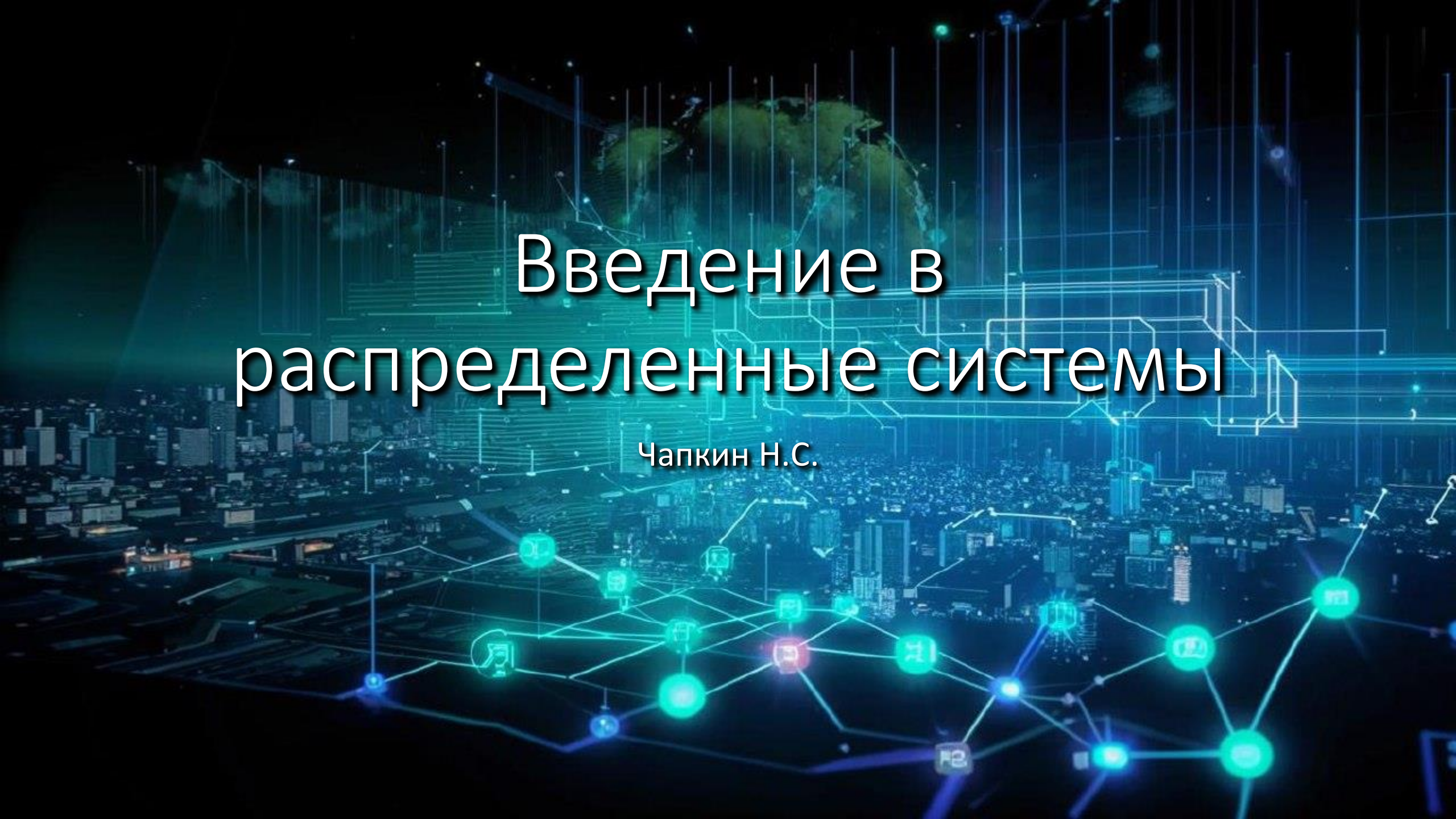


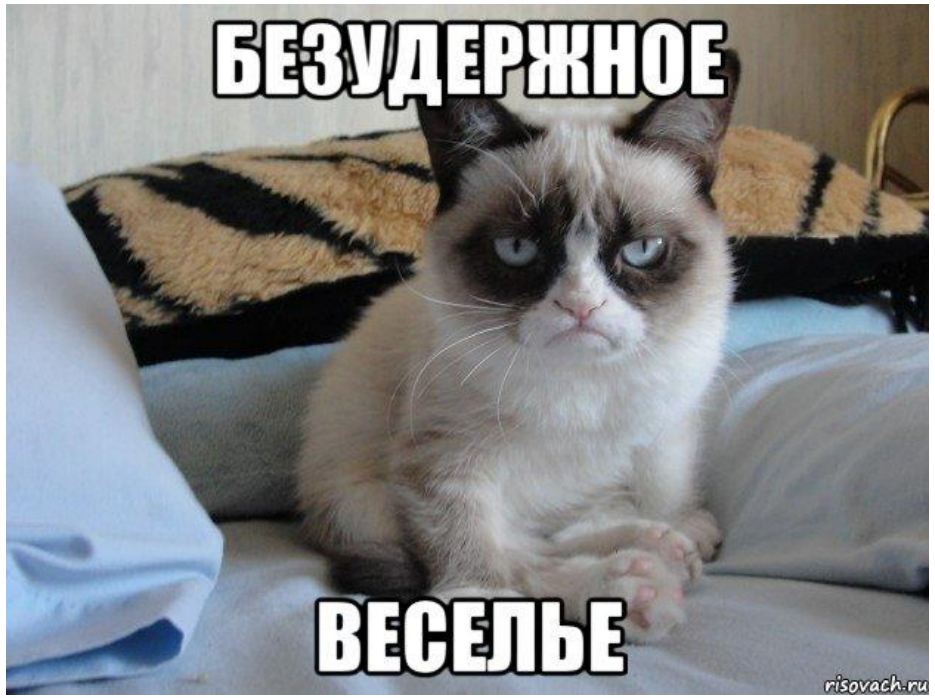
Введение в распределенные системы

Чапкин Н.С.

Понятие распределенной системы

Что такое распределенная система?

- Это вычислительная система, в которой неисправность компьютера, о существовании которого пользователи ранее даже не подозревали, приводит к остановке всей их работы



Понятие распределенной системы

А если, серьезно, что такое распределенная система?

- Это разделение функций системы между несколькими независимыми компьютерами?
- Или… это разделение функций системы между несколькими независимыми подсистемами?

Давайте подумаем…

Погрузимся немного в историю…

Развитие языков программирования:

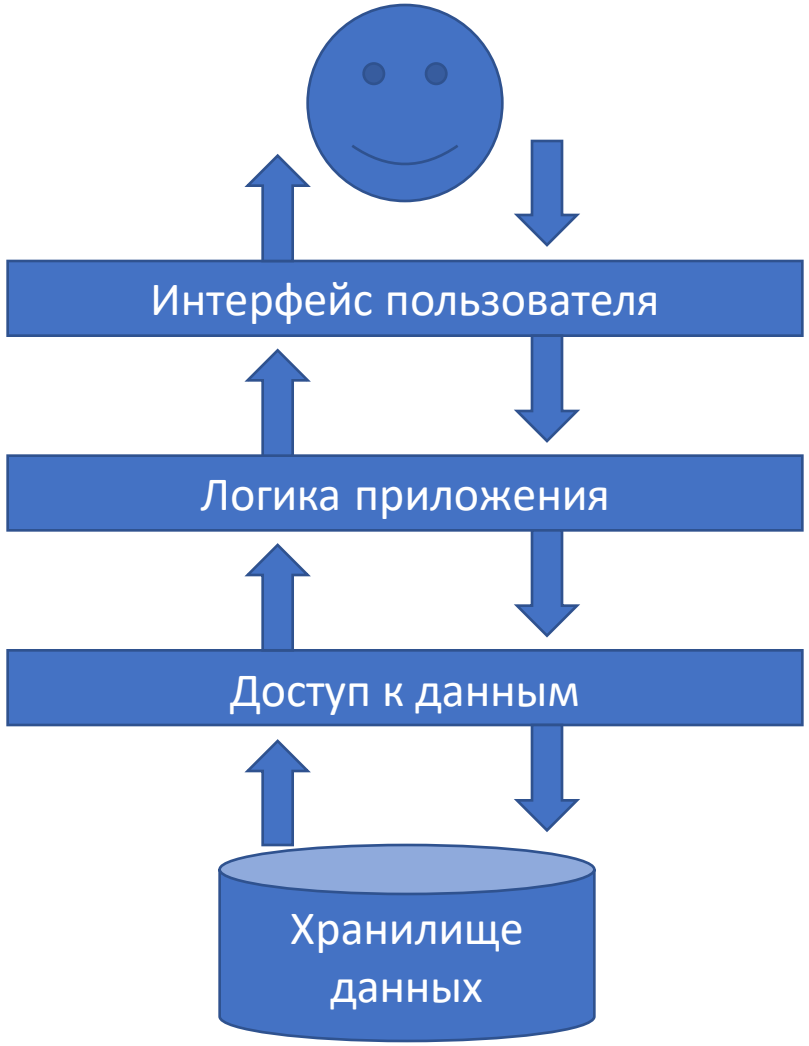
- Линейное программирование
- Процедурное программирование
- Объектно-ориентированное программирование

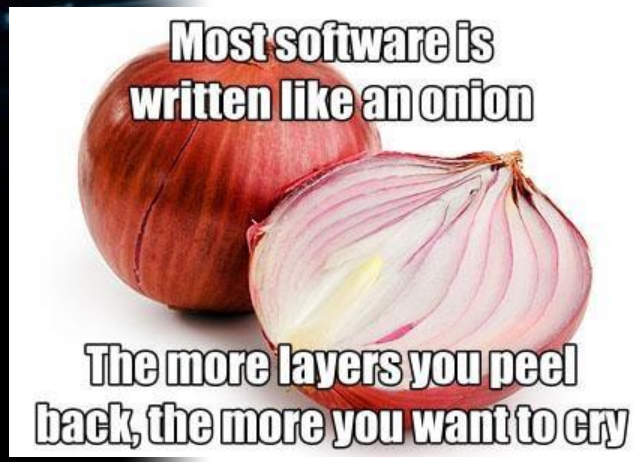
Развитие архитектуры приложений?

- Сначала программы работали на одном компьютере в рамках одного процесса…

Логические уровни приложения

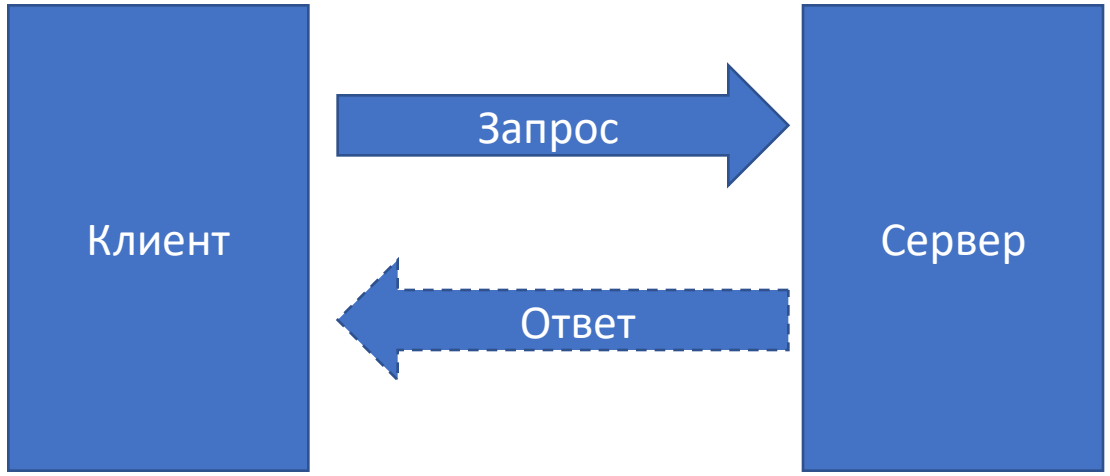
Приложения работали на одном компьютере в рамках одного процесса. НО! Архитектурно, приложение было удобно разделить на слои





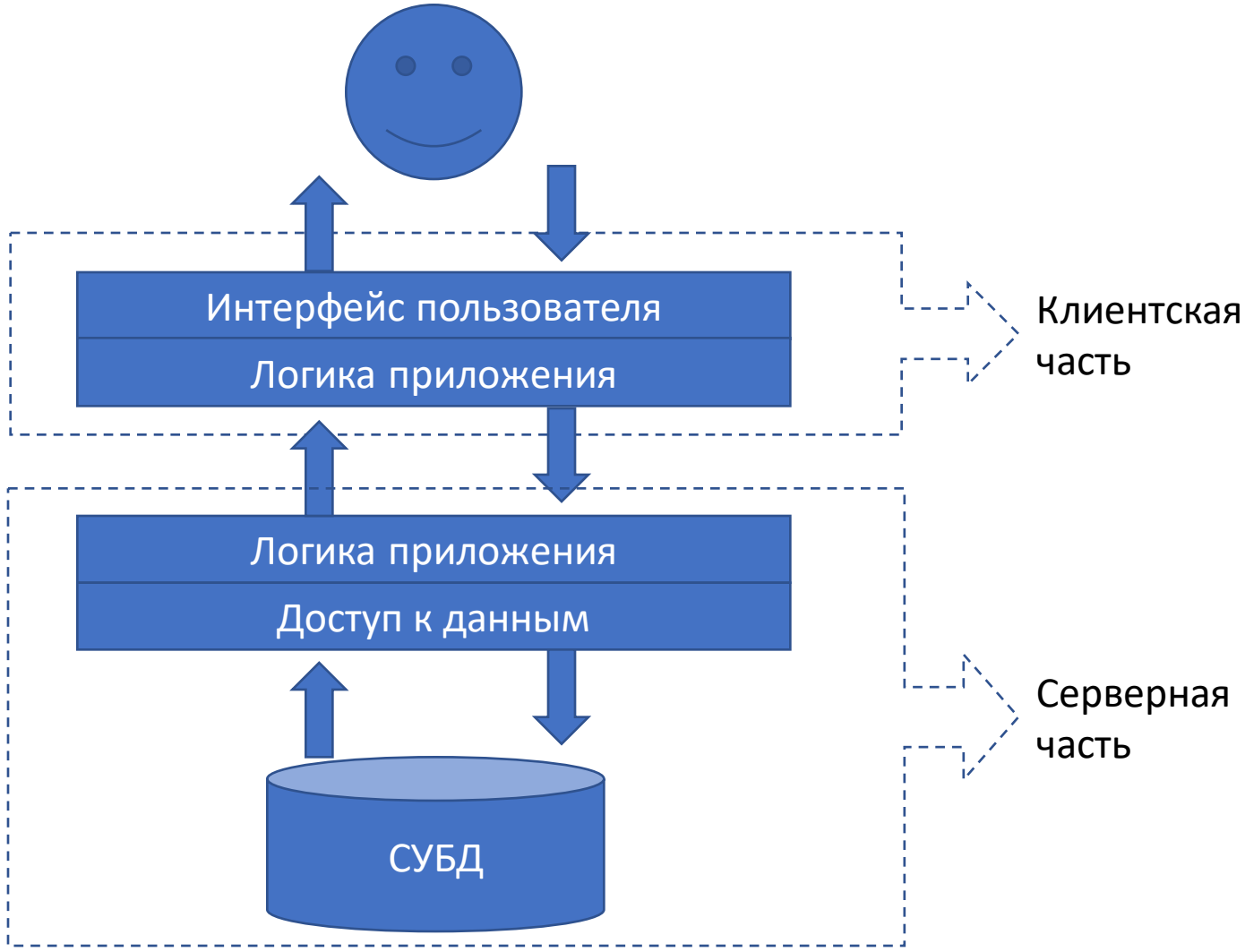
Модель взаимодействия клиент-сервер

Дальнейшая эволюция разработки ПО пошла в сторону клиент-серверных приложений. То есть когда есть один сервер с которым взаимодействуют множество клиентов



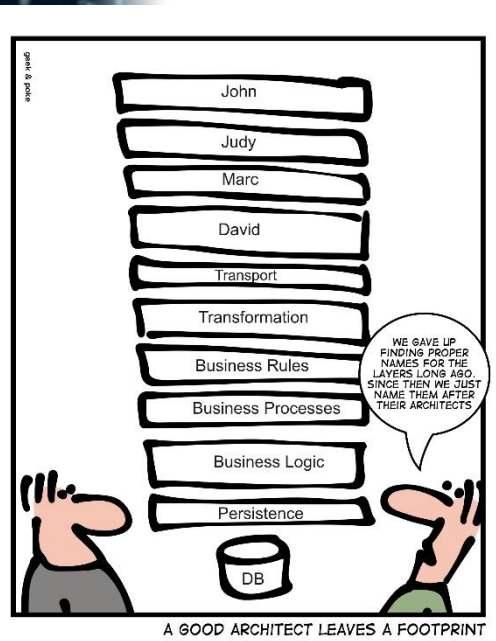
Двухзвенная архитектура

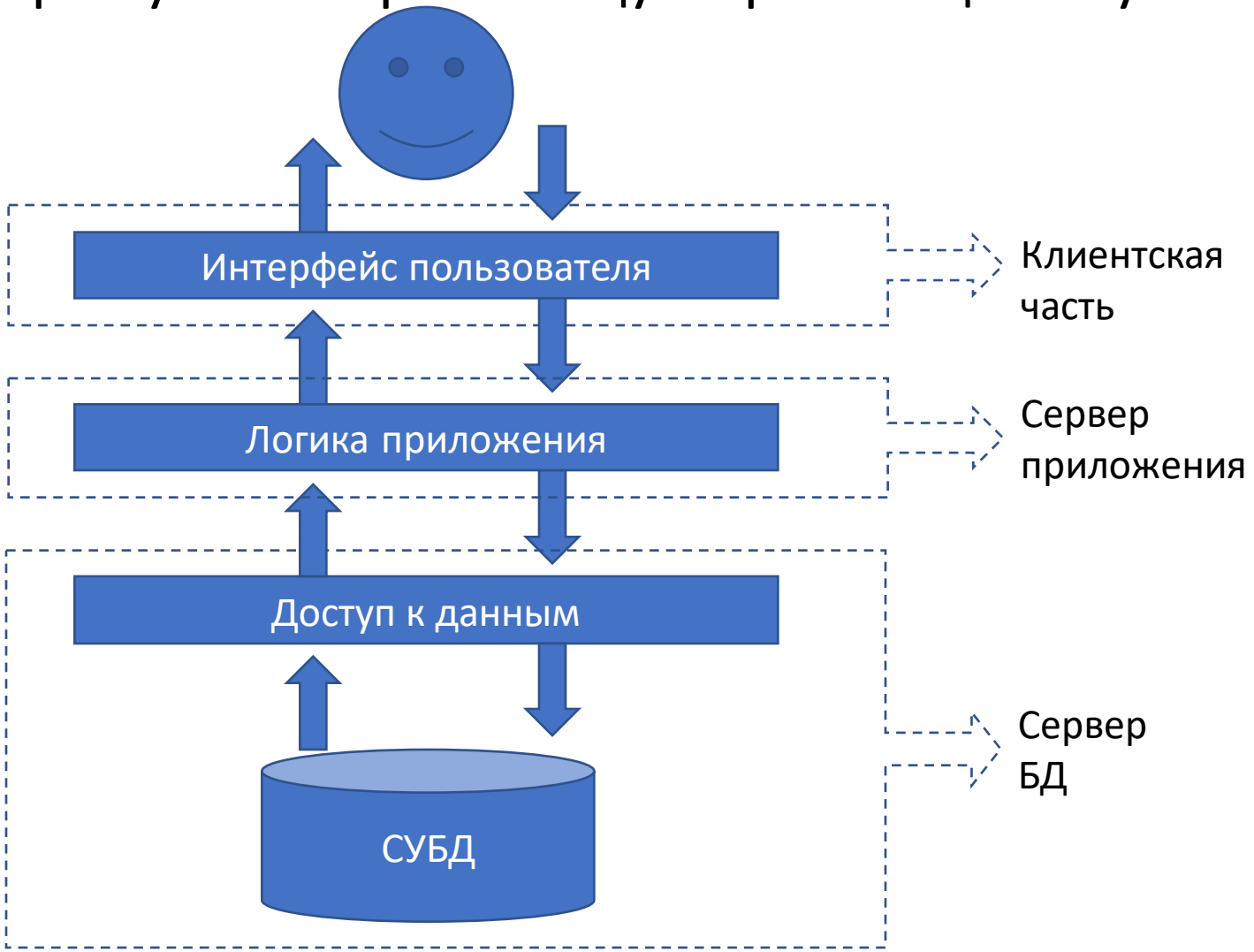
Наложение клиент-серверной архитектуры на трехуровневную логическую архитектуру приложения породило такую схему



Трехзвенная архитектура

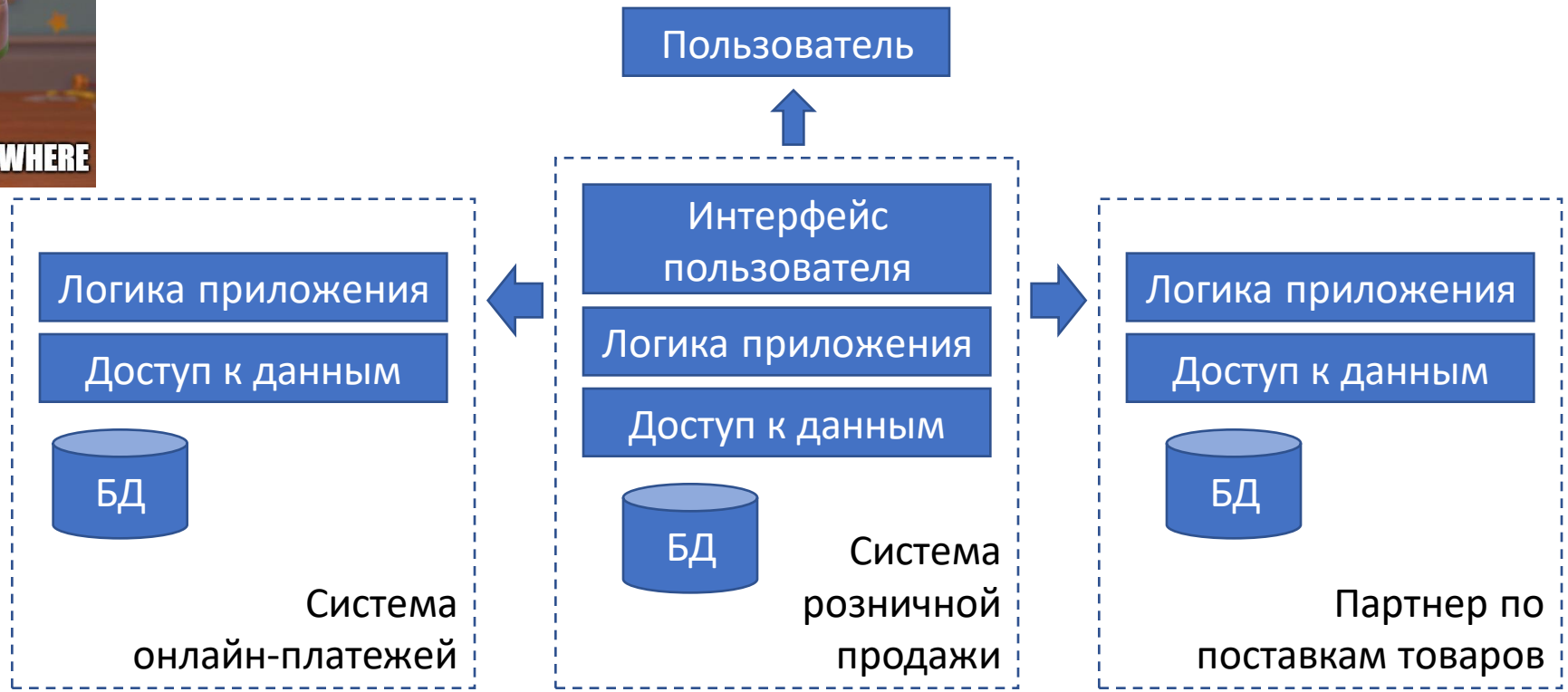
Логичным продолжением двухзвенной архитектуры явилась трехзвенная, которая учла возрастающую транзакционную нагрузку





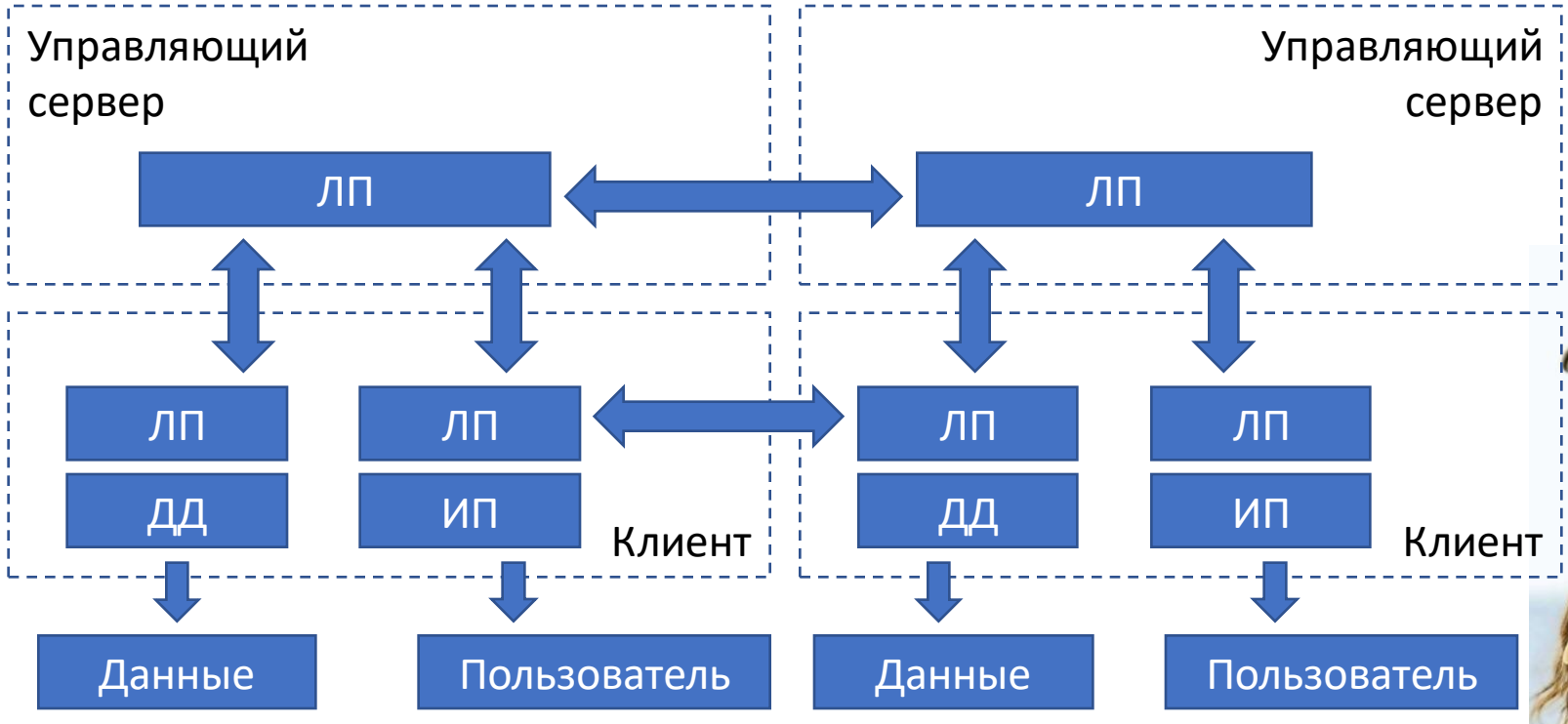
Распределенная система розничных продаж

Дальнейшее развитие связей между программными системами привело к появлению более сложных структур взаимодействия



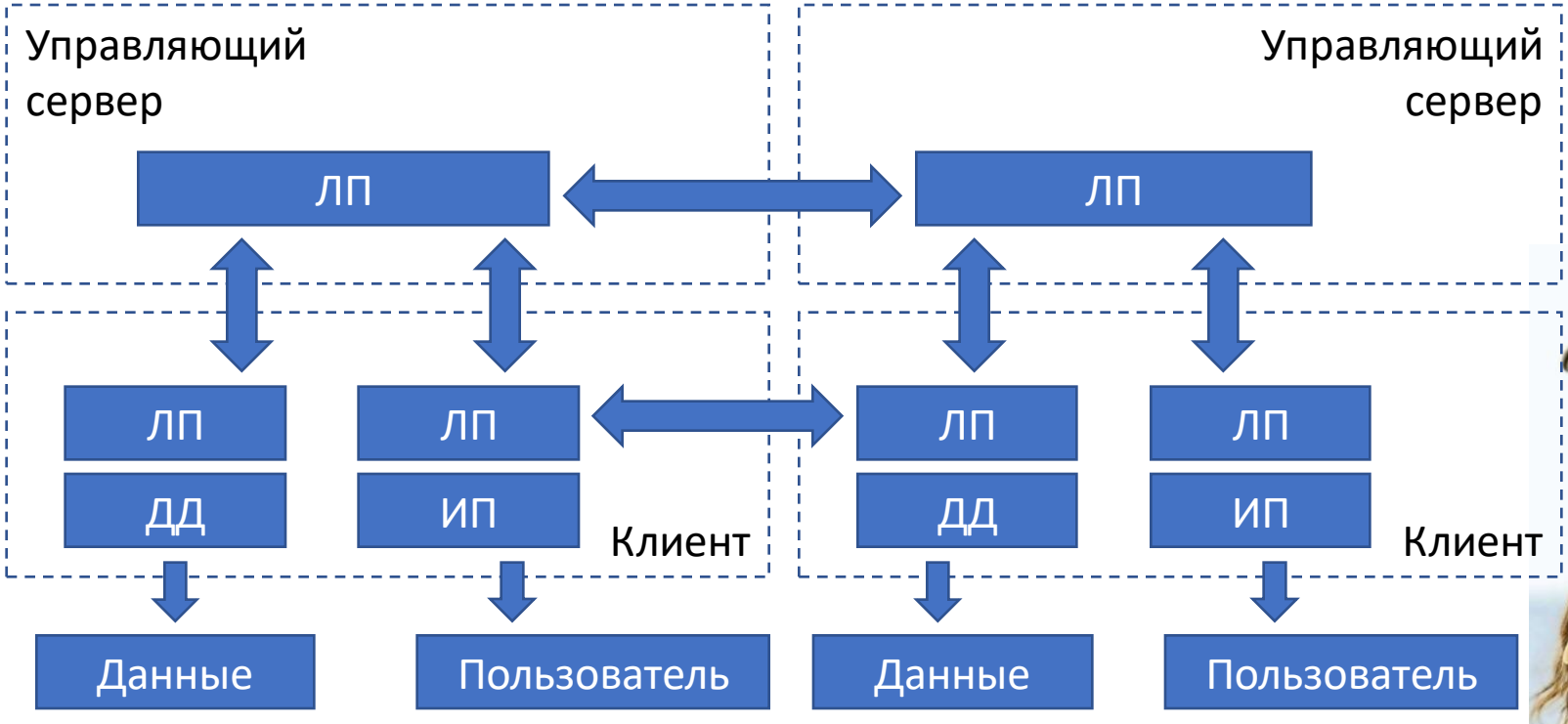
Вопрос на внимательность – какая система здесь представлена?

- HINT: Буквально каждый из вас ею пользовался



Вопрос на внимательность – какая система здесь представлена?

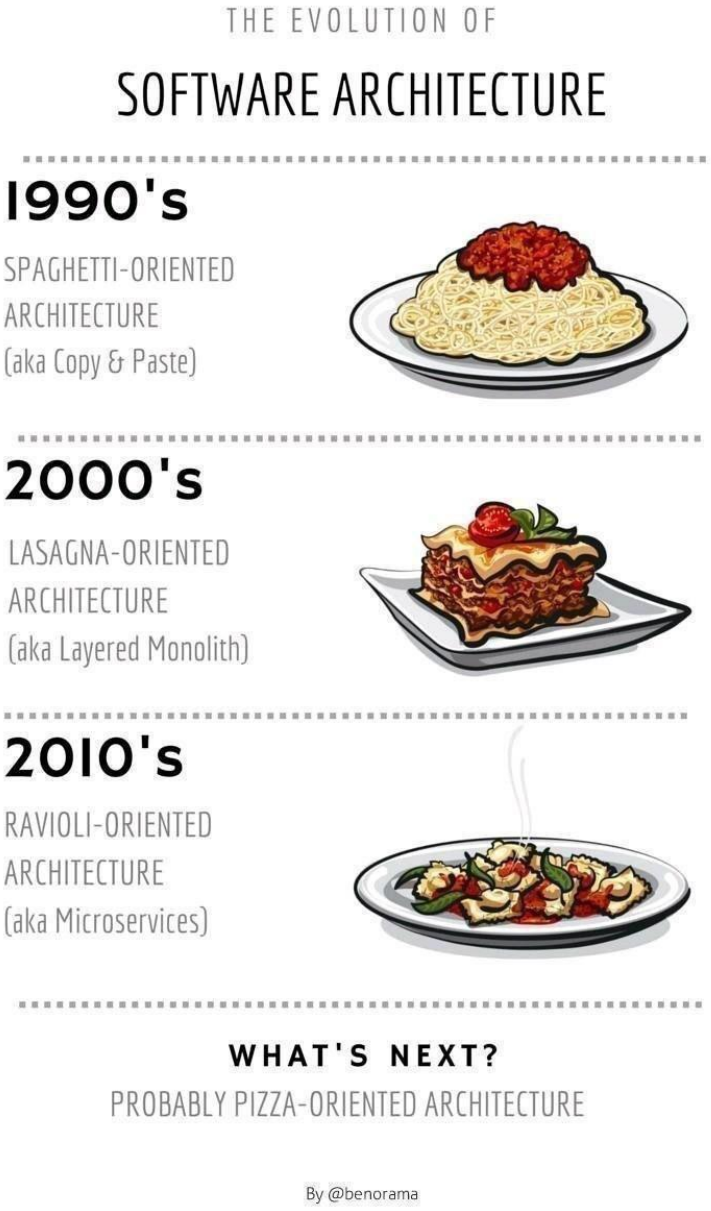
- HINT: Буквально каждый из вас ею пользовался
- Torrent tracker

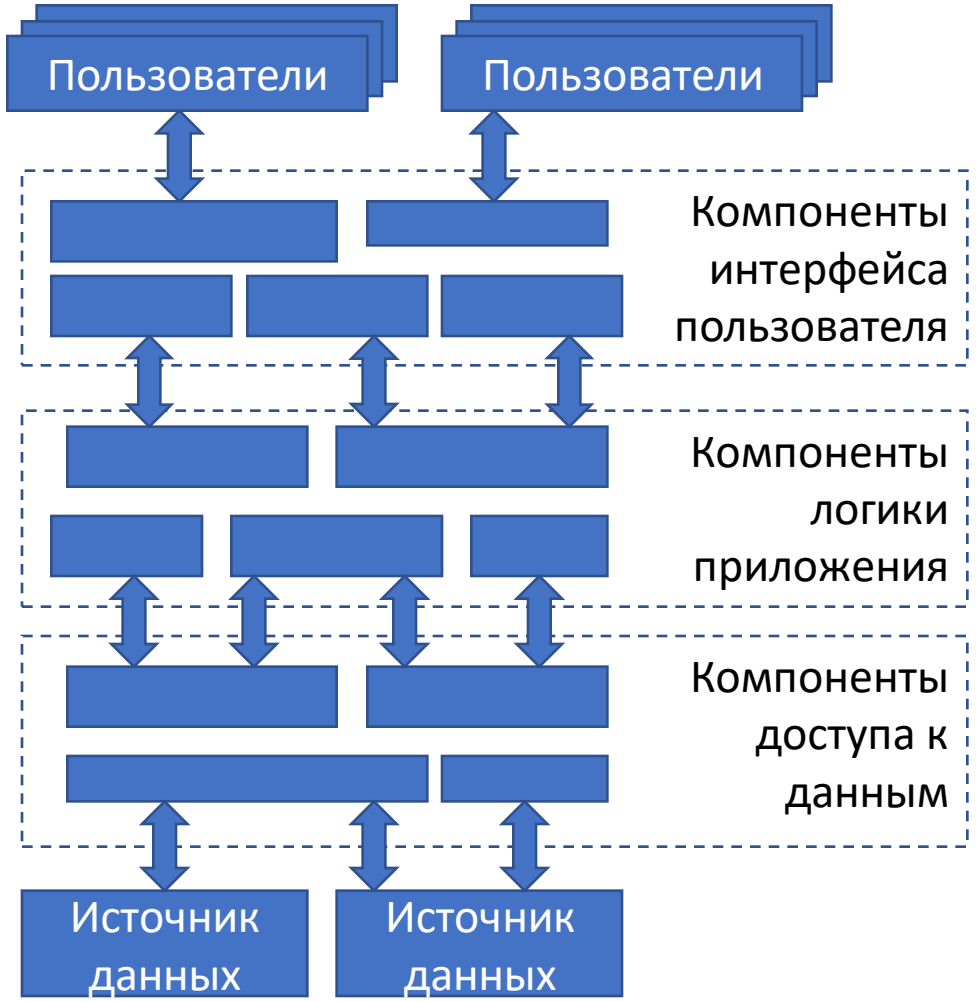


Программная компонента

Программная компонента – это **единица** программного обеспечения, исполняемая на **одном компьютере** в пределах **одного процесса**, и предоставляющая некоторый **набор сервисов**, которыми пользуются через внешний интерфейс программной компоненты другие программные компоненты, выполняющиеся на этом же компьютере, или на удаленных компьютерах

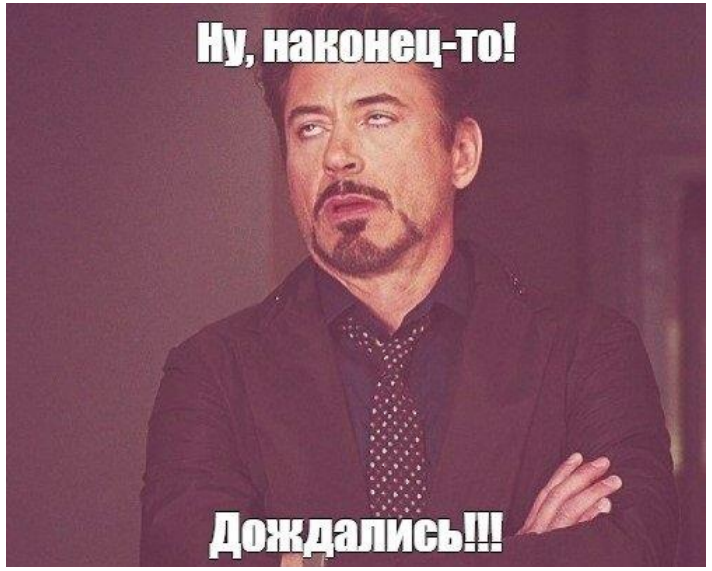
Программные компоненты





И-и-и… определение распределенной системы

Распределенная система есть **набор взаимодействующих программных компонент**, выполняющихся **на одном или нескольких связанных компьютерах** и выглядящих с точки зрения пользователя системы как единое целое



Требования к информационным системам

- Какие обычно вы предъявляете требования к компьютерным программам?

- Можно ли подойти и к распределенным системам с теми же требованиями?
- Ведь они:
  a) потенциально работают на нескольких компьютерах
  b) состоят из нескольких программных компонент
  c) могут не взаимодействовать с пользователями
  d) могут взаимодействовать с другими информационными системами
  e) состояние системы не принадлежит одному программному компоненту
  f) и т.д. и т.п…

Основные требования к распределенным системам

- Открытость
- Масштабируемость
- Поддержание логической целостности данных
- Устойчивость
- Безопасность
- Эффективность

Требование открытости

- Открытая система - это система, реализующая открытые спецификации (стандарты) на интерфейсы, службы и поддерживаемые форматы данных, достаточные для того, чтобы обеспечить:
  - возможность переноса разработанного прикладного программного обеспечения на широкий диапазон систем с минимальными изменениями (мобильность приложений, переносимость)
  - совместную работу (взаимодействие) с другими прикладными приложениями на локальных и удаленных платформах (интероперабельность, способность к взаимодействию)
  - взаимодействие с пользователями в стиле, облегчающим последним переход от системы к системе (мобильность пользователя)

Требование масштабируемости

Масштабируемость бывает разная:

- Размер системы
- Географическое положение компонентов систем
- Административное устройство систем

Для достижения масштабируемости требуется решить ряд проблем в :

- Обслуживании (один сервер для множества клиентов)
- Данных (множественный доступ к одному файлу данных)
- Алгоритмах (перегрузка коммуникаций из-за использования централизованных алгоритмов)

Требование масштабируемости

Основные вызовы:

- Недетерминированный таймаут
- Местоположение необходимой службы заранее неизвестно
- Аппаратные решения могут быть гетерогенными

Вывод:

- Сами системы могут существовать продолжительное время, но отдельные их части могут время от времени отключаться
- При этом пользователи и приложения не должны уведомляться о том, что эти части вновь подключены, что добавлены новые части для поддержки дополнительных пользователей или приложений

Требование поддержания логической целостности данных

- Запрос пользователя в распределенной системе должен либо корректно выполняться целиком, либо не выполняться вообще
- Ситуация, когда часть компонент системы корректно обработали поступивший запрос, а часть – нет, является наихудшей

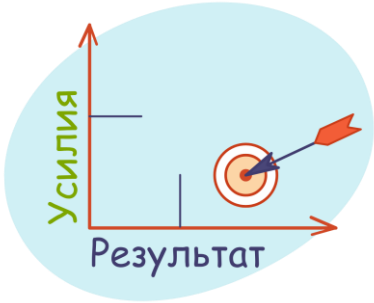
Требование устойчивости

- Под устойчивостью понимается возможность дублирования несколькими компьютерами одних и тех же функций или же возможность автоматического распределения функций внутри системы в случае выхода из строя одного из компьютеров
- В идеальном случае это означает полное отсутствие уникальной точки сбоя, то есть выход из строя одного любого компьютера не приводит к невозможности обслужить запрос пользователя

Требование безопасности

- Каждый компонент, образующий распределенную систему, должен быть уверен, что его функции используются авторизированными на это компонентами или пользователями
- Данные, передаваемые между компонентами, должны быть защищены как от искажения, так и от просмотра третьими сторонами

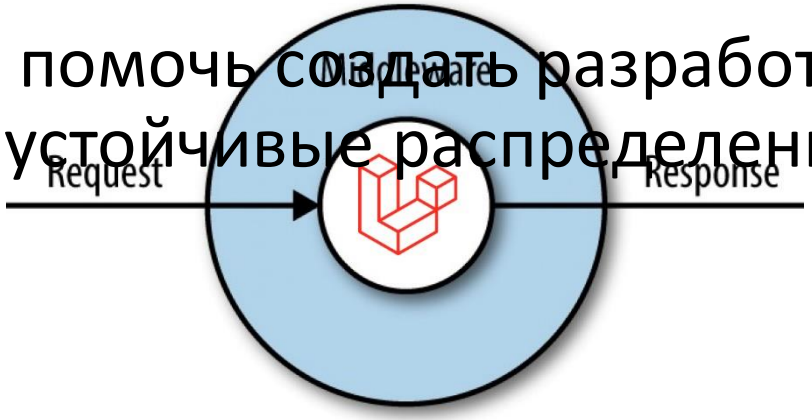
Требование эффективности



- В узком смысле применительно к распределенным системам под эффективностью будет пониматься минимизация накладных расходов, связанных с распределенным характером системы
- Поскольку эффективность в данном узком смысле может противоречить безопасности, открытости и надежности системы, следует отметить, что требование эффективности в данном контексте является наименее приоритетным
- Например, на поддержку логической целостности данных в распределенной системе могут тратиться значительные ресурсы времени и памяти, однако система с недостоверными данными вряд ли нужна пользователям
- Желательным свойством промежуточной среды является возможность организации эффективного обмена данными, если взаимодействующие программные компоненты находятся на одного компьютере
- Эффективная промежуточная среда должна иметь возможность организации их взаимодействия без затрагивания стека TCP/IP. Для этого могут использоваться системные сокеты (unix sockets) в POSIX-системах или именованные каналы (named pipes)

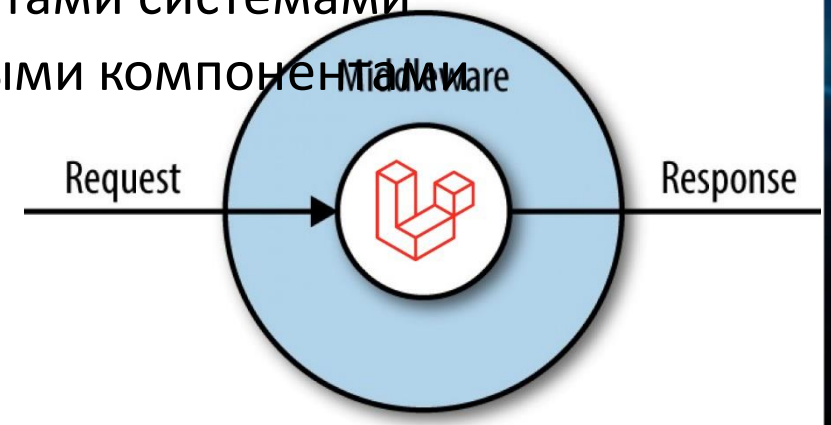
Понятие промежуточной среды

- С точки зрения одного из компьютеров распределенной системы, все другие входящие в нее машины являются удаленными вычислительными системами
- Для выполнения сформулированных выше требований к распределенным системам функции сеансового и представительского уровня должна взять на себя некоторая промежуточная среда (middleware), называемая также промежуточным программным обеспечением
- Такая среда должна помочь создать разработчиками открытые, масштабируемые и устойчивые распределенные системы

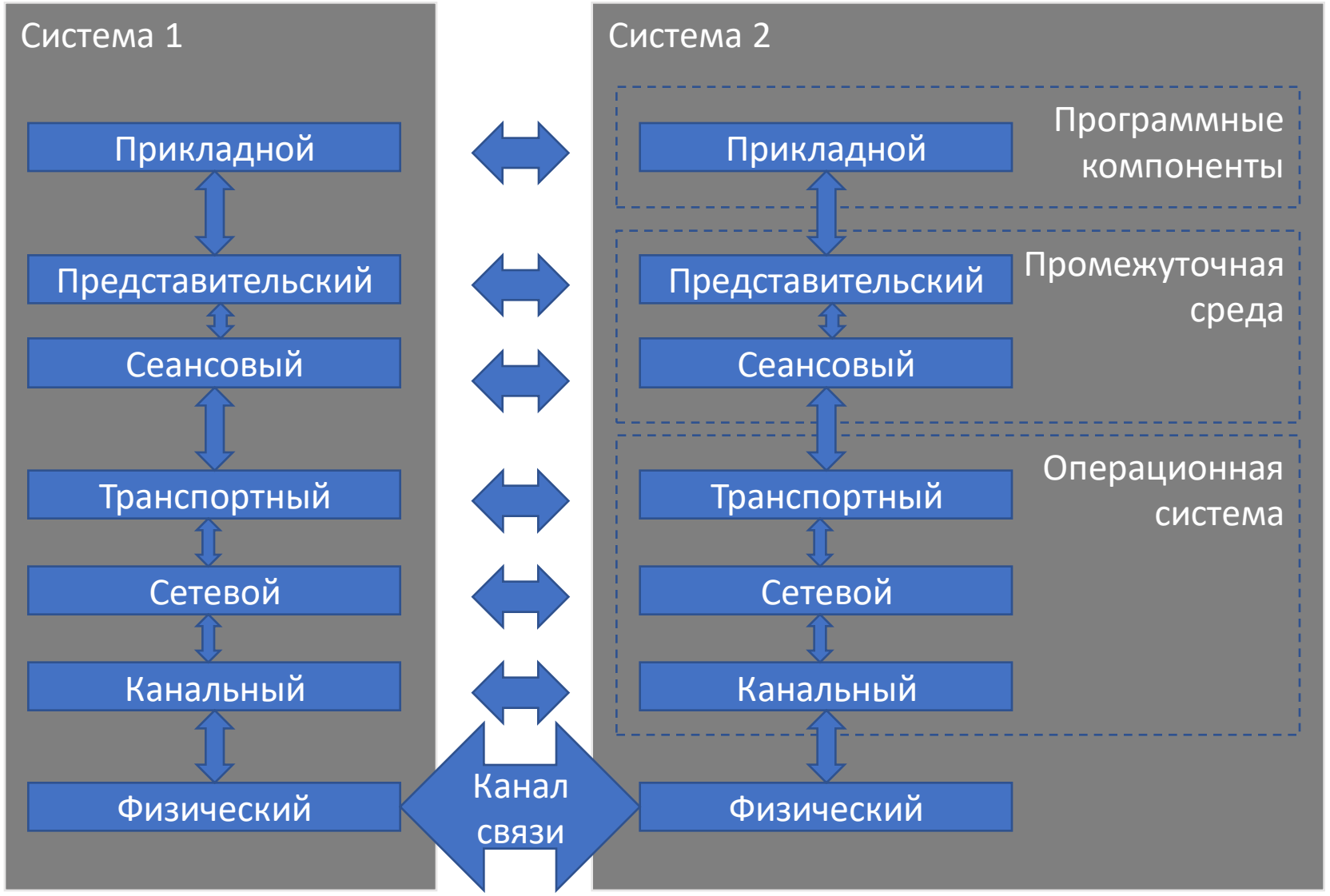


Понятие промежуточной среды

- Для достижения этой цели промежуточная среда должна обеспечить сервисы для взаимодействия компонент распределенной системы:
  - обеспечение единого и независимого от операционной системы механизма использования одними программными компонентами сервисов других компонент
  - обеспечение безопасности распределенной системы: аутентификация и авторизация всех пользователей сервисов компоненты и защита передаваемой между компонентами информации от искажения и чтения третьими сторонами
  - обеспечение целостности данных: управление транзакциями, распределенными между удаленными компонентами системами
  - балансировка нагрузки на серверы с программными компонентами
  - обнаружение удаленных компонент

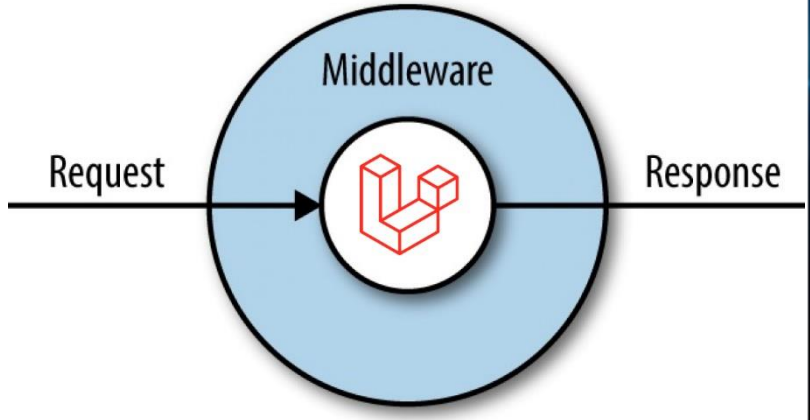


Место промежуточной среды в модели OSI



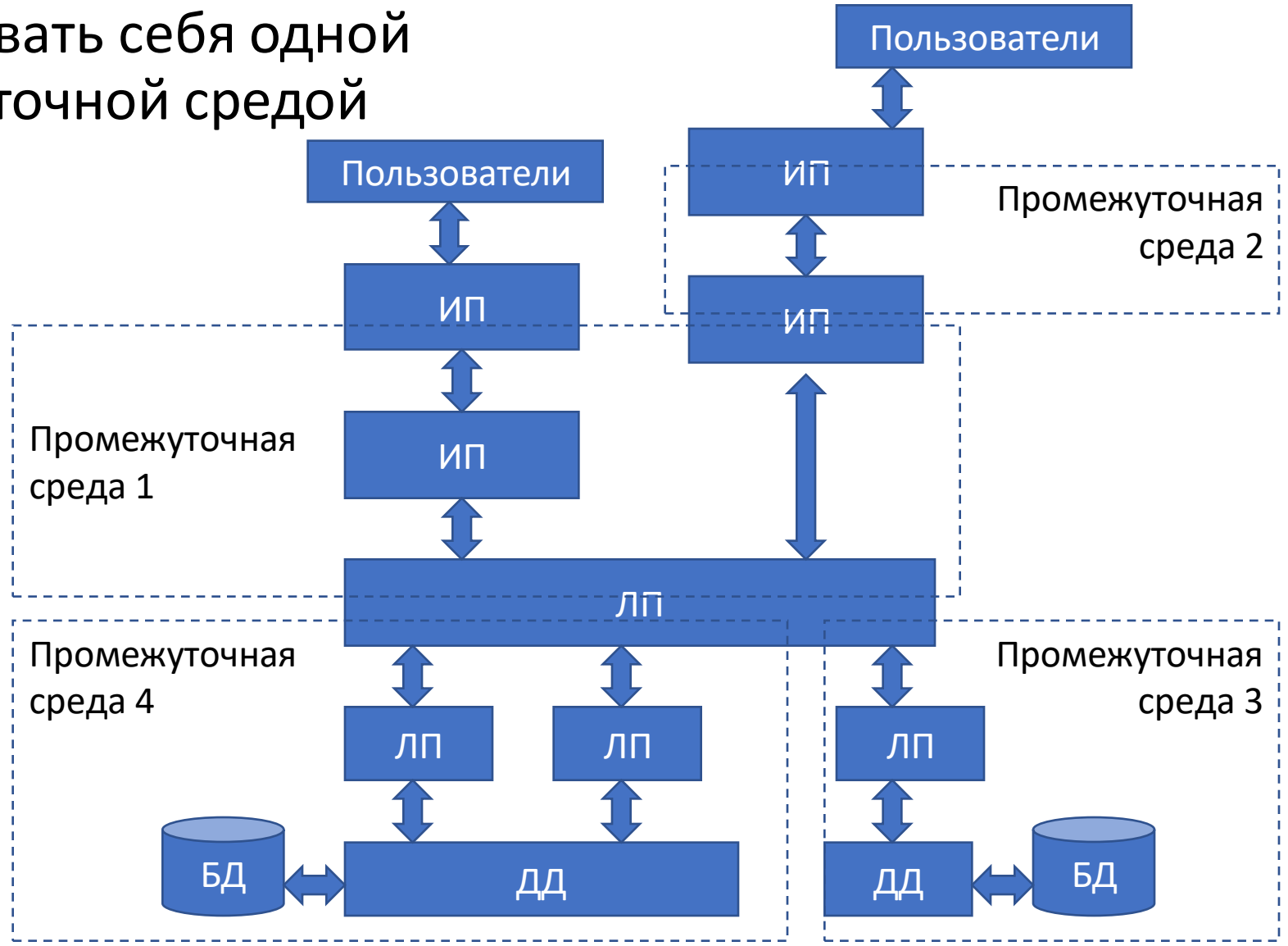
Гетерогенная распределенная система

- В пределах одной распределенной системы может использоваться несколько видов промежуточных сред
- При хорошем подходе к проектированию системы каждая распределенная ее компонента предоставляет свои сервисы посредством единственной промежуточной среды, и использует сервисы других компонент посредством так же единственной промежуточной среды, однако эти среды могут быть различными.



Гетерогенная распределенная система

Распределенная система не обязана ограничивать себя одной промежуточной средой



Примеры распределенных систем

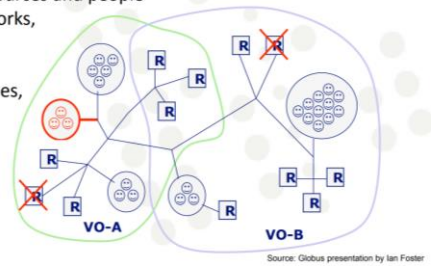
Grid example: e-science - CERN LHC

- Largest machine built by humans: particle accelerator and collider with a circumference of 27 kilometers
  - Said to generate 10 Petabytes (10^7 Gigabytes) of information per year
  - This information must be processed and stored somewhere
- Beyond the scope of a single institution to manage this problem
  - Projects: LCG (LHC Computing Grid), EGEE (Enabling Grids for E-sciencE)

Source: Globus presentation by Ian Foster

Grid structure: Virtual Organizations, Virtual Teams

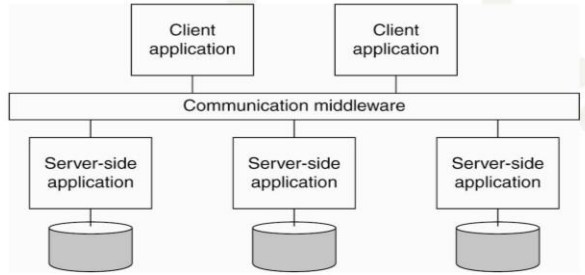
- Distributed resources and people
- Linked by networks, crossing admin domains
- Sharing resources, common goals
- Dynamic



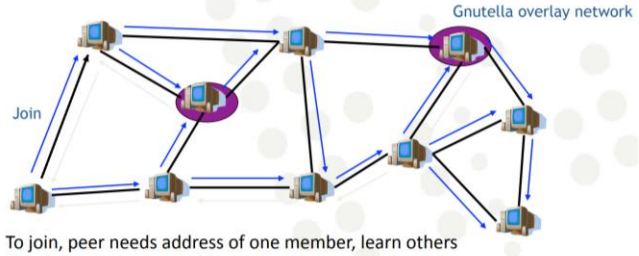
Source: Globus presentation by Ian Foster

Example DS: Enterprise Application Integration (EAI)

Distributed Information System with full integration of business data and business processes



P2P Systems: unstructured

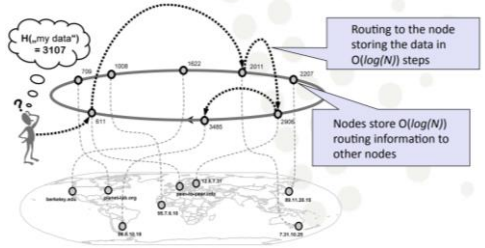


- To join, peer needs address of one member, learn others
- Queries are sent to neighbors
- Neighbors forward queries to their neighbors (flooding)
- Replies routed back via query path to querying peer

Cloud Computing

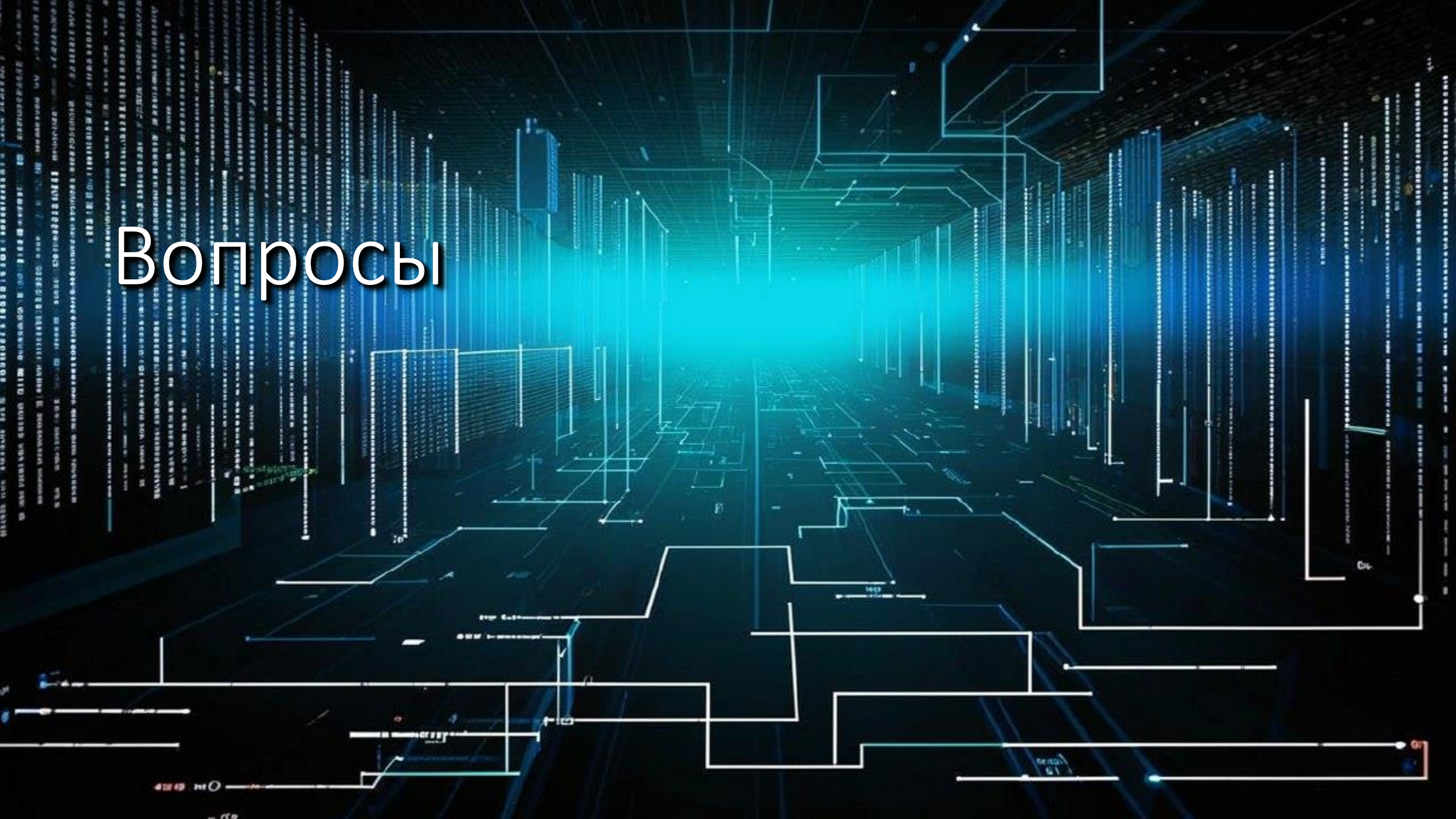
- New “paradigm”
  - Often regarded as Grid computing with a business case: Grid computing minus VOs, VTs
  - Cloud providers sell cloud access as a service e.g. Amazon
- At least two quite different “visions”
  1. Large server farms: high-speed computing and large & secure data storage on demand
  2. Dumb terminals: all applications become services

P2P Systems: structured Distributed Hash Tables (DHT)



Distributed Computing Systems

- History: parallel processing at a growing scale
  - Parallel CPU architectures
  - Multiprocessor machines
  - Clusters
  - (“Massively Distributed“) computers on the Internet: Grid
    - transparency again: “power Grid“ metaphor
    - Most common underlying middleware: Globus Toolkit (now entirely based on Web Services)


Вопросы