MDA. Sync & Async 2021

Message-driven architecture

Что будет на уроке

1. Вспомним понятия синхронных и асинхронных коммуникаций;
2. Поймем в каких случаях необходимо использовать одно из них;
3. Познакомимся с SOA и MSA.

ТЕМА

Синхронное и асинхронное взаимодействия

01

GeekBrains

Что такое API?
Синхронное API

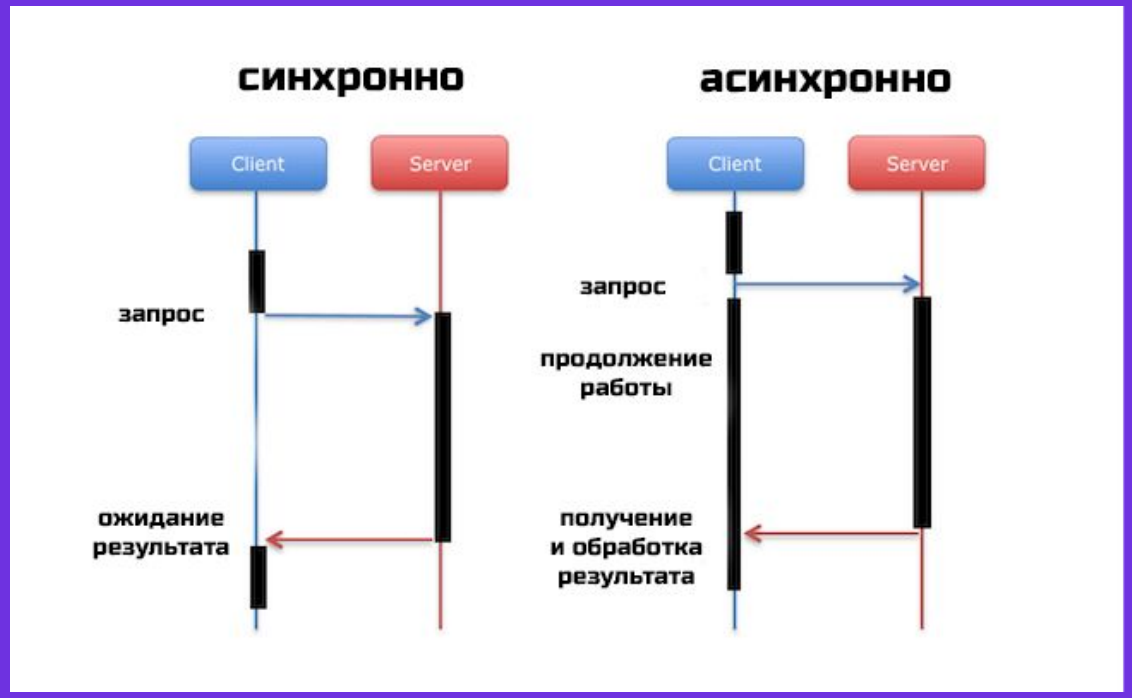
Асинхронное API

Что такое API ?

API (Application programming interface) - это контракт, который предоставляет программа.

“Ко мне можно обращаться вот так, тогда я тебя сделаю вот это и вот так”

Синхронный и асинхронный API



Синхронный API - мы делаем запрос и ожидаем результата, ничего не делая в это время. Например, телефонный звонок.

Асинхронный API - делаем запрос, продолжаем работу. Получаем уведомление об окончании работы и обрабатываем результат. Например, вскипятить чайник.

ТЕМА

Какой стиль взаимодействия выбрать ?

02

GeekBrains

Два уровня выбора

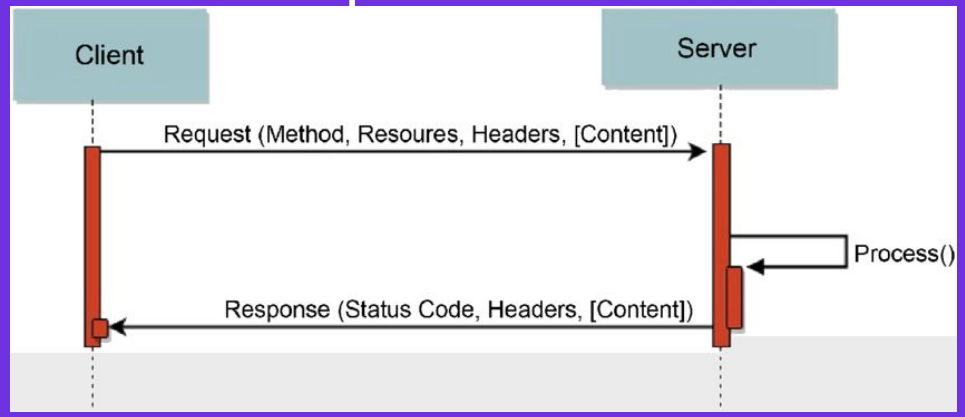
Взаимодействие “один ко многим”

Взаимодействие “один к одному”

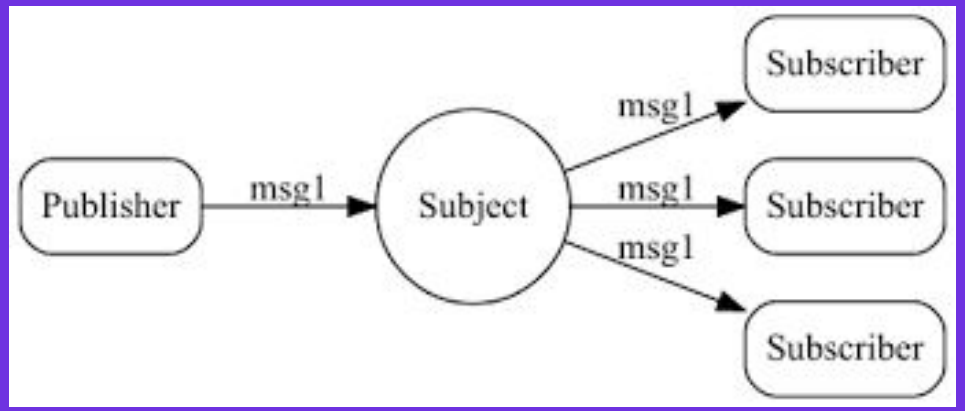
Два уровня выбора

Первый уровень

Один к одному - каждый клиентский запрос обрабатывается ровно одним обработчиком



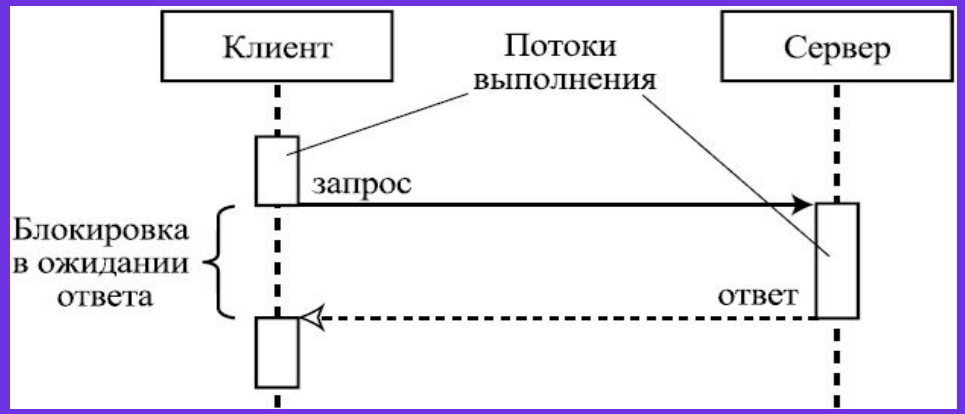
Один ко многим - каждый запрос обрабатывается несколькими обработчиками



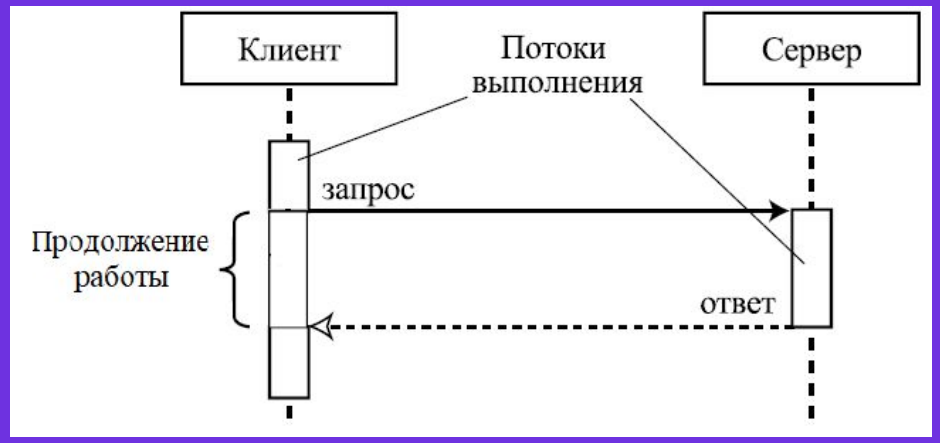
Два уровня выбора

Второй уровень

Синхронное - клиент ожидает своевременный ответ и может даже заблокироваться ;



Асинхронное - клиент не блокируется, ответ может быть отправлен не сразу



Тема

SOA и MSA

03

GeekBrains

Что означают аббревиатуры SOA и MSA

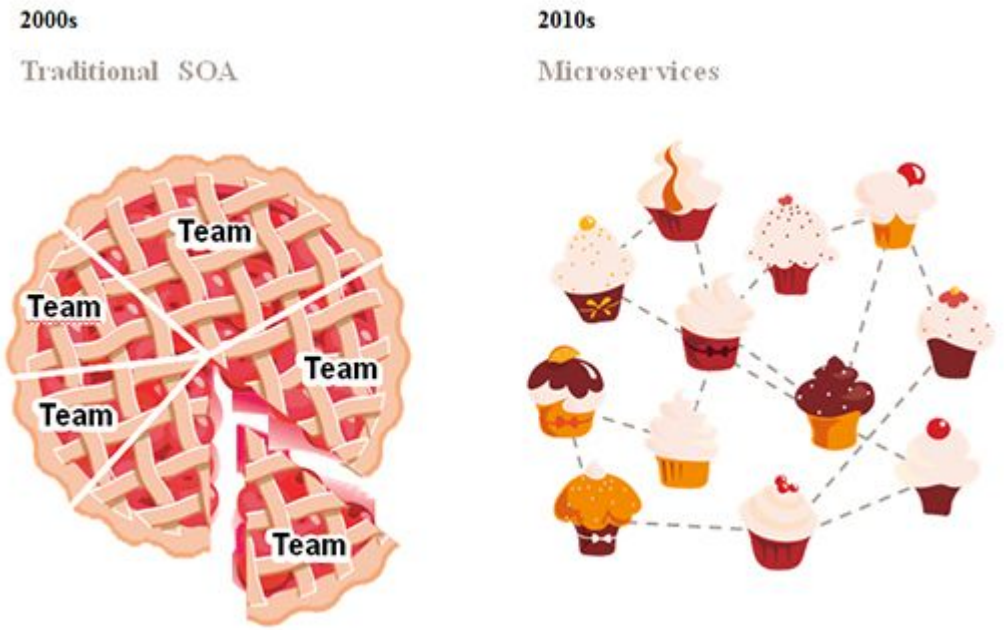
Подробнее о SOA

Подробнее о MSA

Значение аббревиатур SOA и MSA

SOA - service-oriented architecture, в основе подхода лежат несколько основных идей – переиспользование сервисов и корпоративная шина. Где сервисы это отдельный приложения, поддерживаемые отдельными командами, для корпоративной шины также существует команда поддержки и разработки.

MSA - microservices architecture, архитектура основанная на большом количестве сервисов. Логика одного сервиса должна умещаться в голове “одного” человека. Как правильно работает большое количество небольших команд, которые поддерживают сервисы. Взаимодействие происходит посредством данных через брокер сообщений без логики..



Подробнее о SOA

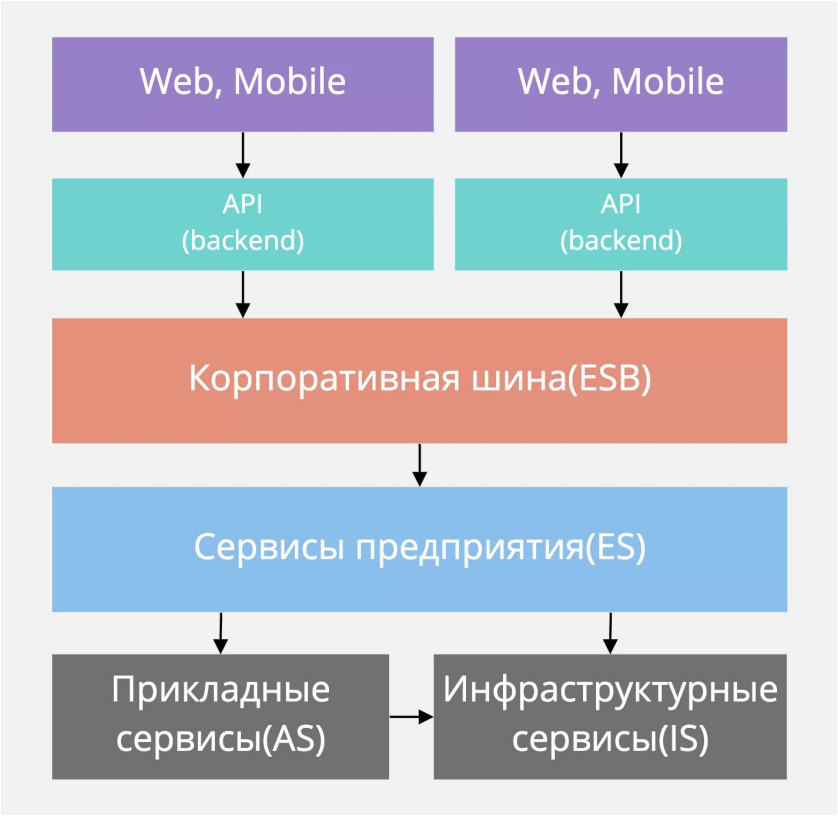
Шина(ESB): в случае взаимодействия сложных событий действует как посредник и управляет различными рутинными операциями, такими как передача сообщений и координация вызовов.

Инфраструктурные сервисы (infrastructure services): группа легко переиспользуемых сервисов, таких как аутентификация/авторизация, отправка смс и прочее.

Прикладные сервисы (application services): не могут быть переиспользованны под разные задачи, так как ограничены определенным прикладным контекстом, но их можно встраивать в более высокоуровневые сервисы.

Сервисы предприятия (Enterprise services): эти сервисы отвечают за реализацию крупных частей бизнес процессов компании, они потребляют более низкоуровневые сервисы.

API: по сути это бэкенды, предоставляющие API, доступное в интернет, для сайтов и мобильных приложений компании. Они взаимодействуют с ESB и раскрывают функциональность для конечных потребителей.

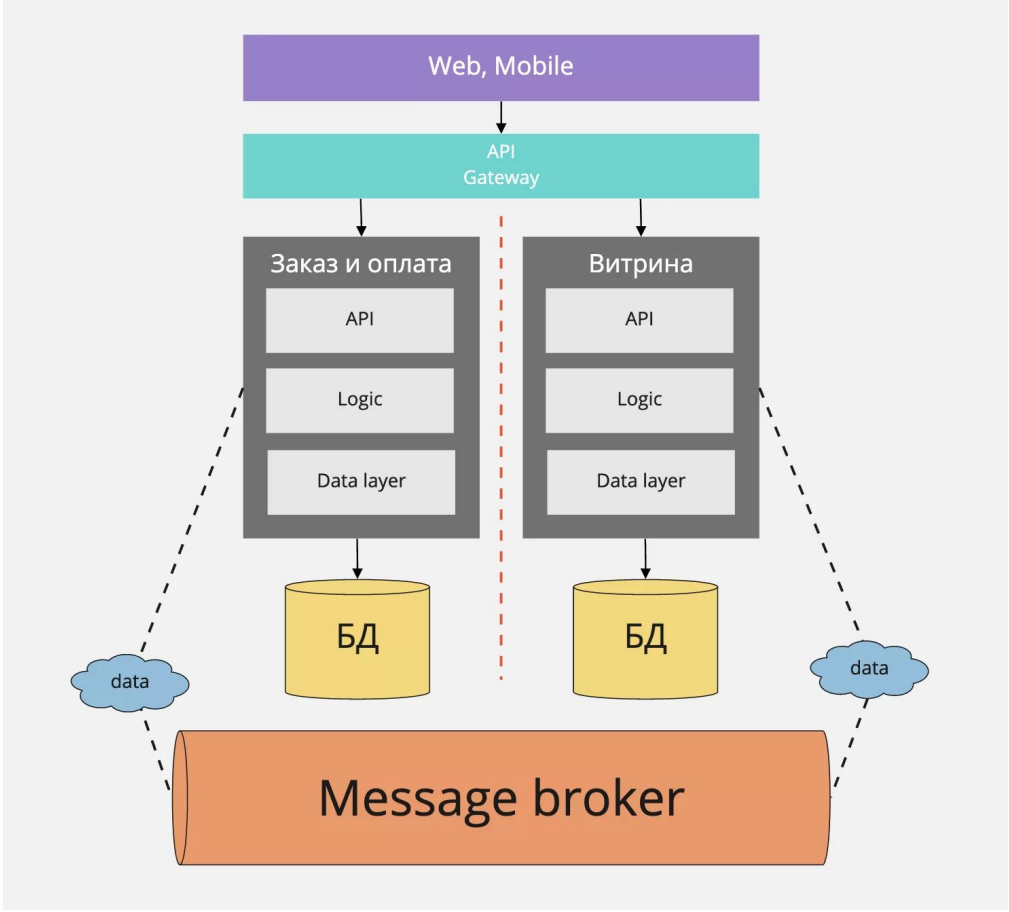


Подробнее о MSA

В отличие от SOA каждый сервис обладает всеми необходимыми для функционирования частями – имеет свою собственную базу данных и существует как независимый процесс. Такая архитектура делает каждый сервис физически разделенным, самодостаточным, что ведет с технической точки зрения к архитектуре без разделения ресурсов.

Сервисы раскрываются для потребителей также через слой API, но его стараются проектировать с полным отсутствием какой-либо логики. Это фактически просто проксирование API сервисов во вне.

Взаимодействие между сервисами сводится к обмену данными, используя брокер сообщений. Именно к обмену данными, а не вызову методов из других сервисов.



Домашнее задание

- Воспроизвести приложение по бронированию столика в ресторане из методического указания;
- Создать возможность снять бронь стола в ресторане синхронно и асинхронно, используя для этого номер забронированного столика;
- Выделить логику для отправки уведомления в отдельный класс, он будет отвечать за все вопросы связанные с коммуникациями с клиентами, добавить задержку (они будет имитировать создание сообщения) и сделать вызов уведомления асинхронным;
- Добавить автоматическое “снятие брони”. Например, раз в 20 секунд при наличии забронированных мест - бронь должна сниматься. Асинхронно, независимо от ввода-вывода. Подсказка: можно использовать таймер.
- (*) Добавить синхронизацию бронирований для множественных асинхронных вызовов и синхронного, это значит, что бронируя столики не дожидаясь предыдущих ответов мы должны получать последовательный результат.

Спасибо!

Каждый день вы становитесь лучше :)