РАЗПРЕДЕЛЕНИ ПРИЛОЖЕНИЯ

Павел Кюркчиев
Ас. към ПУ „Паисий Хилендарски“
@pkyurkchiev

РАЗПРЕДЕЛЕНИ АРХИТЕКТУРИ MICROSERVICES

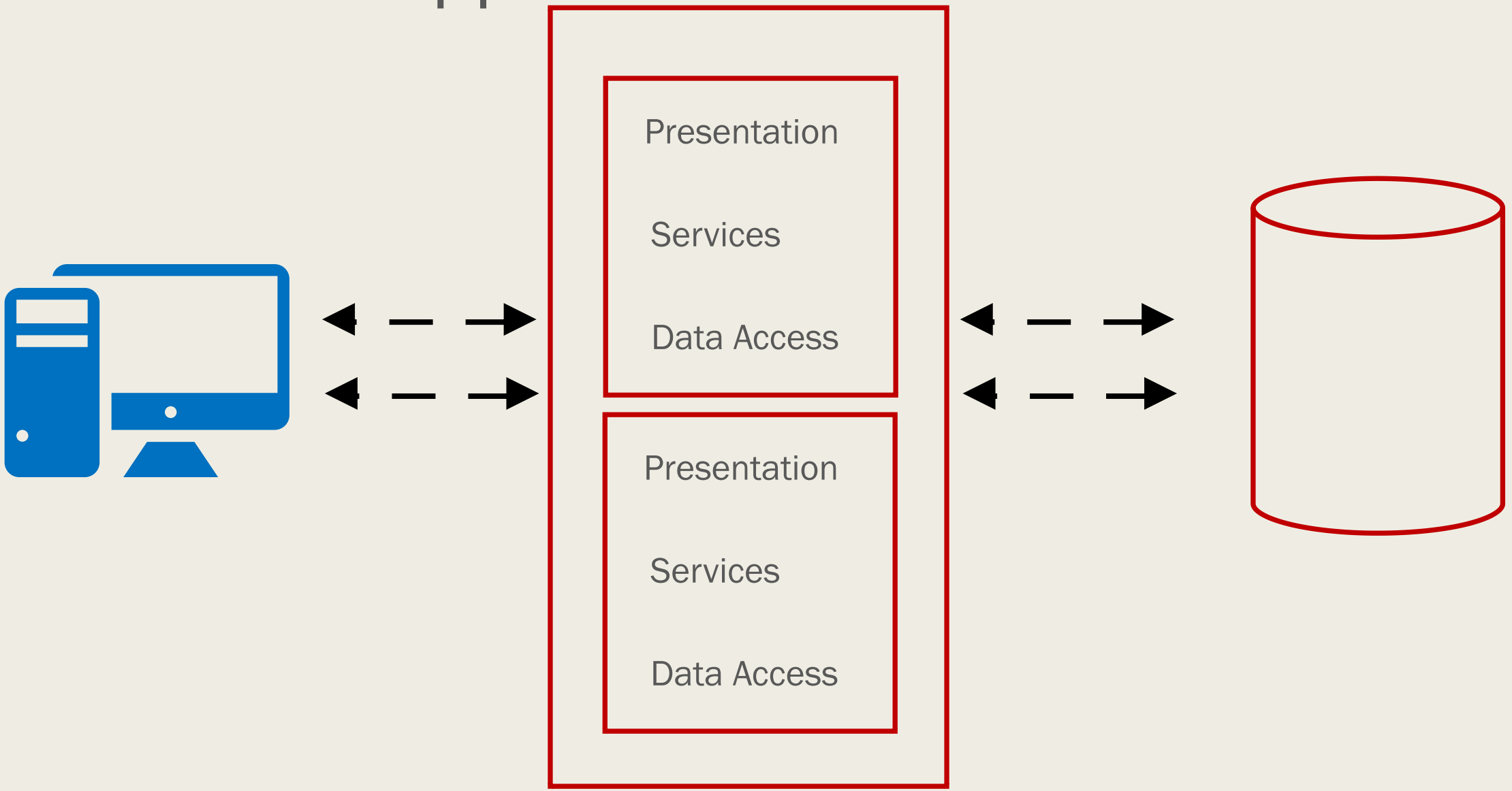
Традиционна Monolithic architecture

- Монолитният софтуер е проектиран да бъде самостоятелен; компонентите на програмата са взаимосвързани и взаимозависими, а не слабо свързани, какъвто е случаят с модулните софтуерни програми. В строго свързана архитектура, всеки компонент и свързаните с него компоненти трябва да присъстват, за да може кодът да бъде изпълнен или компилиран.

Нужна ли е промяна?

- Монолитните приложения могат да се превърнат в "Мега приложение". Ситуация, в която никои разработчик не познава пълната функционално на приложението
- Ограничена преизползваемост
- Разширението на монолитно приложение е голямо предизвикателство
- По дефиниция монолитните приложения са разработвани само от точно дефиниран технологичен стак. Това от своя страна може силно да лимитира разработката

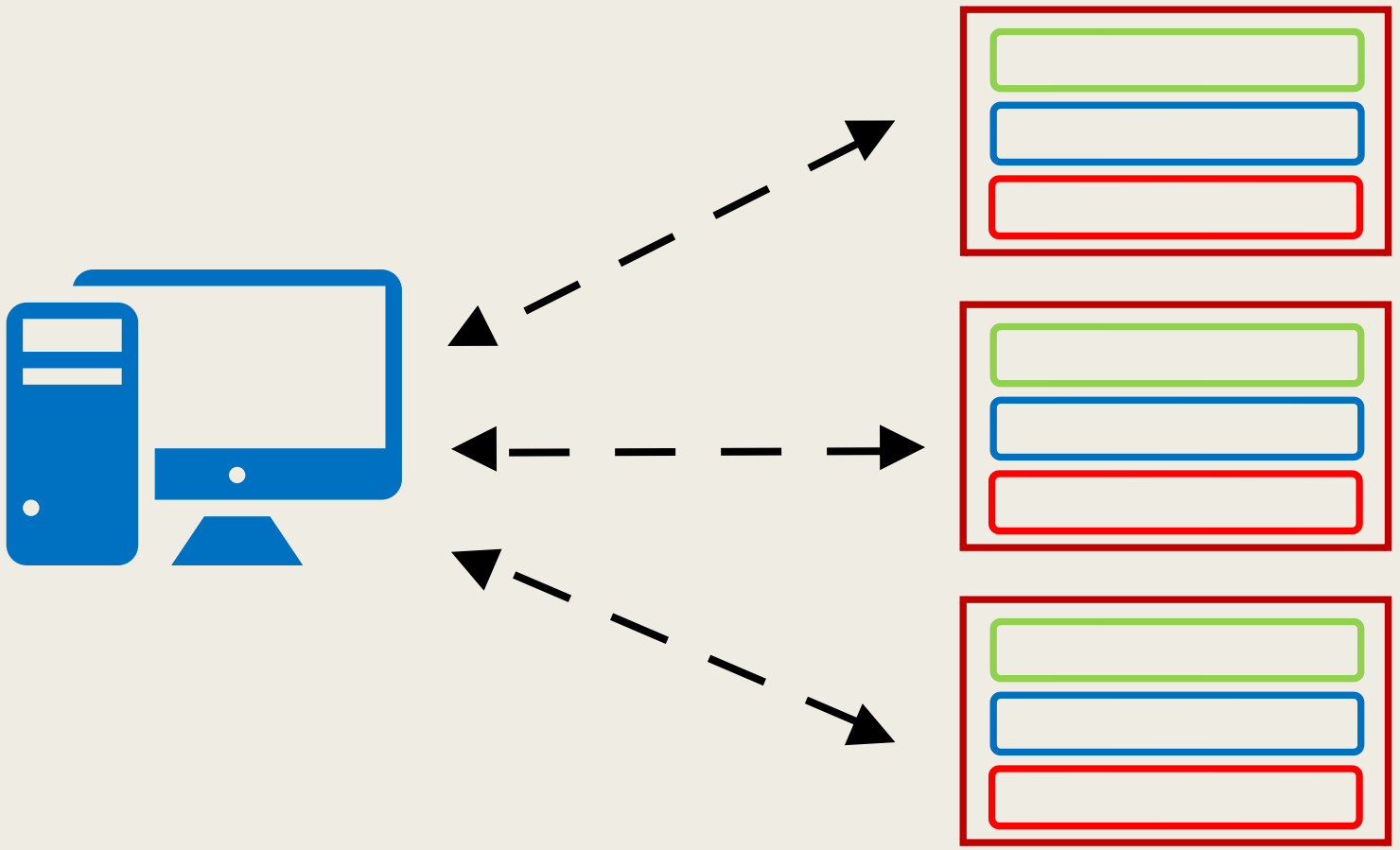
Monolithic application



Microservices

- Microservices architecture е архитектурен стил, който структурира приложение като съвкупност от свободно свързани услуги, които заедно предоставят бизнес логигиката на системата.

Microservices application

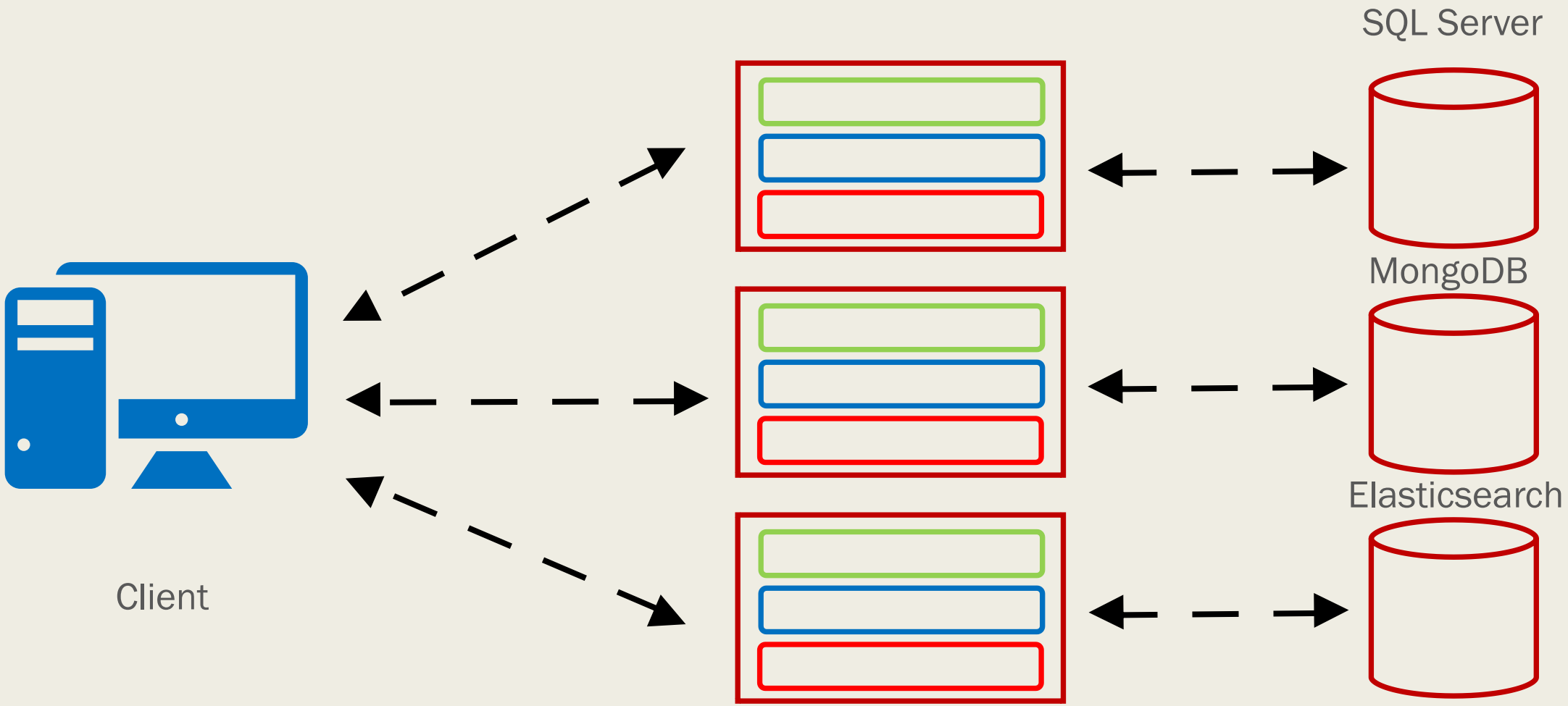


Психология на microservices

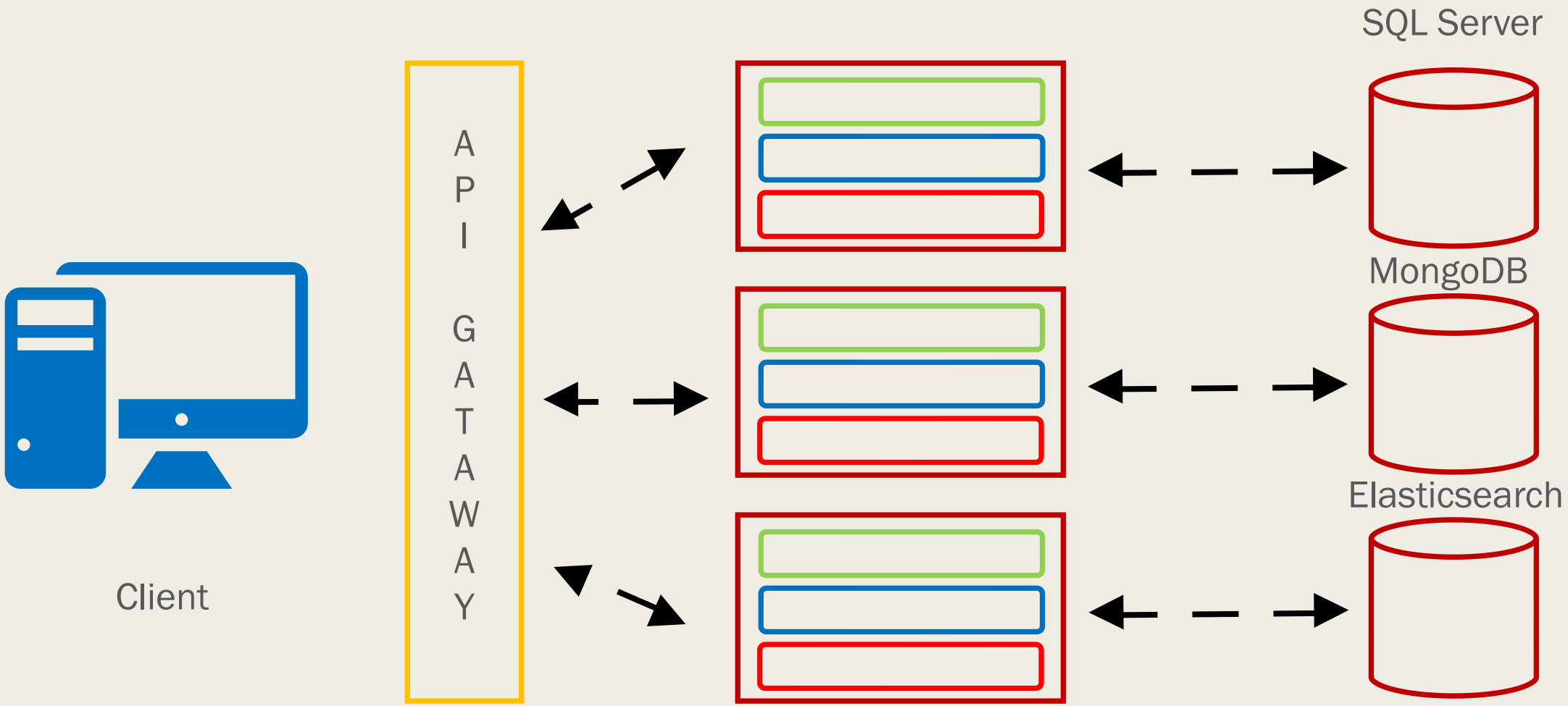
- Услугите трябва да са малки, добре структурирани за да могат да изпълняват само една функция
- Архитектурата трябва да обхваща автоматизираното тестване и внедряване
- Всяка услуга е еластична, композиционна, минимална и пълна.

Direct client communication vs API Gateway

Microservices implementation 1

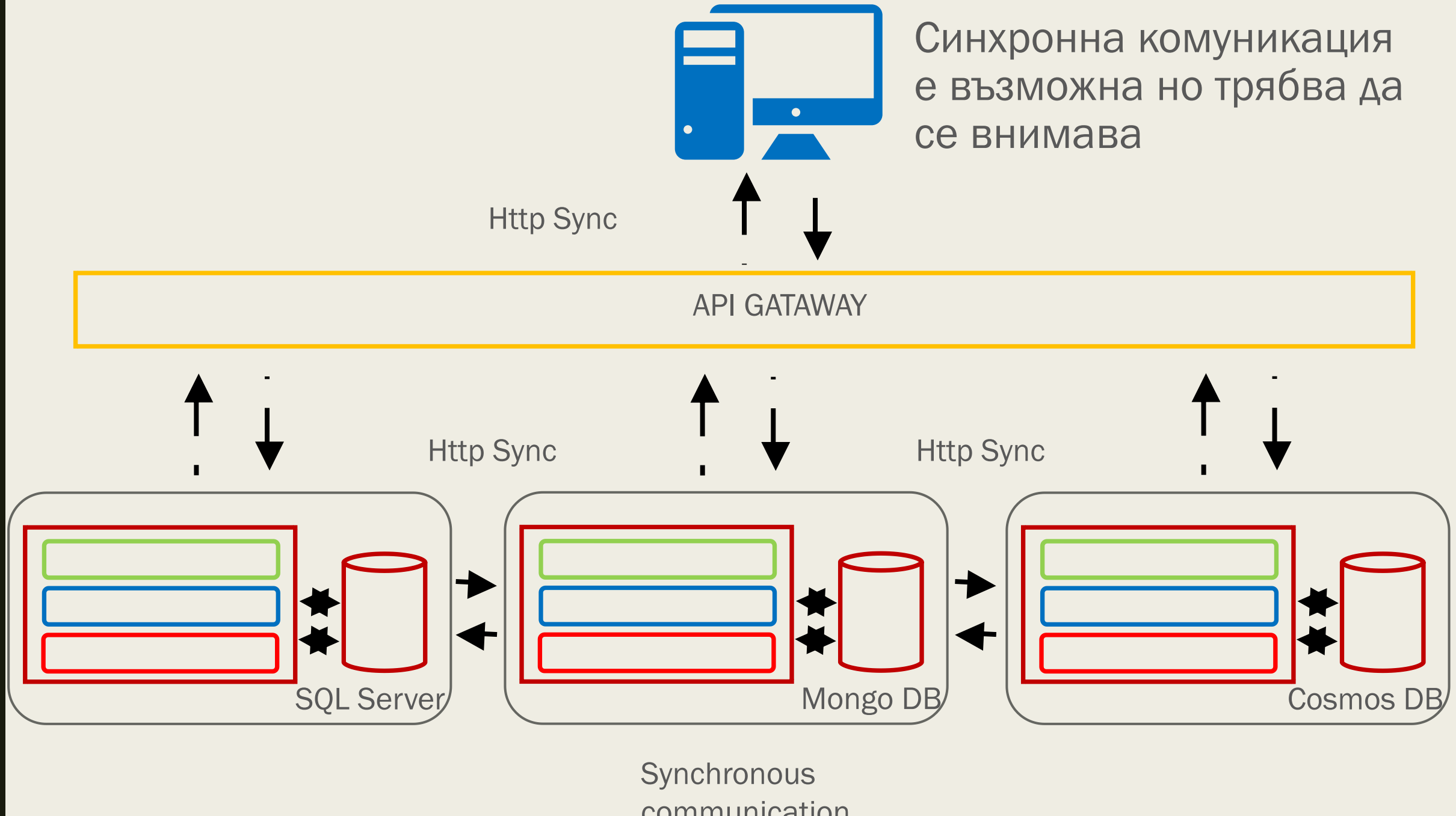


Microservices implementation 2



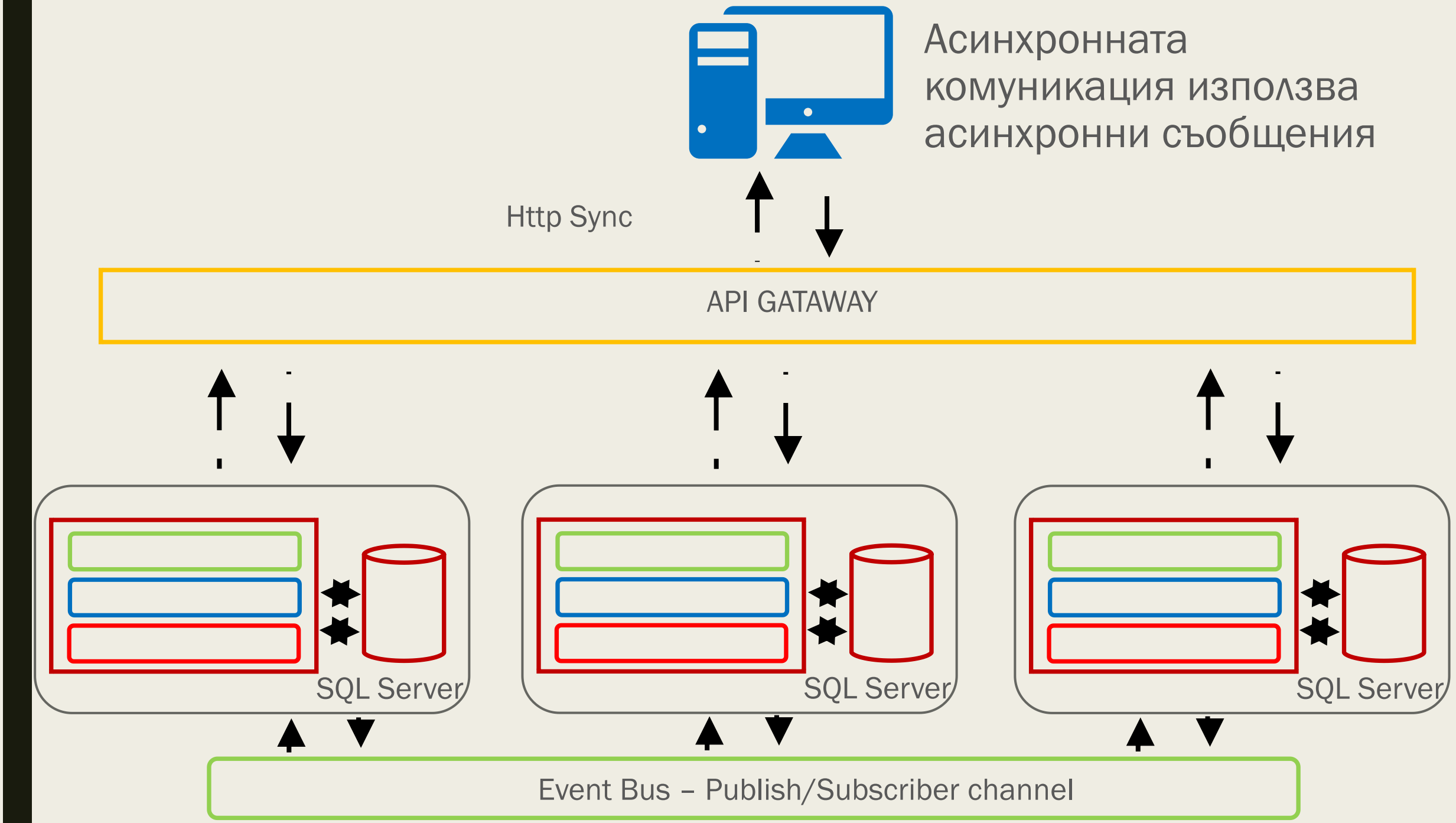
Synchronous vs Asynchronous Microservices communication

Синхронна комуникация е възможна но трябва да се внимава



Synchronous communication

Асинхронната комуникация използва асинхронни съобщения



Основни предимства

- Възможност за разширение – както хоризонтално така и вертикално
- Модулна структура
- Осигуря процес на непрекъснато обновяване.
  - *DevOps(CI/CD)*

Недостатъци

- Тестването и разпространението е по – трудно
- Асинхронната вътрешната комуникация между отделните услуги е значително по – тежка, от комуникацията между услуги изградени на основата на monolithic architecture
- Преместването на отговорностите(логика) между услуги е по-трудно
  - *Това може да включва комуникация между различни екипи, пренаписване на функционалност на друг език или поставянето на логиката в друга инфраструктура*

Недостатъци

- Разглеждането размера на услугите като основен структуриращ механизъм може да доведе до твърде много услуги, докато алтернативата на вътрешната модулация може да доведе до по - опростен дизайн

DEMO MICROSERVICES

https://github.com/pkyurkchiev/microservices_skeleton_net-core

ВЪПРОСИ ?