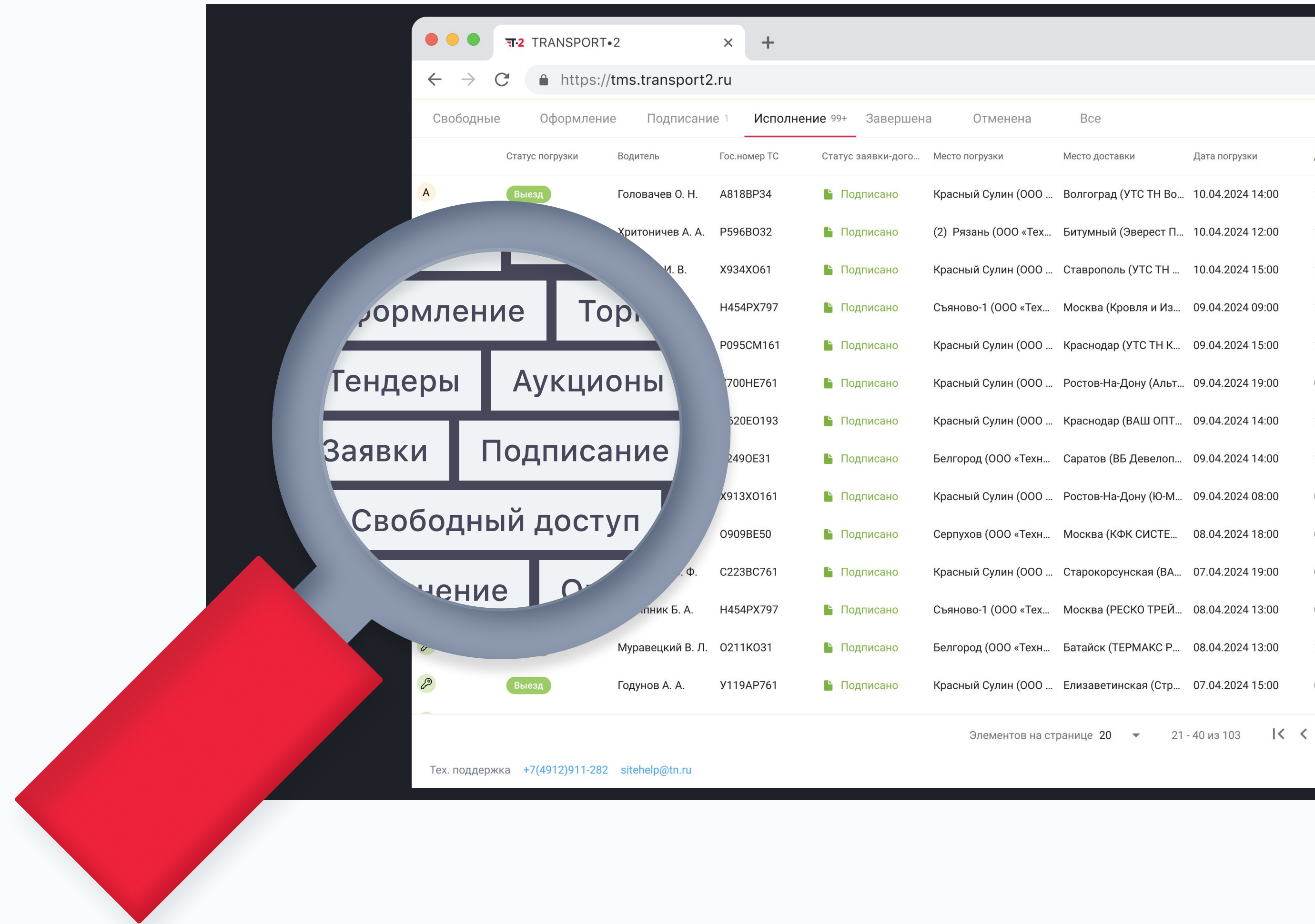
Модульный монолит — это такой монолит, где каждый модуль может функционировать отдельно. Т.е. представляет собой предметную подобласть или бизнес-подпроцесс.



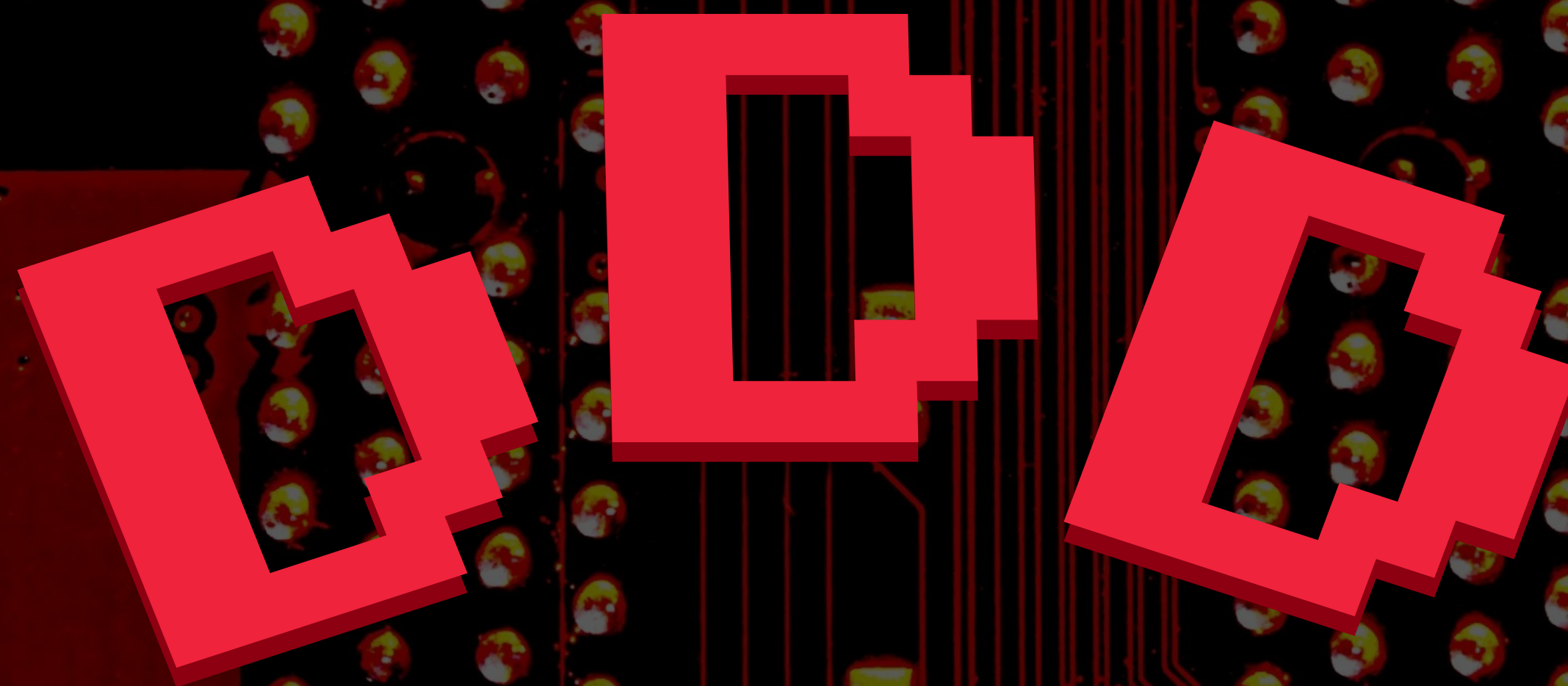


# Domain Driven Design

DDD (с англ. Предметно-Ориентированное Проектирование) — это подход к разработке программного обеспечения, нацеленный на изучение предметной области предприятия или его отдельных бизнес-процессов.







# Domain Driven Design

Почему DDD?



# Ограниченный контекст



# Единый язык

☆ ↩ ↲ ⊙ ∞ ∪ ♂ ♂



Опять на своем  
птичьем объясняешь!



# Практика Event Storming

EventStorming — это воркшоп с простой и четкой структурой, позволяющий собрать вместе экспертов различных компетенций для быстрого, совместного исследования сложной предметной области.



# События

Воркшоп начинается с коллективного поиска решений для событий, связанных с исследуемой предметной областью, которые расставляют в хронологическом порядке.





# Команды

Затем добавляем команды. Они, в свою очередь, описывают операции системы, в результате которых происходят события. Глобально — это методы, которые совершают манипуляции над объектами.



# Акторы

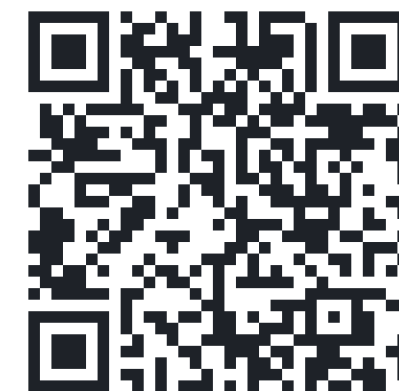
Мы добавляли на карту акторов — инициаторов событий в системе.





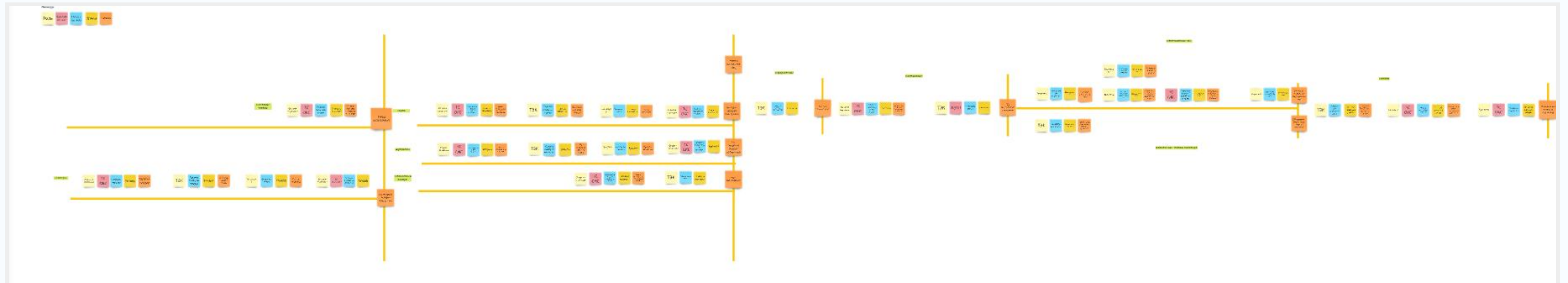
# Модели

Модель — это олицетворение бизнес-сущности в программном коде.



# Event Storming

## заявки на перевозку

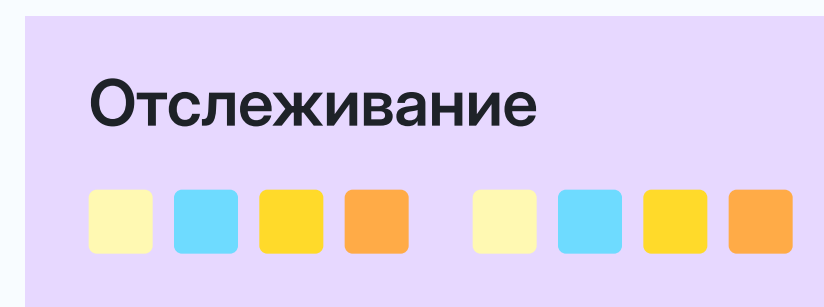
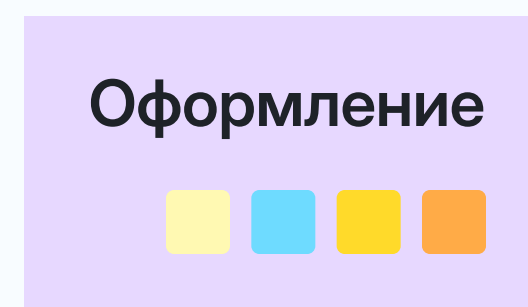
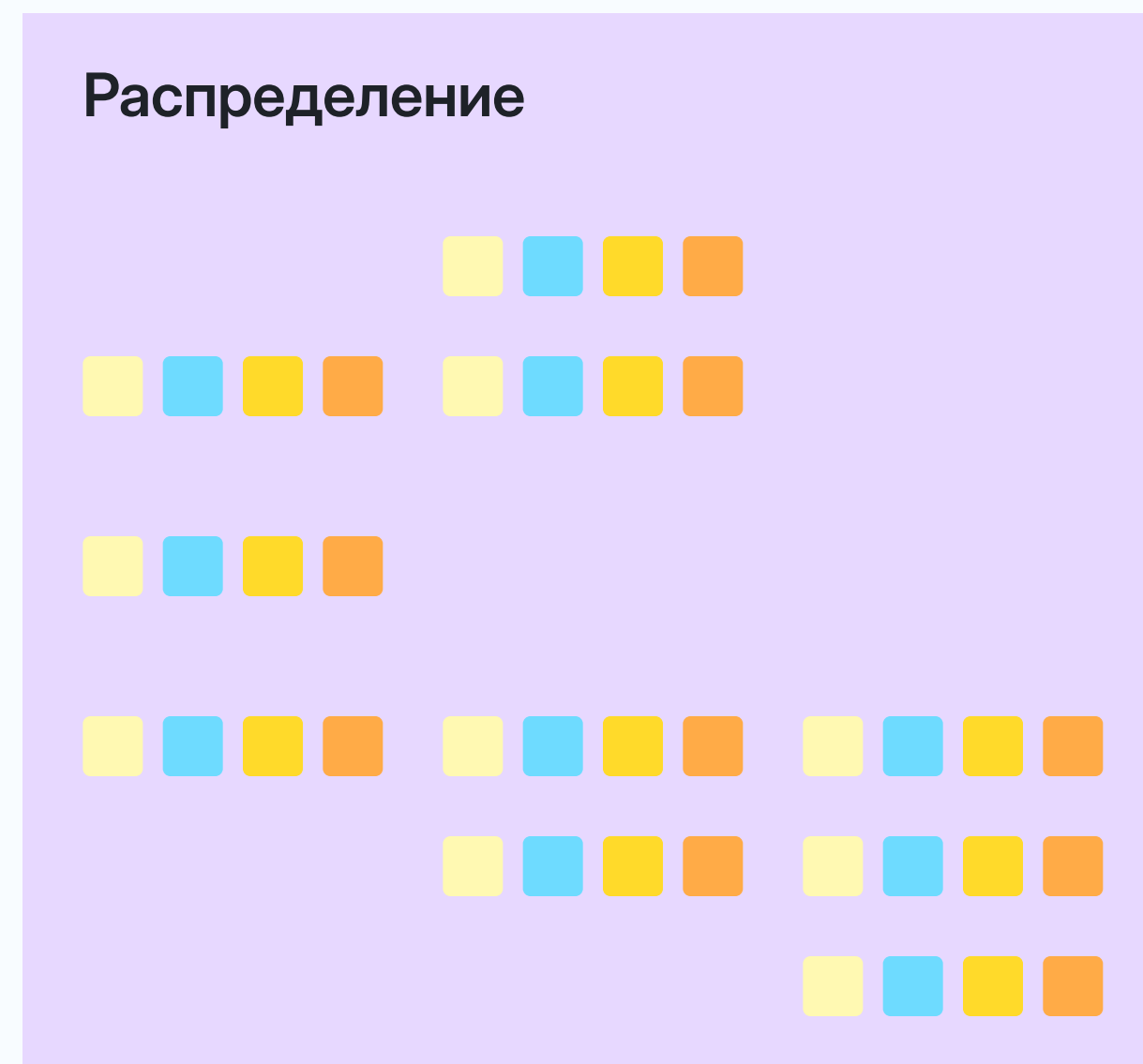






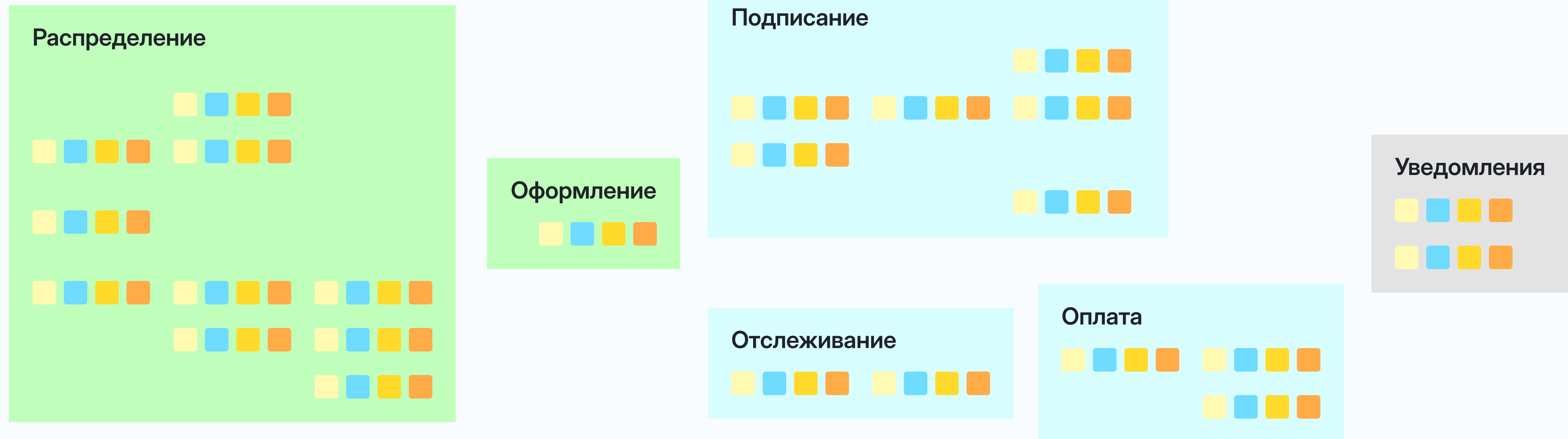
# Результаты Event Storming

Мы собираем связанные концепции в **домены**,  
которые позволяют разбить монолит на **модули/сервисы**.



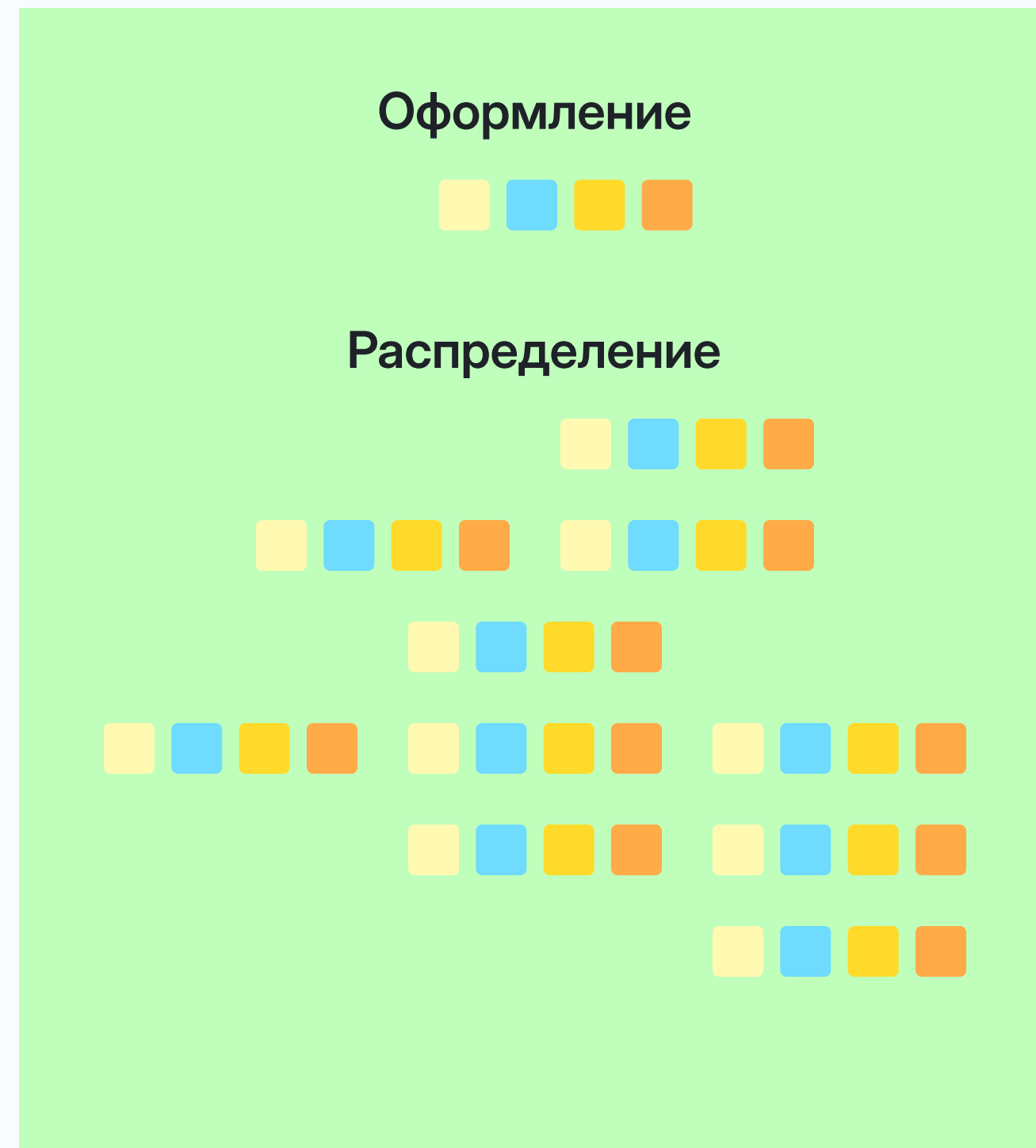
# Разбиение домена

Программные реализации поддоменов являются автономными модулями системы.

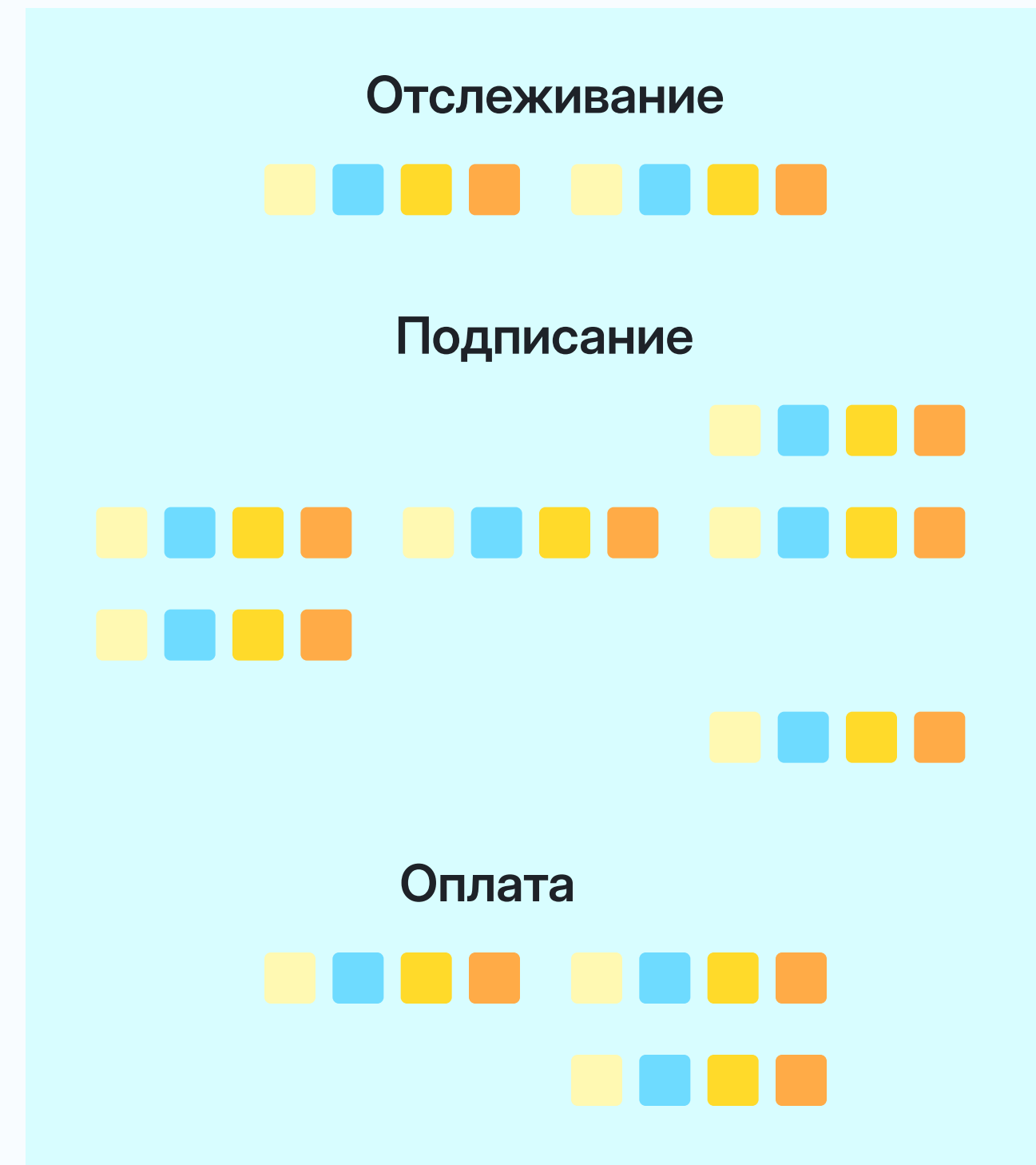




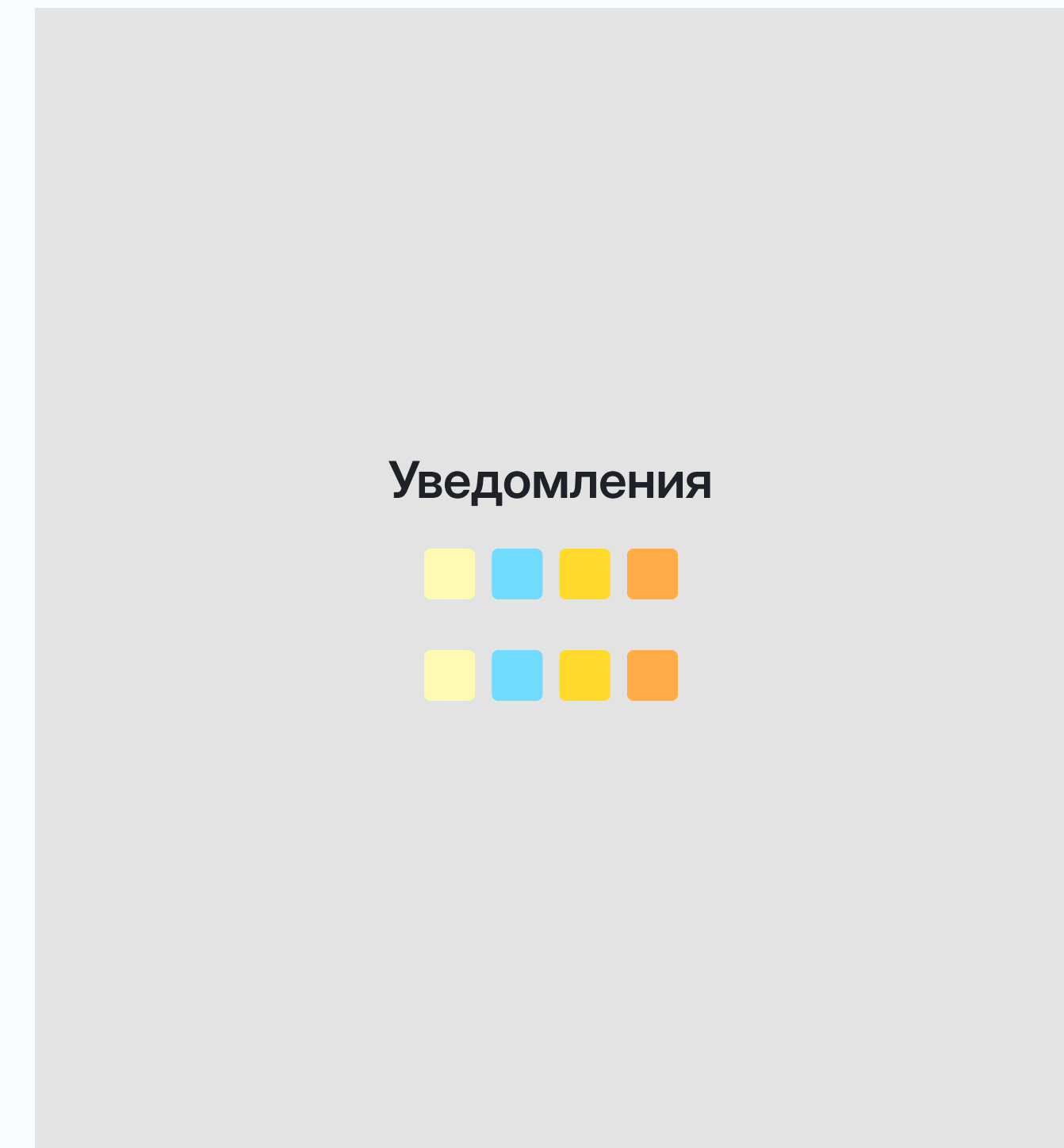
# Разбиение домена



Смысловое ядро



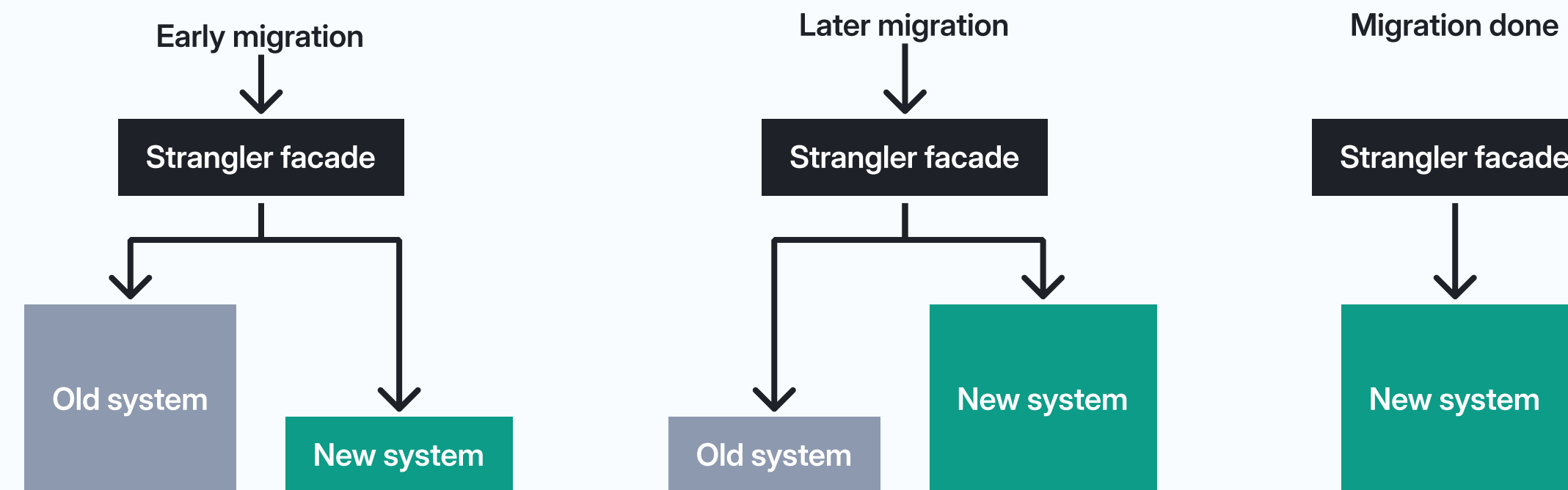
Вспомогательные домены



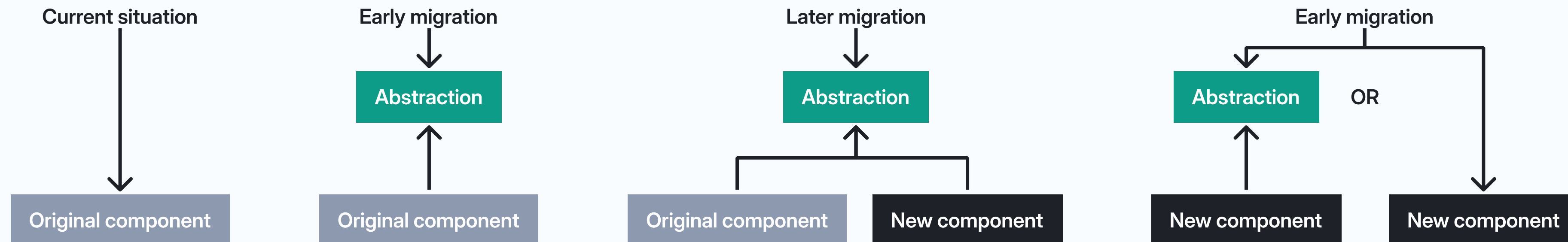
Домены общего назначения

# Примеры паттернов

## Strangler Fig (Душитель)



## Branch by abstraction

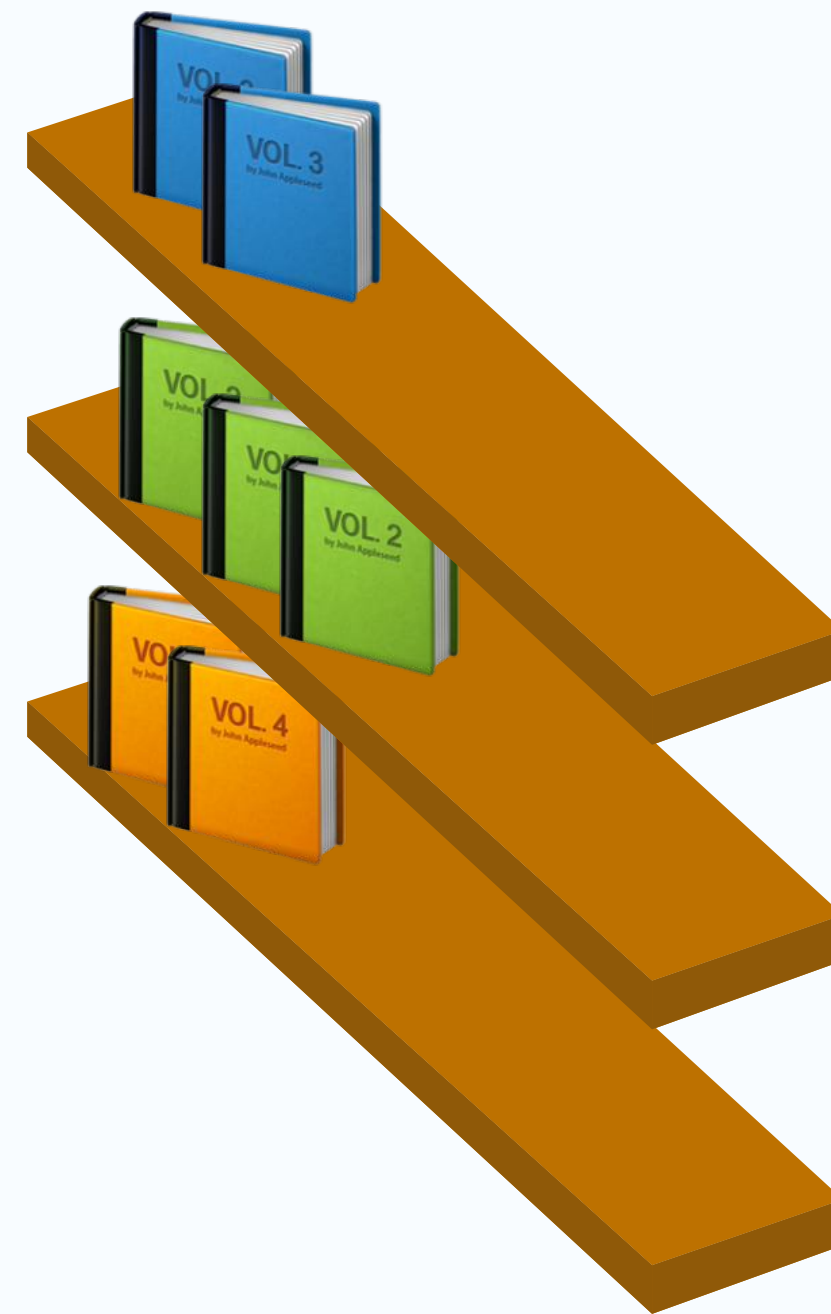




# Паттерн душитель



Старый монолит



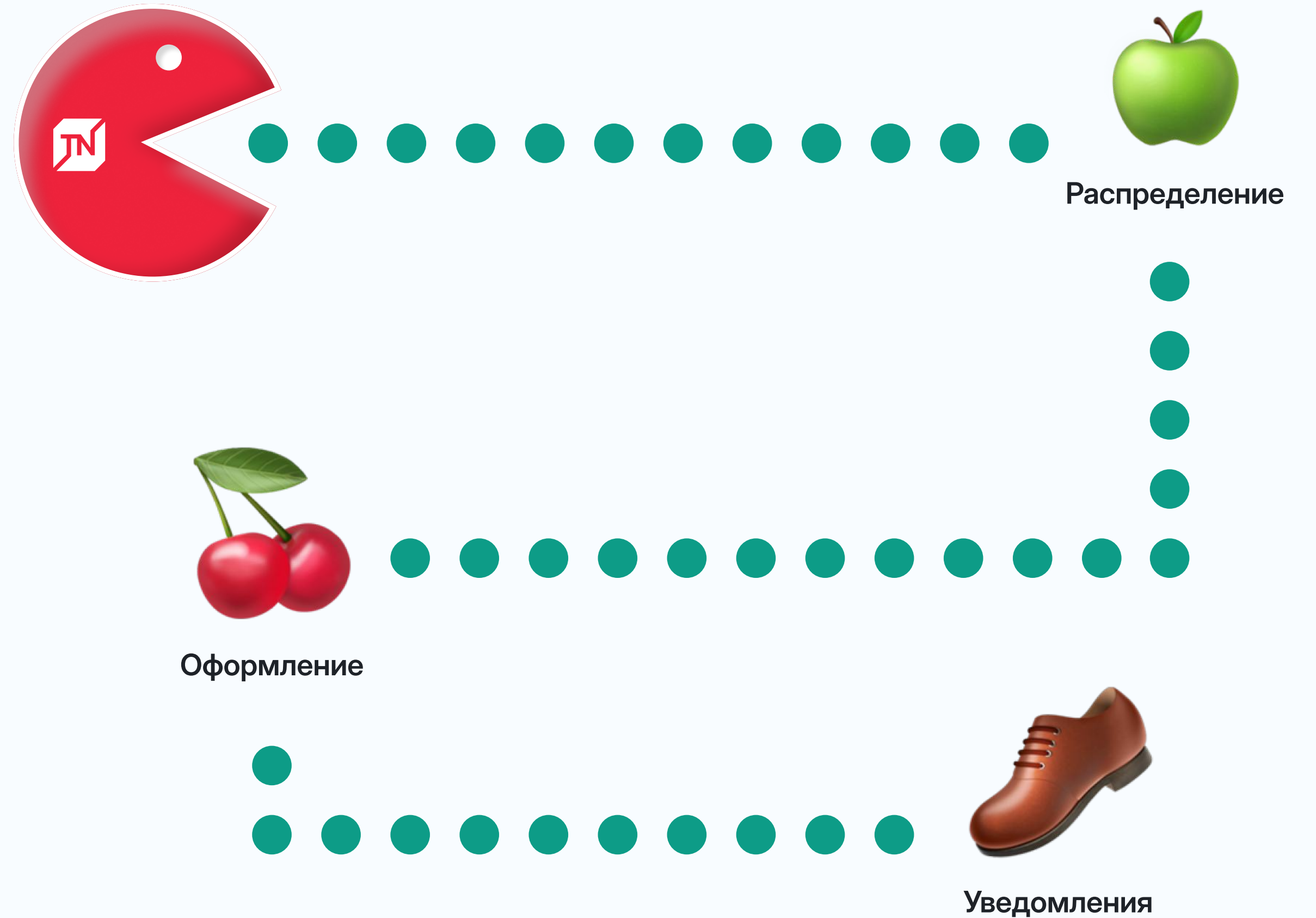
Переход к модульному монолиту



Модульный монолит

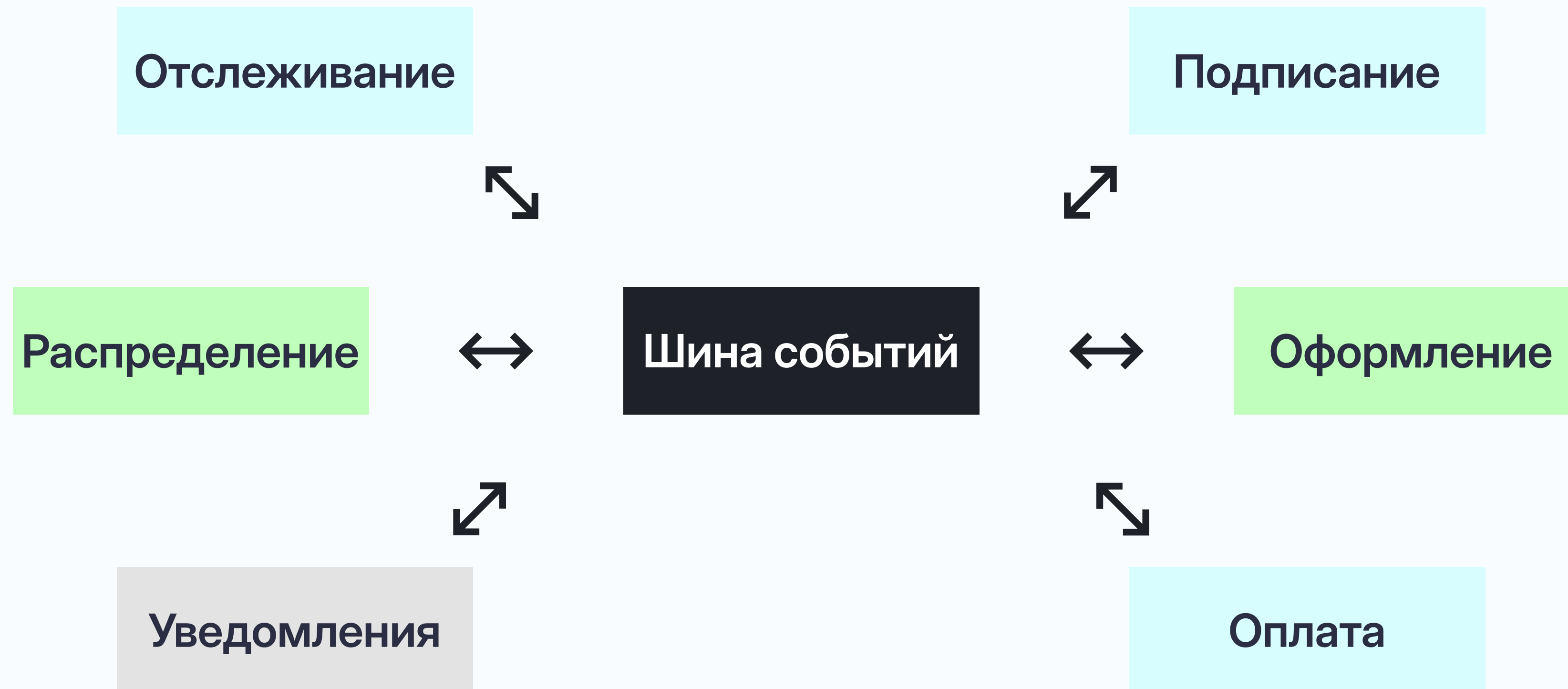
# Сохранение экспертизы

Смысловое ядро — это конкурентное преимущество компании, экспертиза по которому всегда должна оставаться внутри корпорации.





# Архитектурная схема



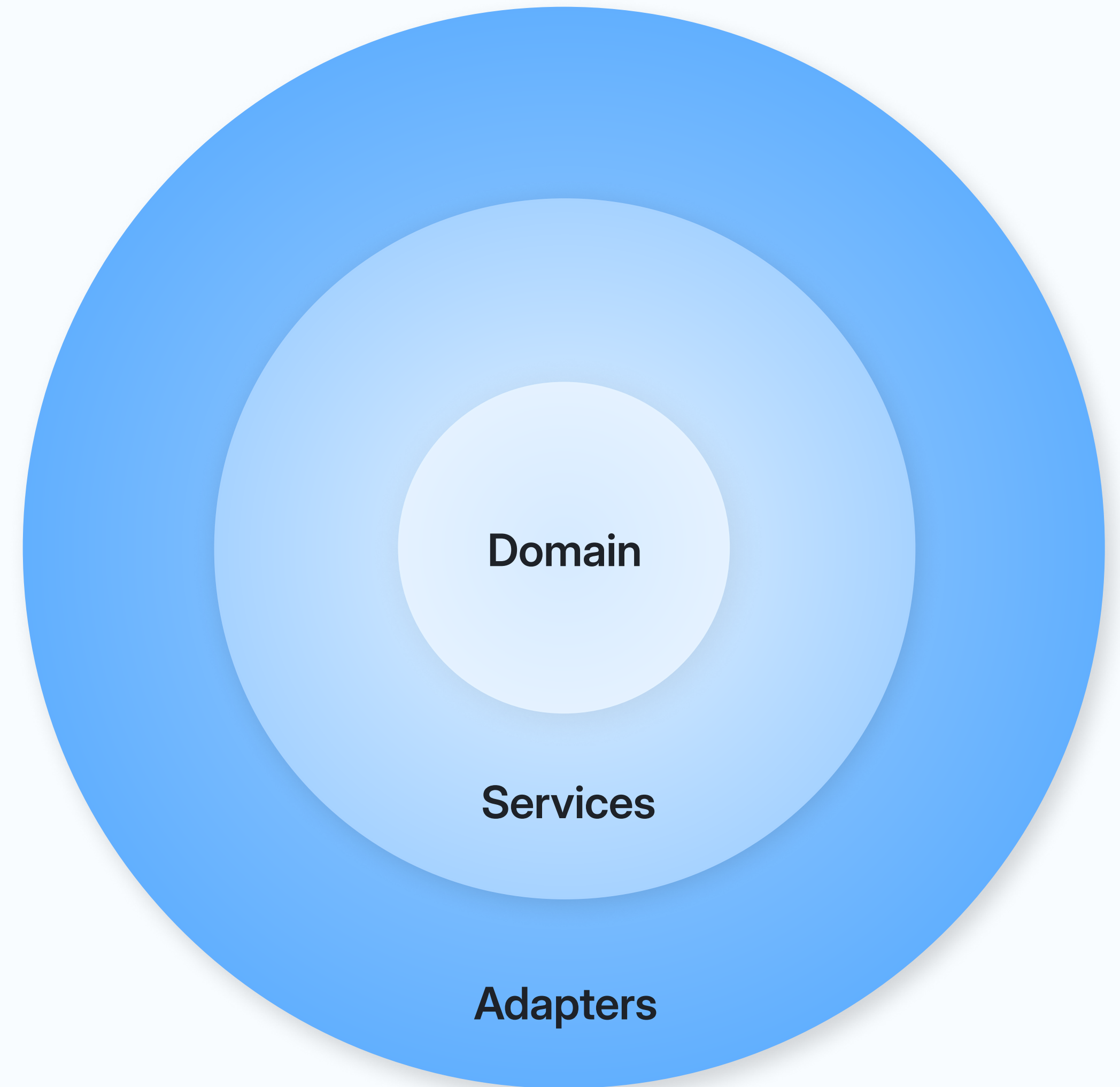
# Архитектура домена

Домен разделен на три слоя:

**Domain** — логика бизнес-сущности

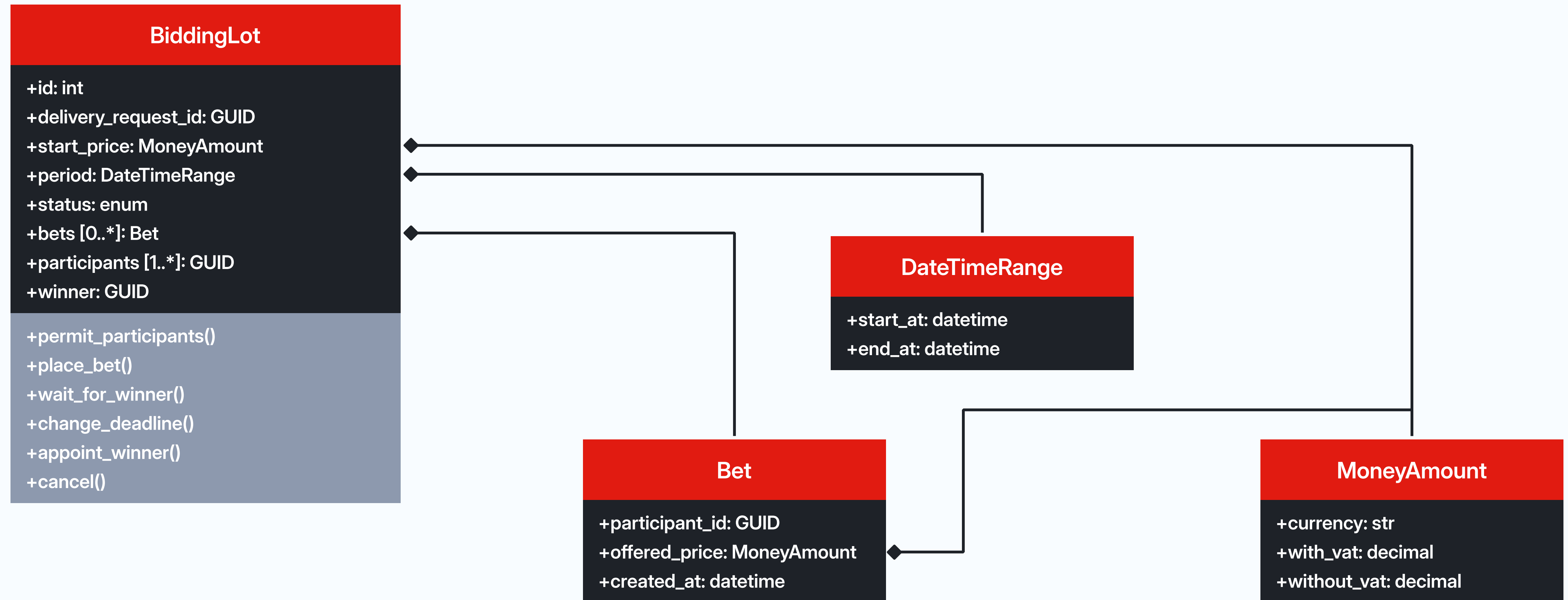
**Services** — логика приложения

**Adapters** — инфраструктура

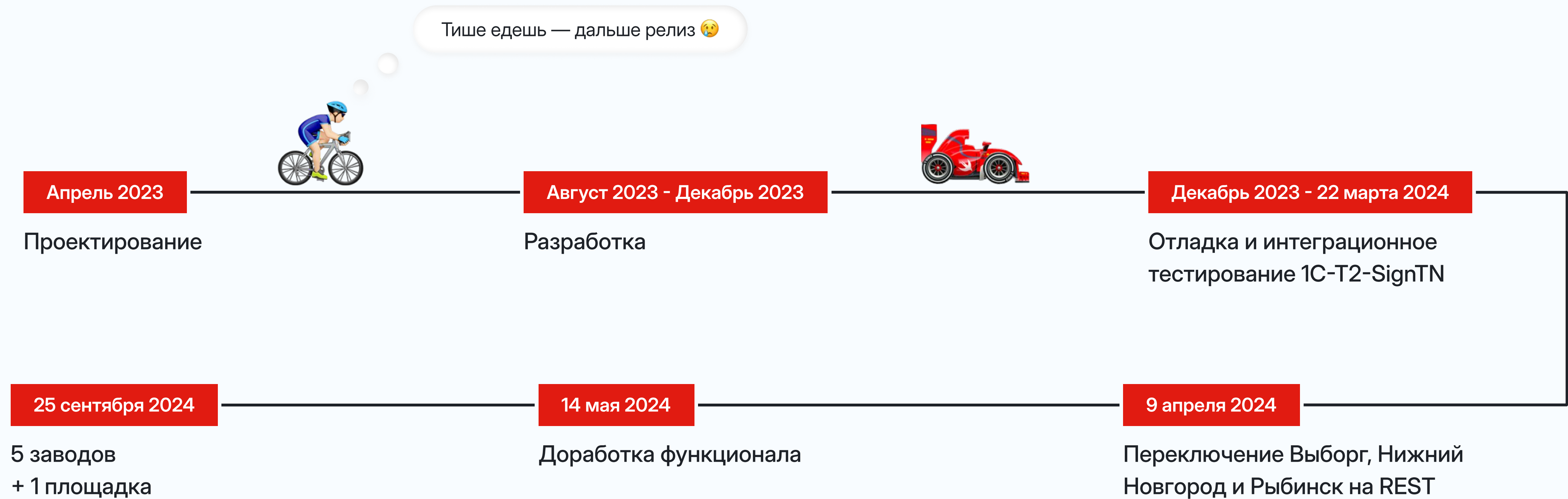




# Моделирование бизнес-сущностей



# История проекта



# Что мы получили?



Уменьшилось время  
разработки фич ЖД  
и контейнерных перевозок

**В 2 раза**



# Что мы получили?



Уменьшилось время  
разработки счетчиков  
в торгах и аукционах

**В 4 раза**

# Что получили от внедрения новой архитектуры и применения DDD:

Бизнес и разработка говорят на одном языке: термины бизнеса находят прямое отражение в программном коде

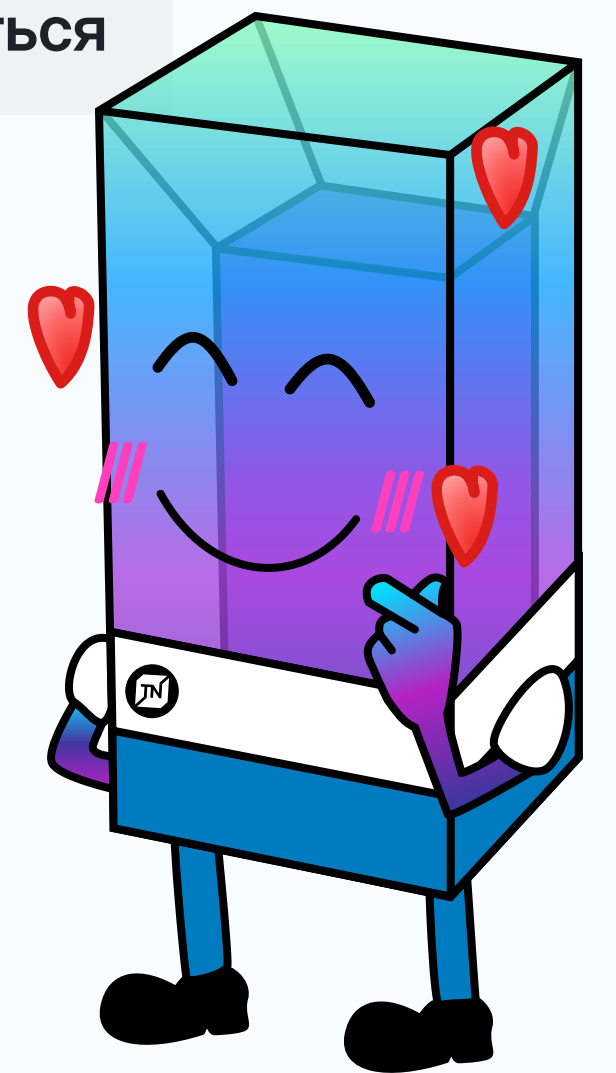
Гибкость в выборе языков программирования, баз данных и фреймворков

Снизилось время на онбординг

Ключевая экспертиза остаётся внутри компании

Ускорилась разработка

Разработка стала лучше параллелиться

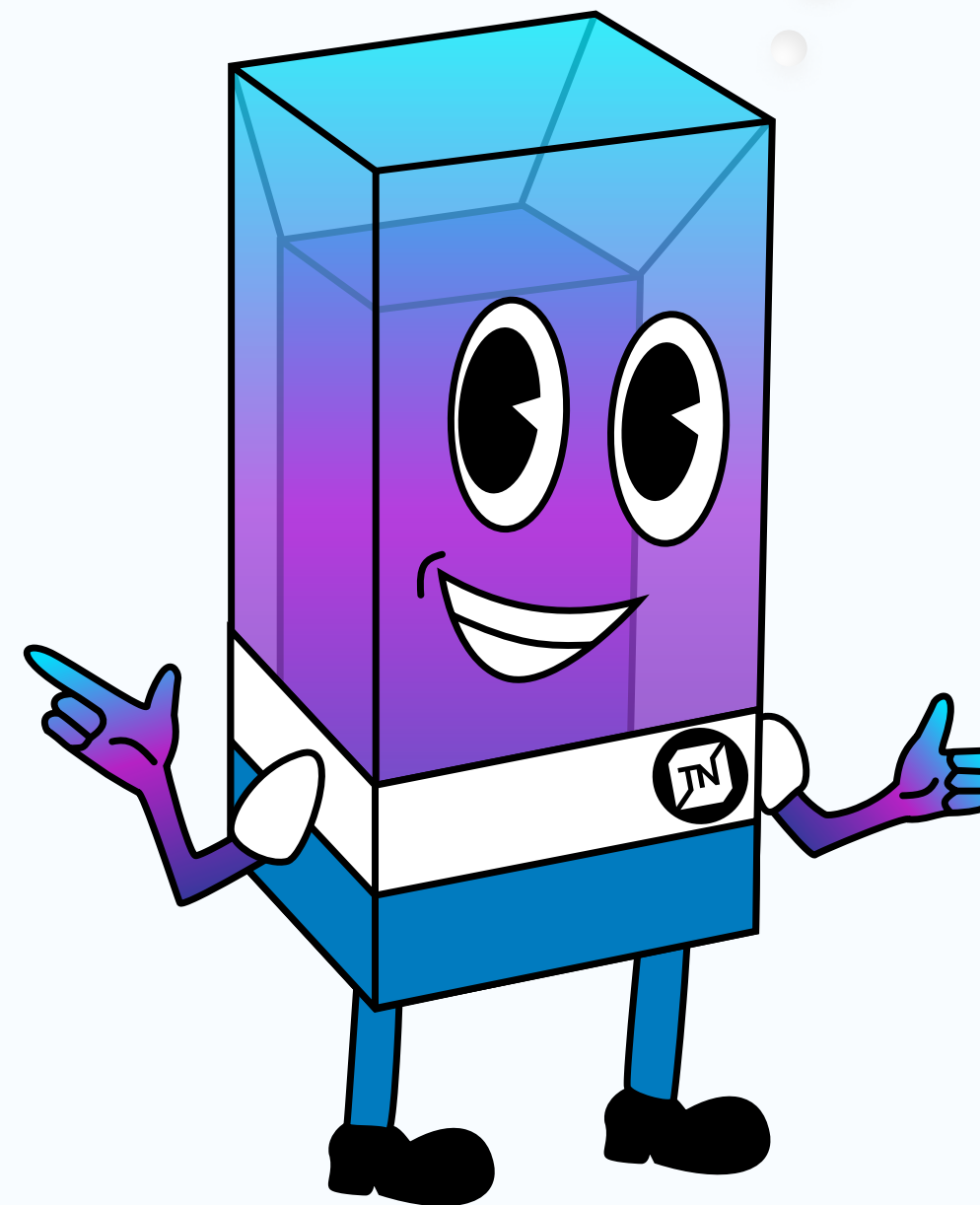




Статусно-событийная  
схема сущности



Практика Event Storming



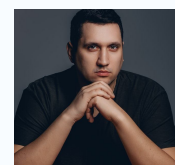
Не забудь скачать!



Список endpoints

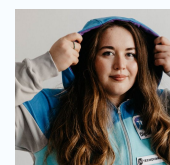


Архитектурная схема



**Стас Балахонов**

balahonov@tn.ru +7 996 721-00-33



**Альбина Бикбулатова**

bikbulatova.a@tn.ru +7 963 155-56-10



**ТЕХНОНИКОЛЬ**



**Digital**