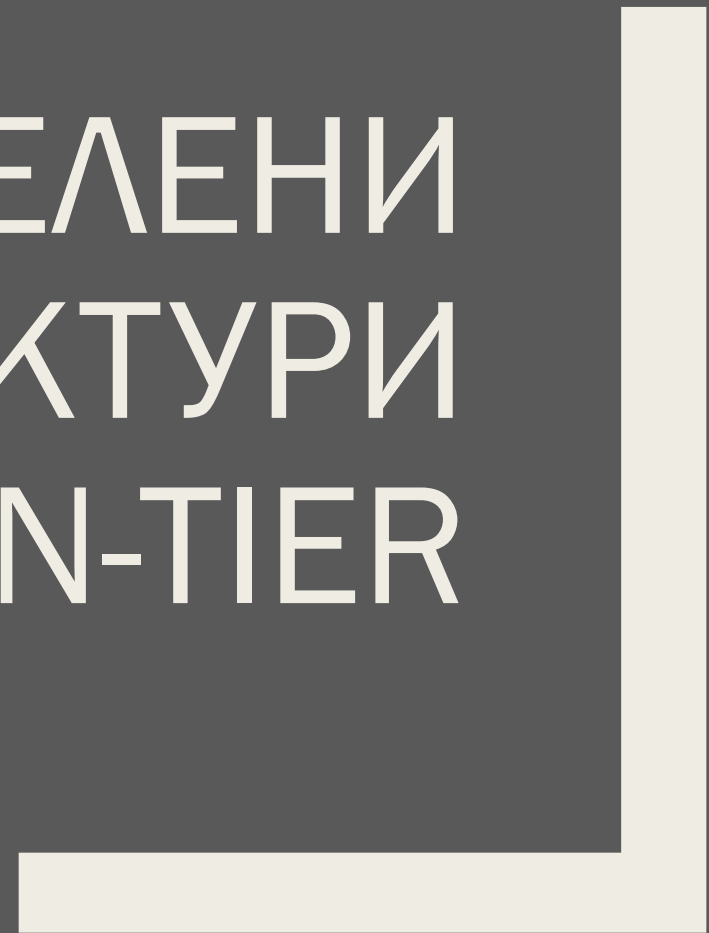


РАЗПРЕДЕЛЕНИ ПРИЛОЖЕНИЯ

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

РАЗПРЕДЕЛЕНИ АРХИТЕКТУРИ CLIENT-SERVER И N-TIER



Софтуерна архитектура

- Софтуерната архитектура е съвкупност от важни решения за организацията на програмните системи.

История

- Софтуерна архитектура като концепция има своите корени в изследванията на Едсгер Дейкстра през 1968 г. и Дейвид Парнас в началото на 1970.
- Добива широка популярност като термин в началото на 1990 -те години.

Архитектура

- Софтуерните архитектури са необходими за обосноваване на софтуерните системи. Всяка архитектура се състои от софтуерни елементи, връзките между тях и свойствата както на елементите, така и на връзките.

Основни твърдения:

- Софтуерните архитектури са важни за успешното развитие на софтуерни системи.
- Софтуерните системи се изграждат, за да удовлетворят бизнес целите на организациите.
- Архитектурата е мост между (често абстрактни) бизнес цели и крайната система.
- Софтуерните архитектури могат да бъдат проектирани, анализирани, документирани и реализирани с определени технологии, с цел да бъдат постигнати целите на бизнеса.

Архитектурни дейности

■ Архитектурен анализ

- *Включва процес на разбиране на средата, в която ще работи планираната система или системи, и определяне на бизнес изискванията.*

■ Архитектурен синтез

- *Синтез, или дизайн, е процесът на създаване на архитектура.*

■ Оценяване на архитектурата

- *Процес на определяне колко добре настоящият дизайн, или част от него, отговаря на изискванията, получени по време на анализа.*

■ Еволюция на архитектурата

- *Процес на поддържане и адаптиране на съществуваща софтуерна архитектура. Целта е тя да продължи да отговаря на изискванията и промените в нейната заобикаляща среда.*

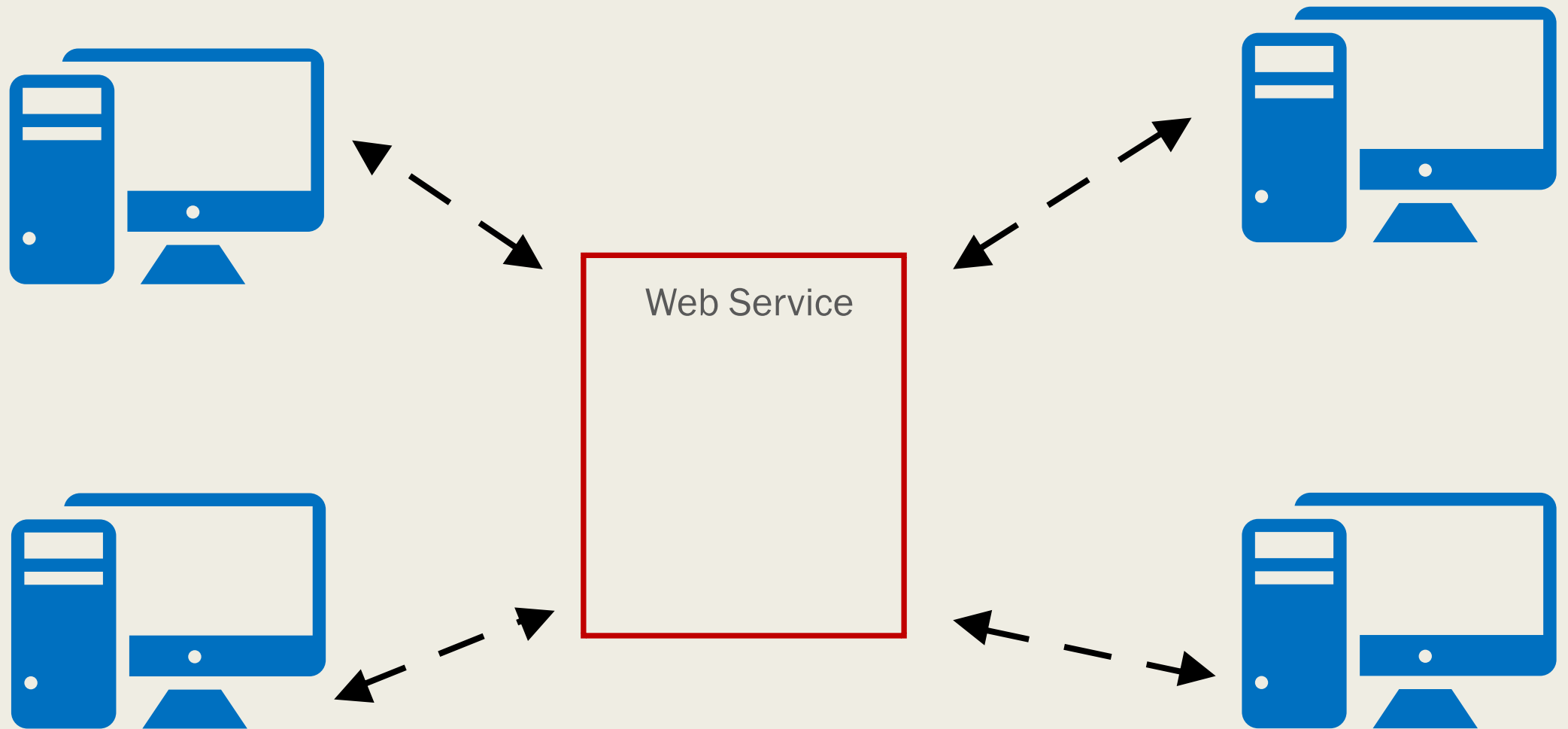
Архитектури, които ще разгледаме

- Client-Server
- N-tiers (3-tiers)
- SOA
- Microservices

Client-Server

- Моделът клиент-сървър е разпределена архитектура на приложения, която разделя задачите (натоварването) между доставчиците на ресурс (услуга), наречени сървъри и инициатори, наречени клиенти.

Какво е Client-Server?



Какво е клиент-сървър приложение?

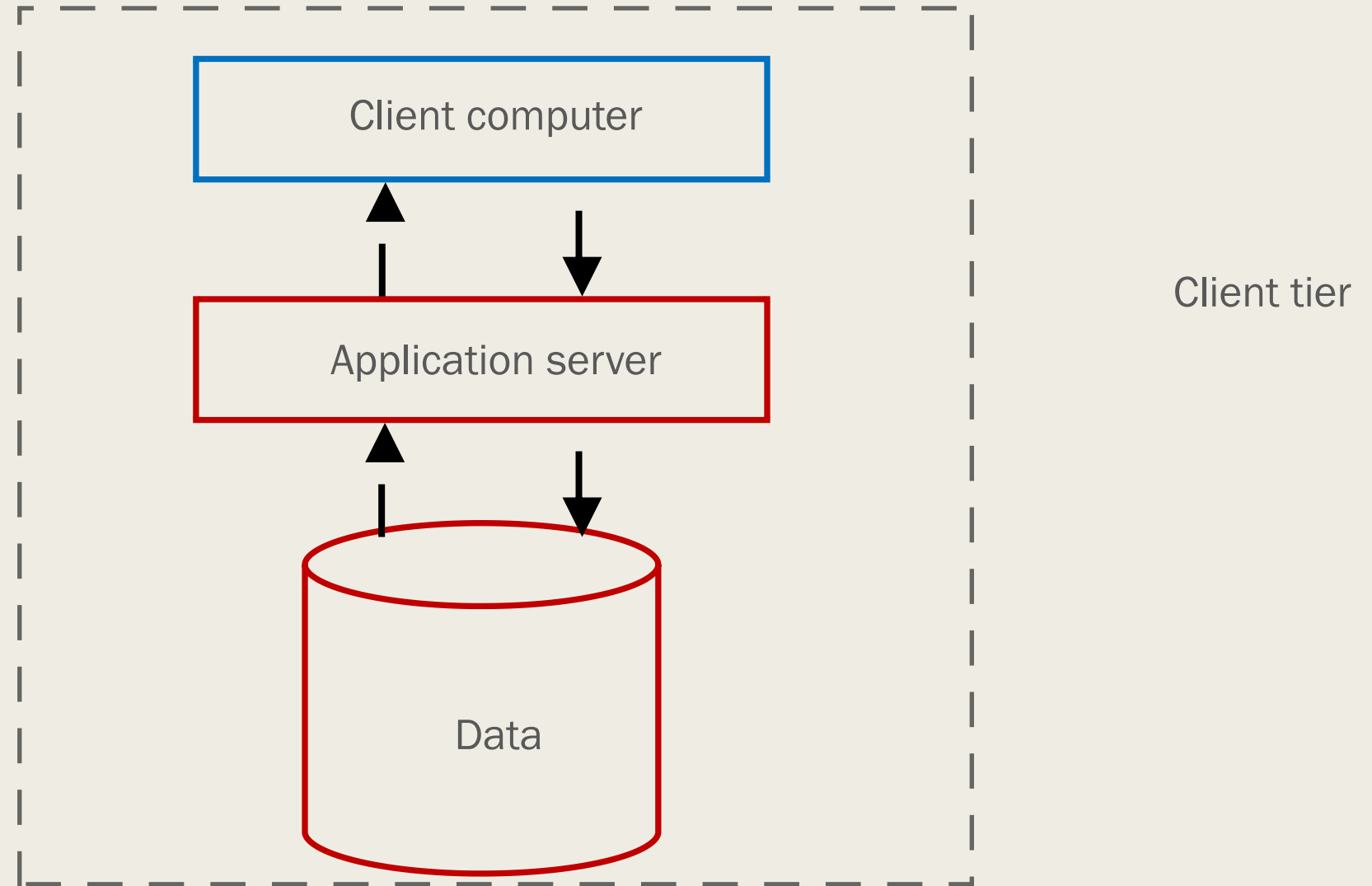
- Посредством клиента, потребителят взаимодейства с приложението.
- Един сървър може да обслужва много клиенти и да предоставя функционалности, които не могат да се реализират самостоятелно при клиента.
- Когато клиентът има нужда от информация или да предаде информация на сървъра в резултат на взаимодействието на потребителя с приложението, клиентът може да изпрати съобщение към сървъра.
- В отговор на това съобщение, сървърът изпълнява необходимите действия и може да върне съобщение (наречено още отговор).

- При някои клиент-сървър приложения сървърът може да изпраща съобщения към клиента, които не е задължително да бъдат отговор на съобщения, изпратени от клиента.
- Често клиентът и сървърът комуникират през компютърната мрежа (локална мрежа или интернет) и се намират на различно хардуерно осигуряване.

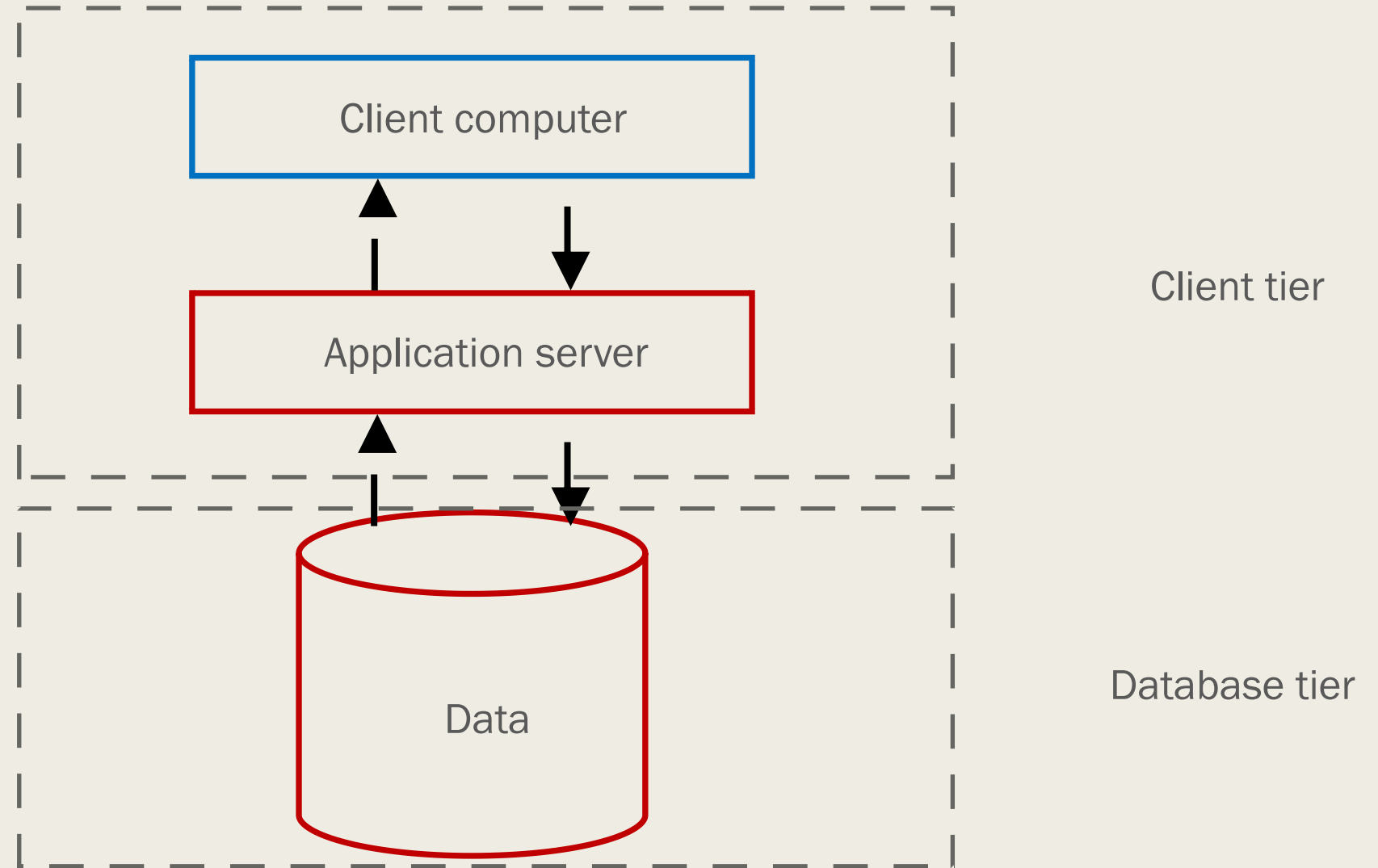
N-tier(3-tier)

- Многослойната архитектура е вид клиент-сървър архитектура, при която бизнес логиката е разделена на отделни компоненти. Най-често презентационният слой, бизнес логиката и логиката за управление на данните са отделени физически.

1-tier architecture



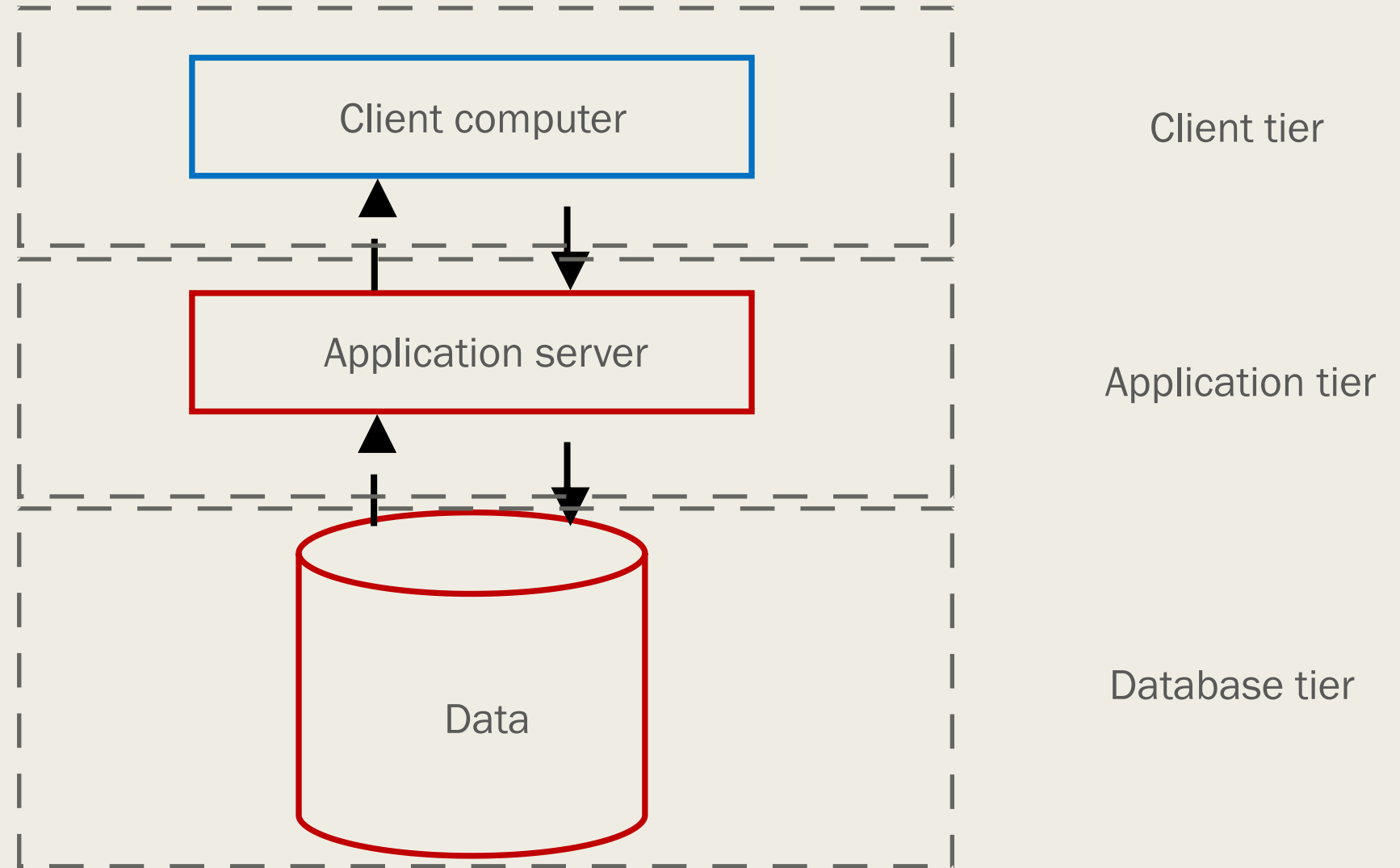
2-tier architecture



Трислойна архитектура

- Презентационен слой
 - Презентационният слой е слой, до който потребителят има директен достъп.
- Слой за бизнес логика (БЛ)
 - БЛ слой контролира функционалността на приложението (бизнес логиката) и е съсредоточен върху обработката на данните.
- Слой за данните
 - Този слой се състои от сървър за база данни (СУБД). Служи за управление на данните.

3-tier architecture



Типове N-tier

- Closed layer architecture

- *В затворената архитектура, всеки слой може да се свърже само с най-близкия до него.*

- Open layer architecture

- *В отворената архитектура, всеки слой може да се свърже с всеки слой, намиращ се под него.*

Кога да използваме N-Tier?

- При опростени уеб приложения.
- При миграция на локални приложения към облачна среда (AWS, Azure, Google Cloud).
- При обединена разработка на локални и облачно базирани приложения.

Предимства

- Преносимост на едно и също приложение между облачна и локална платформа, или между различни облачни платформи.
- По-малко време за обучение на голяма част от разработчиците.
- Естествена еволюция от традиционния модел на разработка.
- Отворен към комбинация от различни среди на разработка (Windows, Linux, Unix ...).

DEMO N-TIER

https://github.com/pkyurkchiev/n-layer_skeleton_net-core

ВЪПРОСИ ?

