Tuesday, May 17, 2022 11:02 AM

Продолжение про SOA.

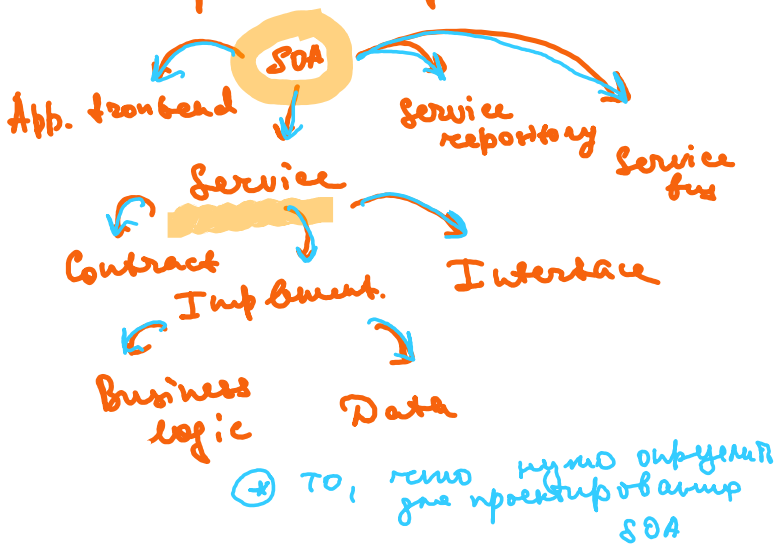
SOA – сервис-ориентированное приложение

≠ сетевое

SOA – это система, структурными эл-тами которой являются сервисы (но четкое опред. сложно найти)

Свойства: компоненты имеют стандартизированный интерфейс

Есть четкие правила построения и протокол взаим-вия.



* то, что нужно определить для проектирования SOA

SOA и микросервисы

разновидность SOA

(либо ≈ альтернатива)

* Раньше были протоколы SOAP, стандарт CORBA от OMG

что изменилось?

– Стали популярными легковесные процессы (Agile, Scrum)

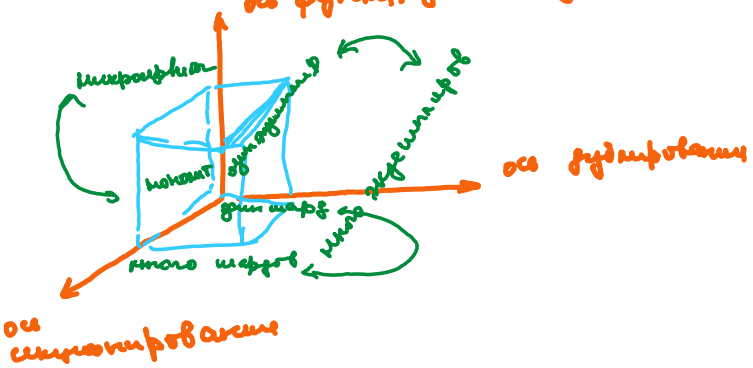
=> микро-команды

микро-сервисы

Главная проблема, которую решают с помощью микро-сервисной архитектуры, — несоответствие устройства программной системы и организации команд компании-разработчика.

- Ещё одна цель: улучшить нефункциональные характеристики.
  - надёжность
  - масштабируемость !
  - сложность ??????

Модель масштабирования



- Ещё одна причина для М.-С.? убегать "заразной" лицензией GPL.

* т.е. даже если 1 сервис под GPL, то другие необязательно.

Сервис и DevOps

Для SOA процесс развертывания и доставки НЕРАЗРЫВНО с процессом разработки.

Сервис - небольшое, сильно сконченное приложение, которое предоставляет (в идеале одну) узкоспециальную услугу

SOA	Микросервисы
Интеграция	

Взаим-е	Удалённые каналы серв. шины (SOAP-прокси)	Прямые каналы протокол типа REST
Данные	Общие БД, глобальная модель	БД для каждого сервиса
Типовой сервис	Крупный монолит	Небольшой монолит

про микросервисы

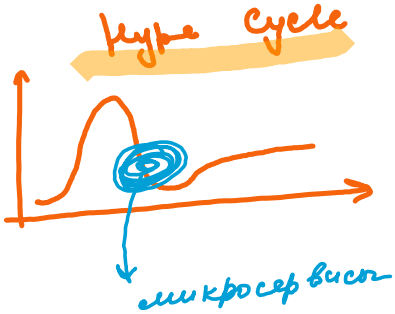
+ сервисы проще
+ сервисы легко заменяются и обновляются
+ их можно разрабатывать независимо
+ их можно масштабировать независимо
+ .. разрабатывать независимо
+ сервисы лучше сочетают технологии
+ упрощается внедрение новых технологий
+ удобно разрабатывать с использованием легковесных методологий

– сложность не исчезает, а переходит на коммуникационный уровень
– сложнее тестировать сервисы, которые работают асинхронно
– сложнее отлаживать (асинхронная и распределенная)
– сложнее декомпозировать
– сложнее развертывание, т.е. сохранить согласованность системы. Нужны инструменты авто-развертывания (Docker, Kubernetes)
– Опасность разработать функционал нескольких сервисов сразу.
– легко получить проблемы с производительностью
– ... с балансировкой нагрузки
– оверхед из-за преобразования форматов данных
– „Доверяем данные" сторонним сервисам → безопасность, приватность
– Проблемы с отказоустойчивостью.

3. Это развитие UNIX подхода pipe-and-filter

4. активно исп-ся контейнеризация

Hype cycle (Gartner)



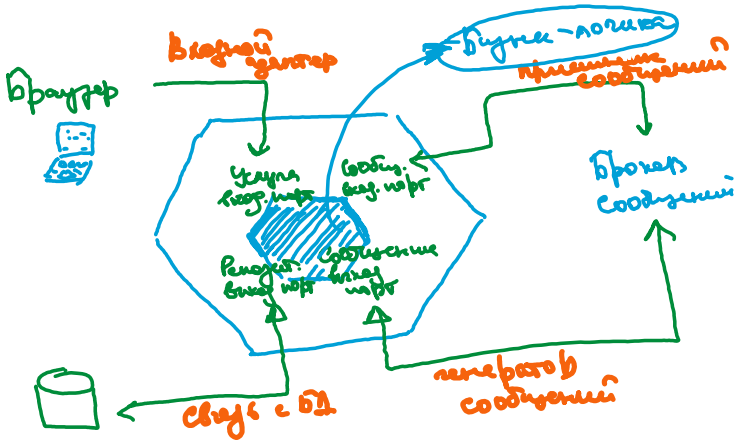
Как проектировать микросервисные системы?

- это всего лишь архитектурный стиль

+ представление прецедентов
+ логическое
+ процессное
+ реализации
- - - - -

это всё тоже применимо

1. отдельный сервис – маленький, но небольшой



Блоки, из которых собирается приложение.