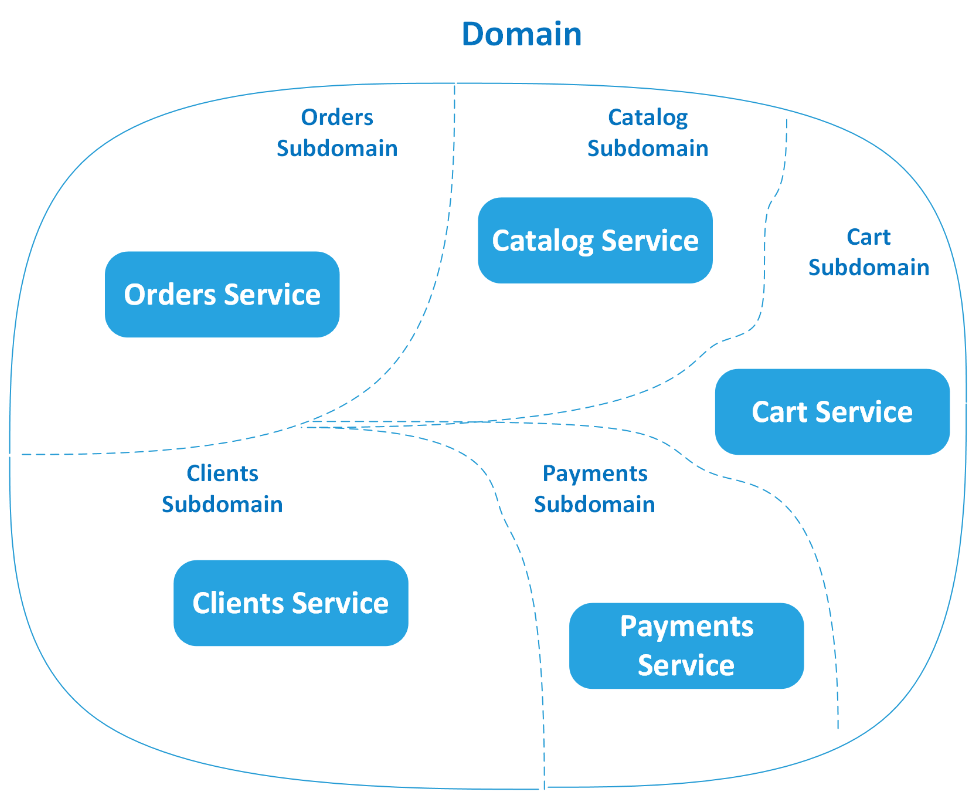
**Паттерны микросервисной разработки**

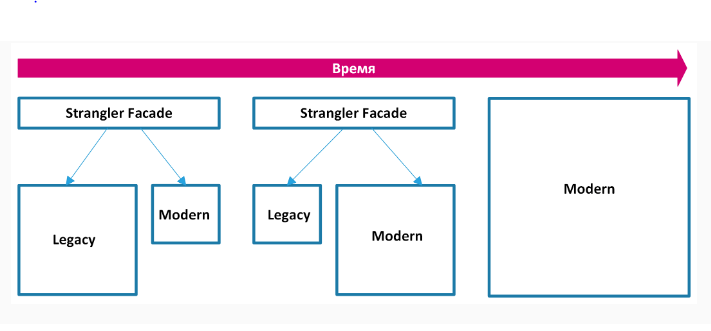
1. **Decompose by Business Capability: разбиение по бизнес-возможностям**



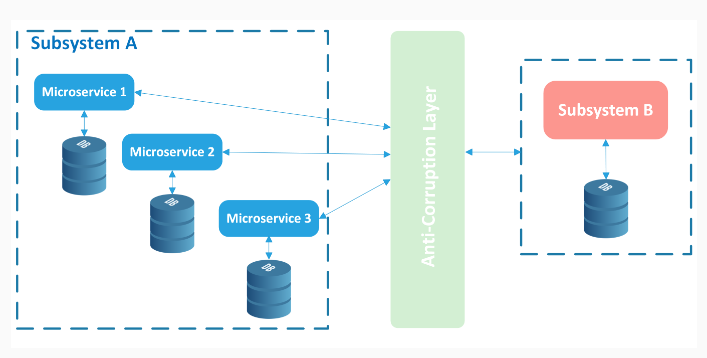
1. **Decompose by Subdomain: разбиение по поддоменам,** Domain-Driven Design (DDD)



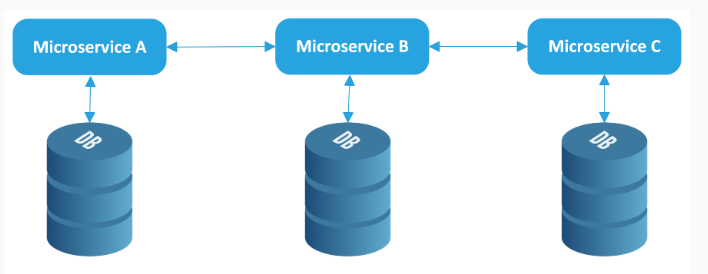
1. **Strangler: душитель, паттерн рефакторинга**



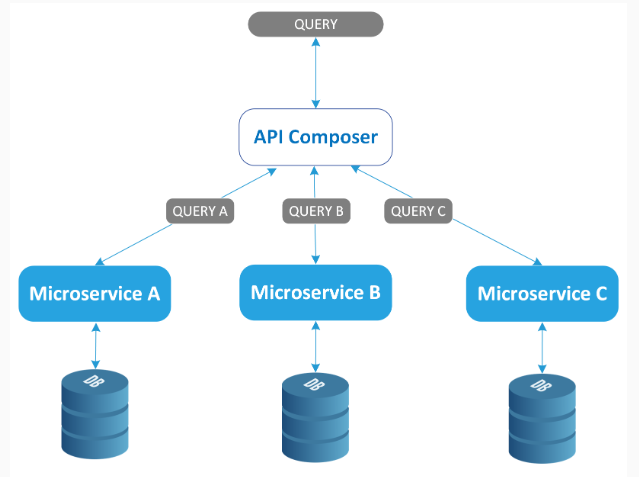
1. **Anti-Corruption Layer: уровень защиты от повреждений, паттерн рефакторинга, Anti-Corruption Layer - отдельный сервис для взаимодействия с Legacy-подсистемами (Subsystem B).**



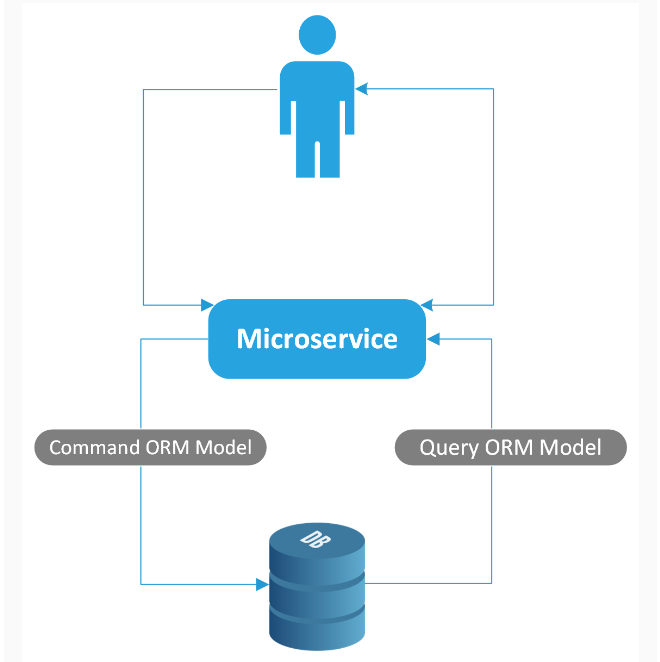
1. **Database per Service**: база данных на сервис, противопоставляется антипаттерну Shared Database, проблема ACID.

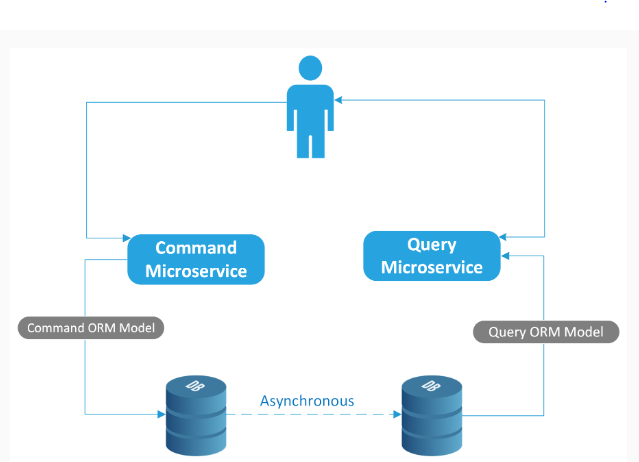


1. **API Composition: API-композиция, создание иллюзии одной базы данных**

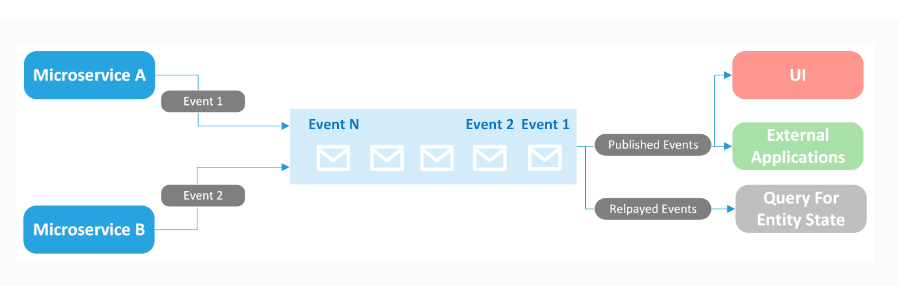


1. **Command Query Responsibility(CQRS): разделение команд и запросов, простая и расширенная формы**

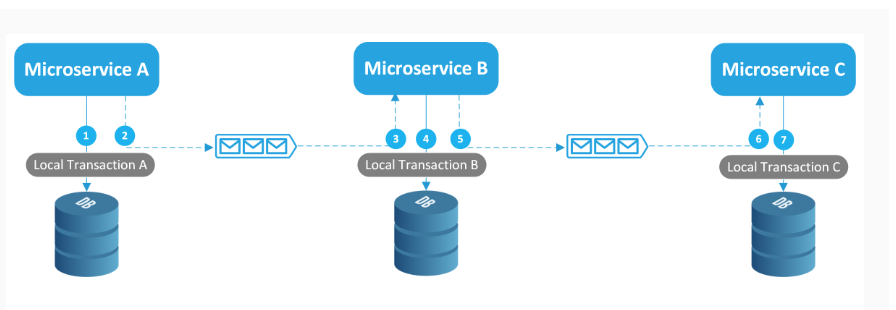




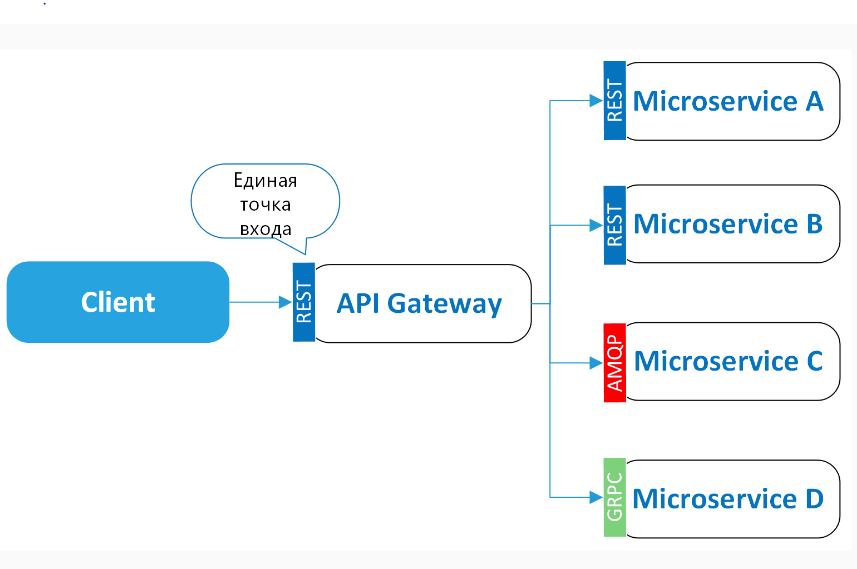
1. **Event Sourcing: поиск событий, изменение в данных приложений (UI, External Application, Entity State) осуществляется только через очередь событий, для высокопроизводительных, масштабируемых приложений, CQRS**



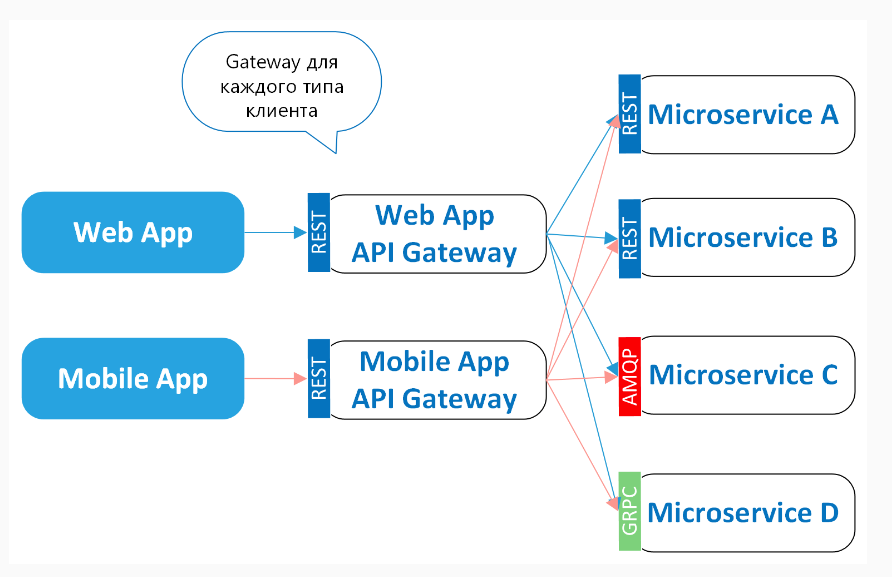
1. **Saga: Сага, для распределенной транзакции, изменение по очереди в локальных базах данных, сообщение в другой сервис, выполнение изменение в другой базе данных, ошибка – компенсирующие операции (откат). Хореография – децентрализованная координация, микросервис ждет-принимает сообщение (событие), выполняет (или нет) соответствующее действие. Оркестровка – централизованная координация, центральный компонент (сервис) отсылает сообщение другим сервисам команды-сообщения, сервисы выполняет соответствующее действие.**



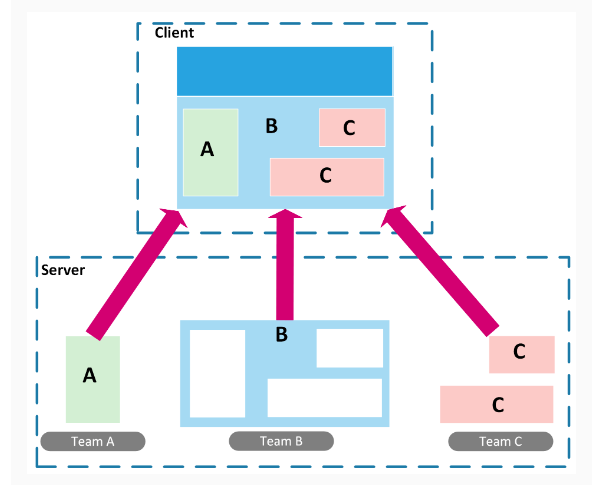
1. **API Gateway: API-шлюз, единая точка входа, виды: Routing – обратный Proxy (перенаправляет запросы клиента), Aggregation – разветвление одного запроса на несколько, Offloading – общая задача (аутентификация, авторизация, журналирование,…).**



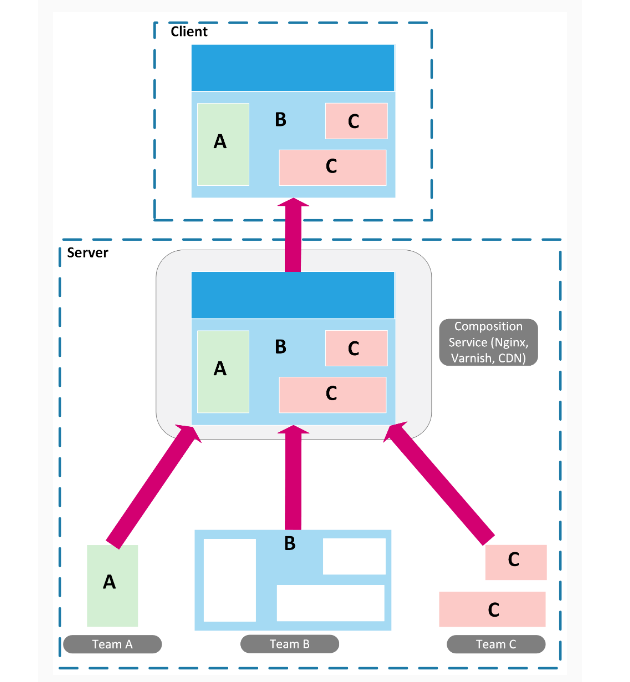
1. **Backend for Frontends: BFF**



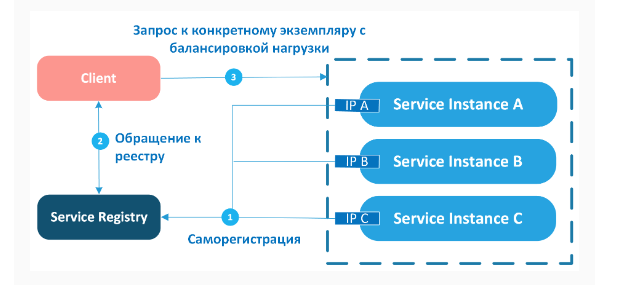
1. **Client-Side UI Composition:** сборка пользовательского интерфейса на стороне клиента, каждый UI-фрагмент – микросервис (React.js, Angular.js).



1. **Server-Side Page Fragment Composition:** сборка фрагментов страницы на стороне сервера



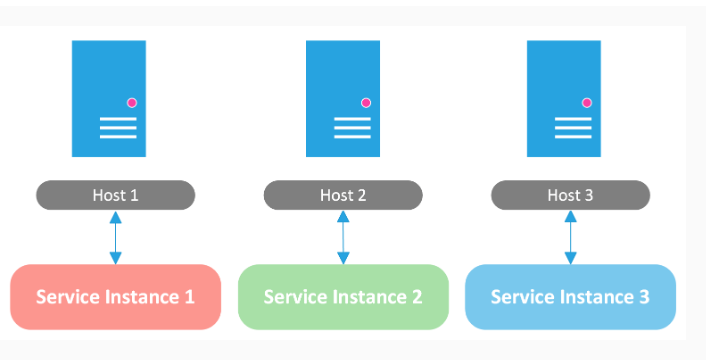
1. **Client-Side Service Discovery:** обнаружение сервисов на стороне клиента, Registry.
2. сервис регистри­рует свое сетевое местоположение в реестре; может предоставить URL-адрес для проверки своей работоспособности (Health Check);
3. клиент обращается к реестру, чтобы получить список сервисов; клиент может кэшировать список;
4. клиент использует алгоритм балансировки нагрузки.



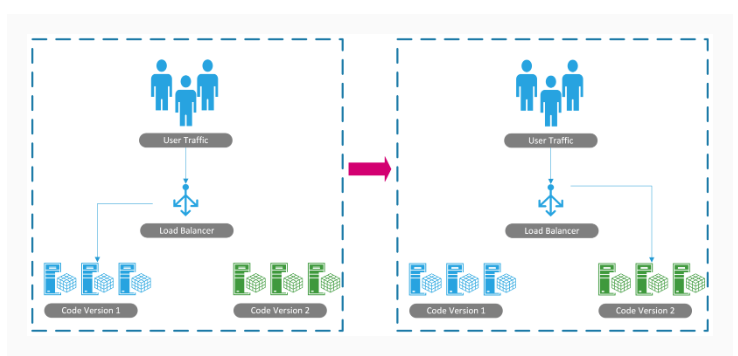
1. **Server-Side Service Discovery:** обнаружение сервисов на стороне сервера, Router, Registry, Registrator.
2. регистратор(часть платформы развертывания) прописывает все сервисы в реестре сервисов (DNS-имя, виртуальный IP-адрес);
3. по DNS-имени клиент обращается в маршрутизатор (часть платформы развертывания);
4. маршрутизатор обращается к реестру для получения сетевого расположения сервиса;
5. маршрутизатор балансирует нагрузку выбором конкретного экземпляра сервиса для отправки запроса.



1. **Service Instance Per Host: экземпляр сервиса на хост.**



1. **Blue-Green Deployment**: сине-зеленое развёртывание, синие сервисы (legacy), зеленые (modern), для возможности отката на legacy.

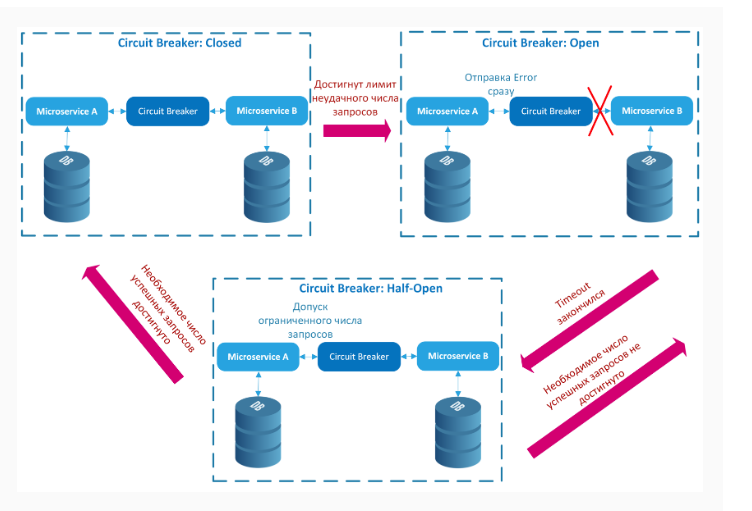


1. **Circuit Beaker**: автоматический выключатель- промежуточный сервис, для снижения непроизводительной нагрузки, три режима:

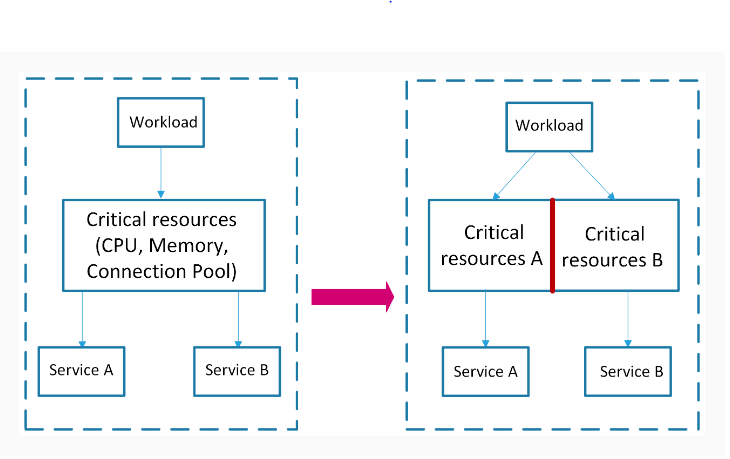
**Closed:**передача запросов между сервисами, подсчет сбоев; если число сбоев превышает порог, выключатель переводится в состояние Open;

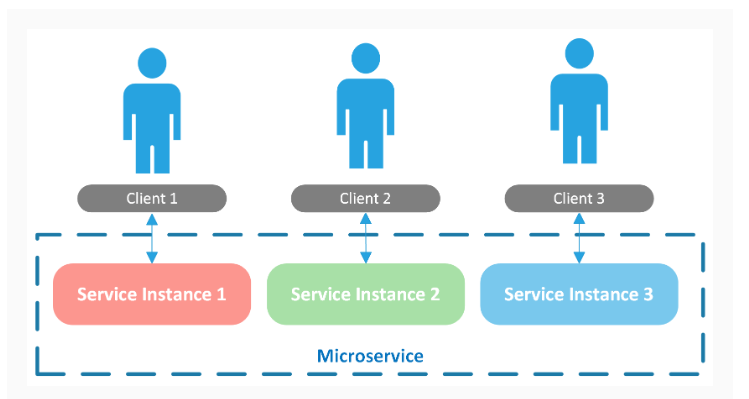
**Open:**запросы немедленно возвращаются с ошибкой; через тайм-аут выключатель переводится в состояние Half-Open.

**Half-Open:**выключатель пропускает ограниченное количество запросов подсчитывает число успешных запросов; если необходимое количество достигнуто, выключатель переходит в состояние Closed, если нет — возвращается в Open.

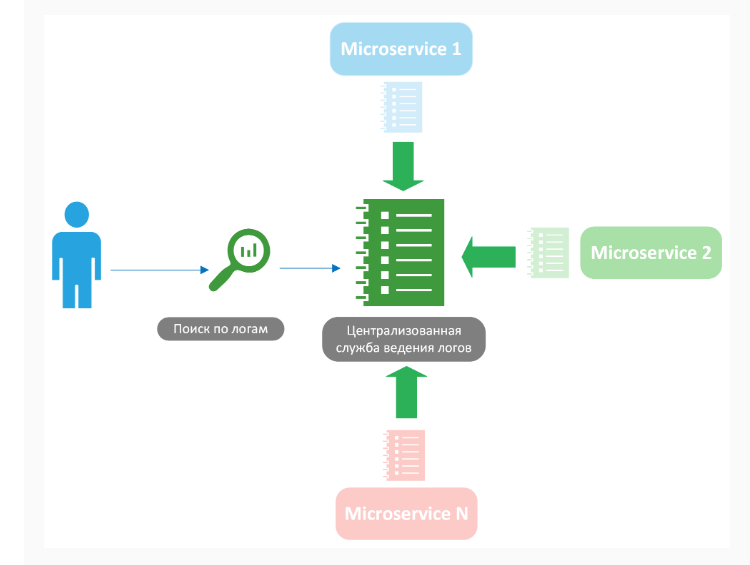


1. **Bulkhead:** переборка, пулы микросервисов разделяются изолирующими «переборками», позволяющим отключить пул с случае сбоя, ради работоспособности других, второй вариант: каждому клиенту свой экземпляр сервиса.

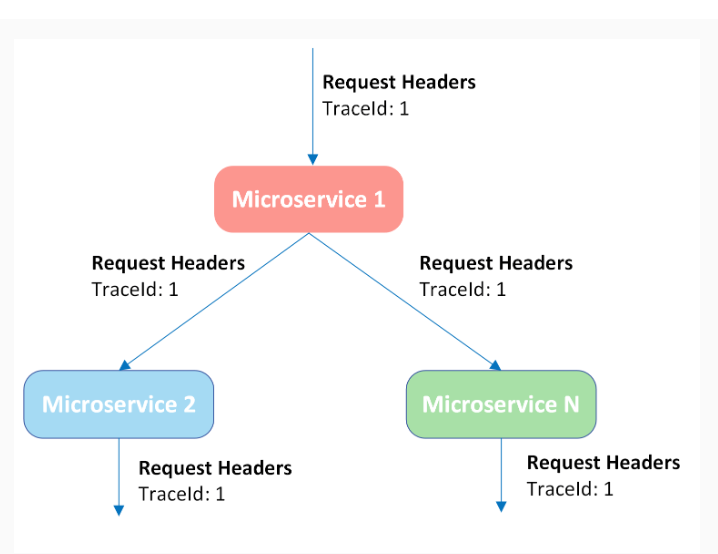




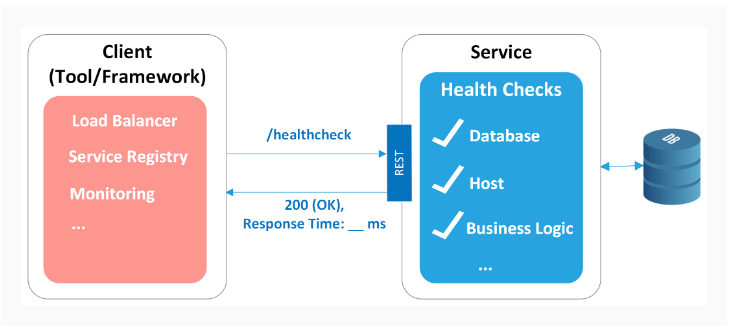
1. **Log Aggregation: агрегация журналов, централизованный журнал для всех сервисов.**



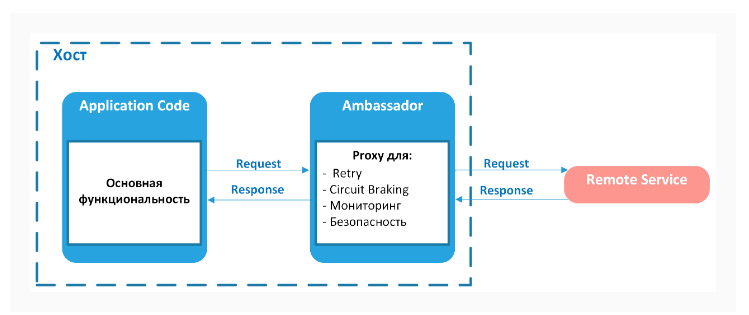
1. **Distributed Tracing: распределенная трассировка, уникальный номер запросов, проходящий через все сервисы.**



1. **Health Check: проверка здоровья**



1. **Ambassador: посредник, объединение нескольких функций**



1. **https://mcs.mail.ru/blog/26-osnovnyh-patternov-mikroservisnoj-razrabotki**