Ссылка на схему: <https://drive.google.com/file/d/1E78zaIayIgXuX4inv4J4ODmcflfYFCzr/view?usp=sharing>

# Event Storming

Критерии для определения событий:

1. Фиксируем только события, про которые знает бизнес и в терминах которые знает бизнес
2. Правила задаем только в том случае, когда цепочка оказывается в подвешенном виде - то есть нет либо инициирующего актора, либо предыдущего события
3. Нотификация не показана на схеме - считаю ее технической.

Критерии для формирования контекста:

* В общем случае я бы контекст определял бы по принадлежности бизнес-пользователей - владельцев информации о событиях, операциях и т.д.(тут вижу есть некие постановщики процессов - по работе с заказами на услуги, комплектование заказов расходниками, финансовый сектор и т.д.). Но в данном случае такие роли неизвестны - и я их представил как виртуальных “собеседников”.
* Дополнительно бы на диаграмме Event Storming я бы добавил агрегаты ( тут в общем духе требований, сформулированных на уроке, агрегаты на схему не добавлял) и контекст дополнительно бы определялся бы агрегатами. Тут агрегаты также являются виртуальными.

Схема: <https://viewer.diagrams.net/?page-id=ijep7U4_XeQ0FfPpVQ-H&highlight=0000ff&edit=_blank&layers=1&nav=1#G1E78zaIayIgXuX4inv4J4ODmcflfYFCzr>

Картинка: <https://drive.google.com/file/d/1wmyiYBEinoummx2LedaTplrErMEMikWX/view?usp=sharing>

# Модель данных

Схема: <https://viewer.diagrams.net/?page-id=JReekw00ZZjvuHovSAet&highlight=0000ff&edit=_blank&layers=1&nav=1&page-id=JReekw00ZZjvuHovSAet#G1E78zaIayIgXuX4inv4J4ODmcflfYFCzr>

Картинка: <https://drive.google.com/file/d/1HzPQSmYyYc3ESs9BAOsle7bEpDs_uPtD/view?usp=sharing>

# Коммуникации системы

Схема: <https://viewer.diagrams.net/?page-id=yQwNVBoLQCFls6H9wJbg&highlight=0000ff&edit=_blank&layers=1&nav=1&page-id=yQwNVBoLQCFls6H9wJbg#G1E78zaIayIgXuX4inv4J4ODmcflfYFCzr>

Картинка: <https://drive.google.com/file/d/1wuaCg8tatSbZl8gFtUfV26aGlTWgdQ1M/view?usp=sharing>

# Выбор структуры системы

## Оценка

| **Критерий** | **Оценка критерия с точки зрения требований** | **Монолит** | **Микросервисы** |
| --- | --- | --- | --- |
| Time To Market | Так как это новый проект (который возможно еще и не взлетит), то хотелось бы быстрее начать работу чтобы подтвердить или опровергнуть гипотезы. | + | - |
| Производительность | В требованиях производительность указана достаточно низкая и нет отдельных требований на скорость обработки данных - поэтому считаем, что производительностью в данном случае можно пренебречь. | +  В данном случае монолит все равно быстрее за счет скорости коммуникаций и отсутствия требований к обеспечению консистентности распределенных данных. | -  Микросервисная архитектура могла бы выиграть, если для каких-то задач были бы заданы достаточно жесткие требования к производительности.  В этом случае выделение этого функционала в отдельные сервисы помогло реализовать их на подходящих для быстрой обработки языках, а также позволило исключить влияние смежных процессов (так как они даже с низким приоритетом использовали те же самые ресурсы).  Но в данном случае таких требований нет. |
| Связанность |  | +  Высокая связанность.  Для разных функциональных модулей можно использовать единую схему БД | -  Низкая.  Требуются дополнительные решения, по обеспечению консистентности распределенных данных |
| Масштабируемость | Планы есть, но они еще не подтверждены аналитикой | -  Масштабируется весь монолитный компонент. | +  Отдельные сервисы легко масштабируются. |
| Стоимость разработки | Так как это новый проект (который возможно еще и не взлетит), то хотелось бы иметь более дешевое решение | + | -  Более сложное решение.  Сервисов не так много, чтобы был эффект от типизации разработки (например, применения отдельных шаблонов сервисов)  Также более сложное тестирование, особенно в части сквозных межсервисных процессов |
| Стоимость внедрения |  | +  Инфраструктура нужна только для самого решения. | -  Дополнительно требуется инфраструктура для организации межсервисного взаимодействия |
| Отчуждаемость | Функционал со ставками достаточно сомнителен и “серый”.  Поэтому логично было бы его вынести отдельно. | -  В данном случае выделить эти функции будет достаточно сложно - надо будет программировать (хотя бы для удаления кода), что может повлиять на связанные задачи | +  В данном случае достаточно будет просто выключить и удалить соответствующие сервисы. |

## Решение

Я остановился на комбинированном варианте:

1. монолитное решение покрывает основные задачи, связанные с обработкой заказов на услуги (сам процессинг заказов, комплектование заказов расходниками, аккаунтинг и биллинг и оценкой качества)
2. отдельные сервисы для приема воркеров и тотализатора

Взаимодействие между компонентами - асинхронное.

Объединение функциональных модулей связанных обработкой заказов на услуги и связанных с этой обработкой задач комплектования заказов расходниками позволит:

1. Быстрее вывести в эксплуатацию функции, связанные с основной целью проекта
2. Сэкономить на разработке и эксплуатации (меньше потребности в инфраструктуре, проще тестировать межмодульное взаимодействие, меньше внешнего взаимодействия)

Вынесение отдельных сервисов позволит:

1. для тотализатора: проще будет отказаться от данного функционала
2. для приема воркеров: не будет оказываться влияние на основные функции (в принципе, может даже будет можно использовать какие-то готовые решения, так как реализуется достаточно типовой процесс найма)

Минусы решения:

1. несмотря на монолитность все равно придется делать передачу данных между компонентами (но тут возможна реализация более простых способов асинхронного взаимодействия - так как это процессы с пониженным уровнем влияния на цели проекта и следовательно - с пониженными требованиями к надежности канала обмена)

# Проблемы и слабые места

| **Проблема и слабое место** | **Комментарии** |
| --- | --- |
| Расширение клиентской базы | Если услуги будут предоставляться не только котам-тестировщикам HCB, то в этом случае потребуется отдельный функциональный модуль для работы с клиентами. Данный функционал является побочным относительно основной задачи системы, но его реализация в монолите будет влиять на основные функции - хотя там изменений может не произойти. |
| Изменение финансовых алгоритмов | Потребует изменение логики связанного процесса, но так как разработка будет в монолите, то возможно влияние и на основные функции. |