Лекция 1.2 «Архитектура инфокоммуникационных сетей»

Учебные вопросы:

1. Архитектура инфокоммуникационных сетей.
2. SOA.
3. Назначение, функции и виды middleware.

1. **Архитектура** – это концепция, определяющая взаимосвязь, структуру и функции взаимодействия рабочих станций в сети (функциональная и физическая организацию технических и программных средств сети).

Архитектура сети определяет основные элементы сети, характеризует общую логическую организацию, техническое и программное обеспечение, описывает методы кодирования. Архитектура также определяет принципы функционирования и интерфейс пользователя.

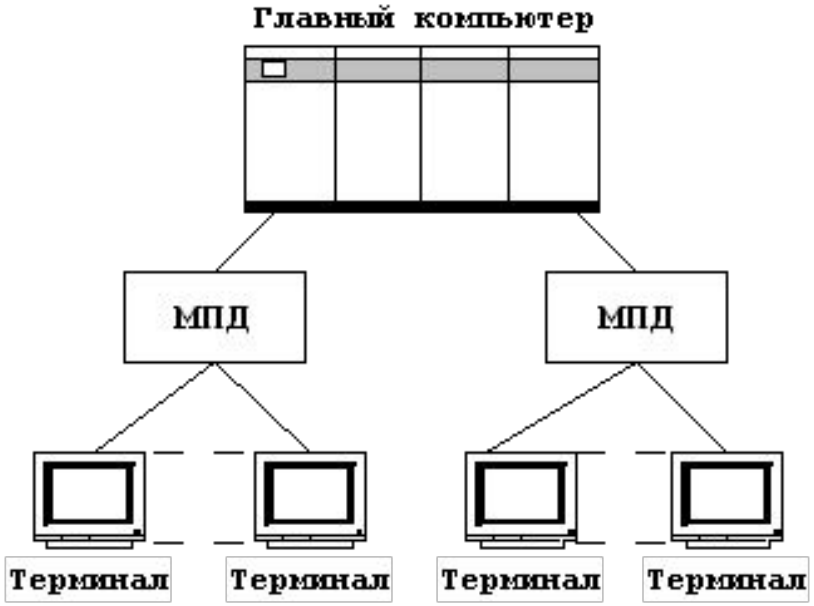
В зависимости от того, как распределены функции между компьютерами сети:

- сеть на основе одноранговых узлов – **одноранговая сеть**;
- сеть на основе клиентов и серверов – **сеть с выделенными серверами**;
- сеть, включающая узлы всех типов – **гибридная сеть**.

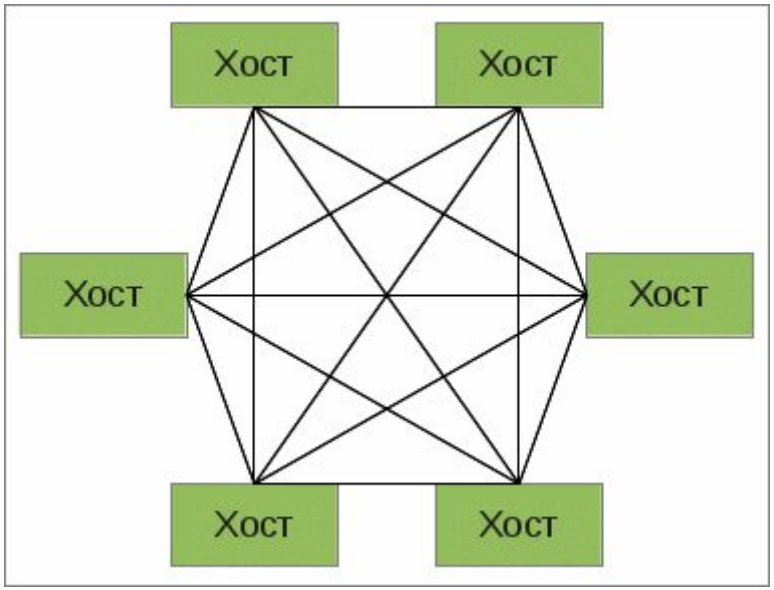
Рассмотрим три вида архитектур (**по организации управления сетью**) :

- архитектура терминал-главный компьютер (выделенный сервер);
- одноранговая архитектура;
- архитектура клиент-сервер.

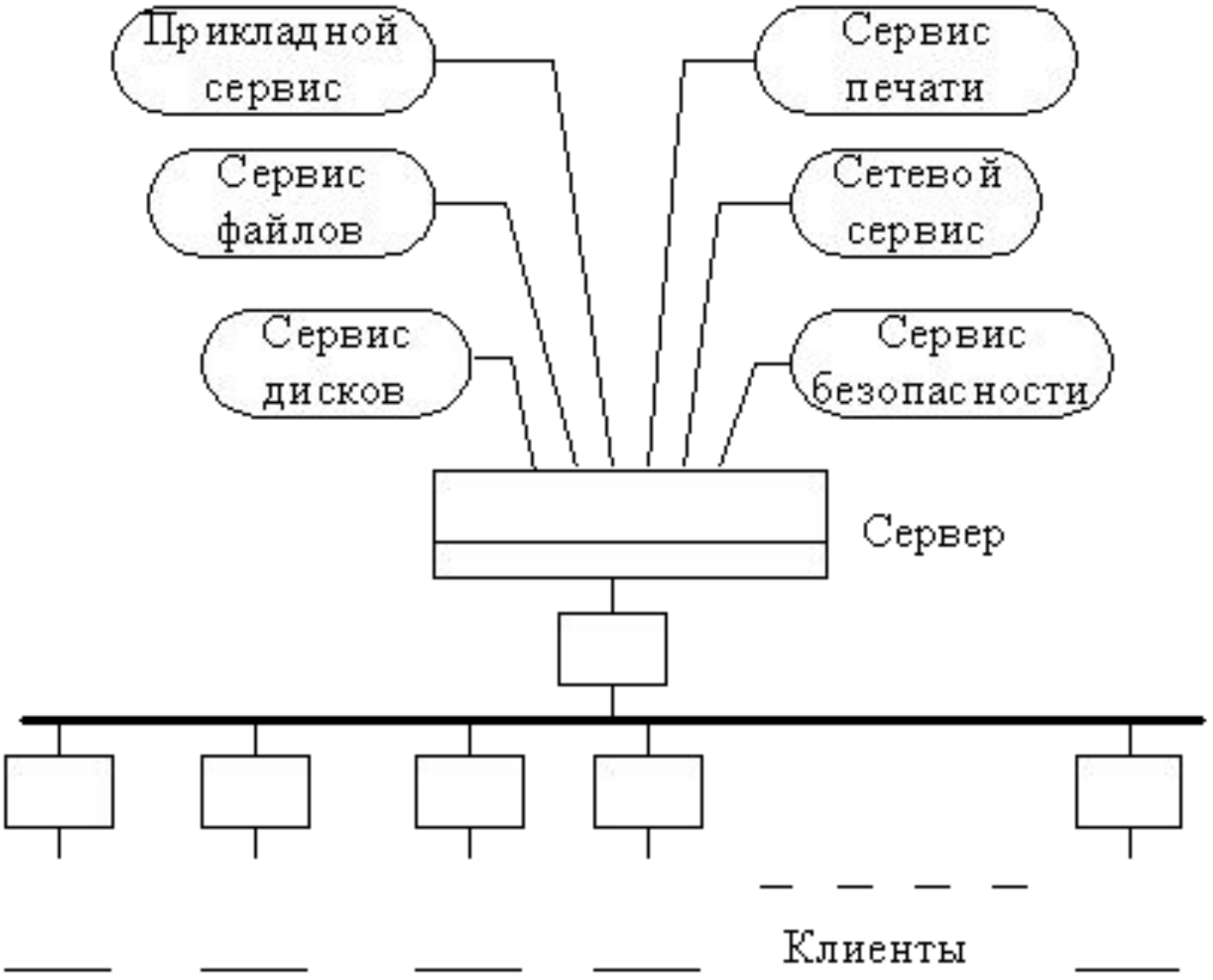
- Архитектура терминал-главный компьютер (выделенный сервер) – это концепция информационной сети, в которой вся обработка данных осуществляется одним или группой главных компьютеров.



- Одноранговая архитектура (peer-to-peer architecture) – это концепция информационной сети, в которой ее ресурсы рассредоточены по всем системам (все системы равноправны).



- Архитектура клиент-сервер (client-server architecture) – это концепция в которой основная часть ресурсов сосредоточена в серверах, обслуживающих своих клиентов.


Прикладной сервис
Сервис печати
Сервис файлов
Сетевой сервис
Сервис дисков
Сервис безопасности
Сервер
Клиенты

Компоненты сетевого сервиса:

- информационные ресурсы;
- программа-клиент;
- набор программ-серверов, осуществляющих обработку запросов клиентского программного обеспечения и управляющих информационными объектами;
- набор протоколов взаимодействия клиентского программного обеспечения с серверами.

Наиболее известные сервисы Интернет:

- электронная почта (E-mail);
- телеконференции, или группы новостей (Usenet);
- сервис FTP – система файловых архивов, обеспечивающая хранение и пересылку файлов различных типов;
- сервис Telnet, предназначенный для управления удаленными компьютерами в терминальном режиме;
- World Wide Web (WWW, W3) – гипертекстовая (гипермедиа) система;
- сервис DNS;
- сервис IRC (chat);

Стандартные номера портов для основных сервисов

Компонент службы	Номер порта	Транспортные протоколы
Электронная почта		
SMTP-сервер	25	TCP
POP3-сервер	110	TCP
IMAP-сервер	143	TCP
Телеконференции		
NNTP-сервер	119	TCP
FTP		
FTP-сервер	20, 21	TCP
Telnet		
Telnet-сервер	23	TCP
WWW		
HTTP-сервер	80	TCP
DNS		
DNS-сервер	53	TCP, UDP

Архитектурные решения доступа к данным в компьютерных сетях:

- централизованная архитектура
- архитектура "файл-сервер"
- архитектура "клиент-сервер"

Практические реализации **архитектуры «клиент-сервер»** называются **клиент-серверными технологиями**.

Клиент-серверные технологии

(сервисы и серверы):

- Web-серверы
- Серверы приложений
- Серверы баз данных
- Файл-серверы
- Прокси-сервер
- Файрволы (брандмауэры)
- Почтовые серверы
- Серверы удаленного доступа (RAS) *(Remote Access Service)*

«Тонкий» клиент

обеспечивает запуск web-браузера, в окне которого и осуществляются все действия.

«Толстый» клиент

станции, работающие под управлением собственной ОС и имеющие необходимый набор ПО. К сетевым серверам «толстые» клиенты обращаются в основном за дополнительными услугами (например к корпоративной БД).
Так же под «толстым» клиентом подразумевается и клиентское сетевое приложение.

«Rich»-client –

компромисс между «толстым» и «тонким» клиентом.

«rich»-клиент представляет

графический интерфейс, описываемый уже средствами XML (как «тонкий» клиент) и

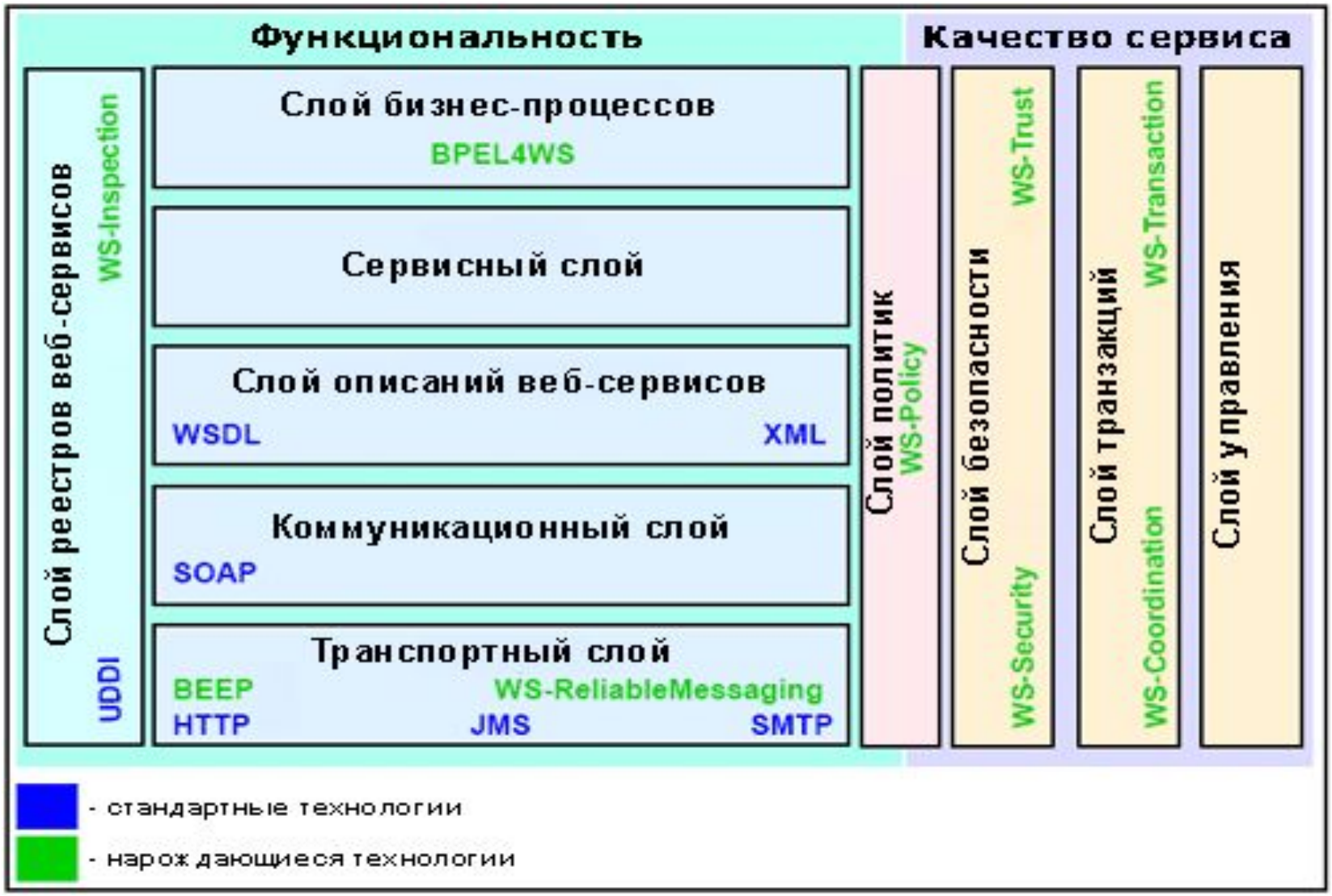
функциональность «толстых » клиентов (интерфейс drag-and-drop, множественные окна, выпадающие меню и т.п.)

2. Компонентная модель, состоящая из отдельных функциональных модулей приложений, называемых сервисами, имеющих определенные интерфейсы и механизм взаимодействия между собой, **называется сервисно-ориентированной архитектурой** (Service-Oriented Architecture, SOA).

- Архитектура приложений, в рамках которой все *функции приложения* являются независимыми сервисами с четко определенными интерфейсами, которые можно вызывать в нужном порядке, называется **SOA**.

Web-сервис это программный интерфейс, который описывает набор операций, которые могут быть вызваны удаленно по сети посредством стандартизированных XML сообщений.

Для реализации сервисно-ориентированных архитектур с помощью веб-сервисов применяется совокупность технологий, образующих так называемый *стек технологий веб-сервисов*.


Функциональность
Качество сервиса
Слой реестров веб-сервисов
WS-Inspection
UDDI
Слой бизнес-процессов
BPEL4WS
Сервисный слой
Слой описаний веб-сервисов
WSDL
XML
Коммуникационный слой
SOAP
Транспортный слой
BEEP
WS-ReliableMessaging
HTTP
JMS
SMTP
Слой политик
WS-Policy
Слой безопасности
WS-Trust
WS-Security
Слой транзакций
WS-Transaction
WS-Coordination
Слой управления
- стандартные технологии
- нарождающиеся технологии

Принципы взаимодействия веб-сервисов в рамках сервисно-ориентированной архитектуры.

В настоящее время технологический фундамент веб-сервисов образуется следующими технологиями:

- eXtensible Markup Language (XML);
- Simple Object Access Protocol (SOAP);
- Universal Description, Discovery and Integration (UDDI);
- Web Services Description Language (WSDL).

Язык XML определяет формат данных, используемый для обмена информацией между потребителем сервиса и самим сервисом. XML - это синтаксис для описания структур данных, (мета-язык, позволяющий осуществлять обмен данными с помощью стандартных методов для кодирования и форматирования информации). В отличие от HTML, XML позволяет не только описывать структуру информации, но и ее контекст.

SOAP (Simple Object Access Protocol, *или Services-Oriented Architecture Protocol*) — это основанный на языке XML стандарт для взаимодействия между сервисами и их потребителями.

Протокол SOAP базируется на сообщениях, которые разделяются на два типа: запросы (вызов метода удаленного объекта) и ответы (результат работы удаленного метода).

SOAP поддерживает два механизма доступа : SOAP RPC и SOAP Message.

SOAP RPC - протокол «запрос-ответ» и базируется на объекте Call. Этот объект (и некоторые низкоуровневые методы для создания и отсылки сообщений) используется для синхронного удаленного вызова методов Web-сервисов.

SOAP Message — это протокол для отсылки и обработки SOAP-сообщений, который может использоваться для асинхронных коммуникаций и подразумевает немедленный или отложенный ответ на запрос.

Интерфейс для доступа к Web-сервису описывается на основанном на языке XML языке Web Services Description Language (**WSDL**) и содержит всю информацию, необходимую для доступа к данному сервису.

WSDL-документы используются средствами разработки для генерации прокси-кода, который выполняет роль переходника между высокоуровневым кодом потребителя Web-сервиса и низкоуровневой реализацией отсылки SOAP-сообщений для вызова методов Web-сервиса и получения результатов работы этих методов.

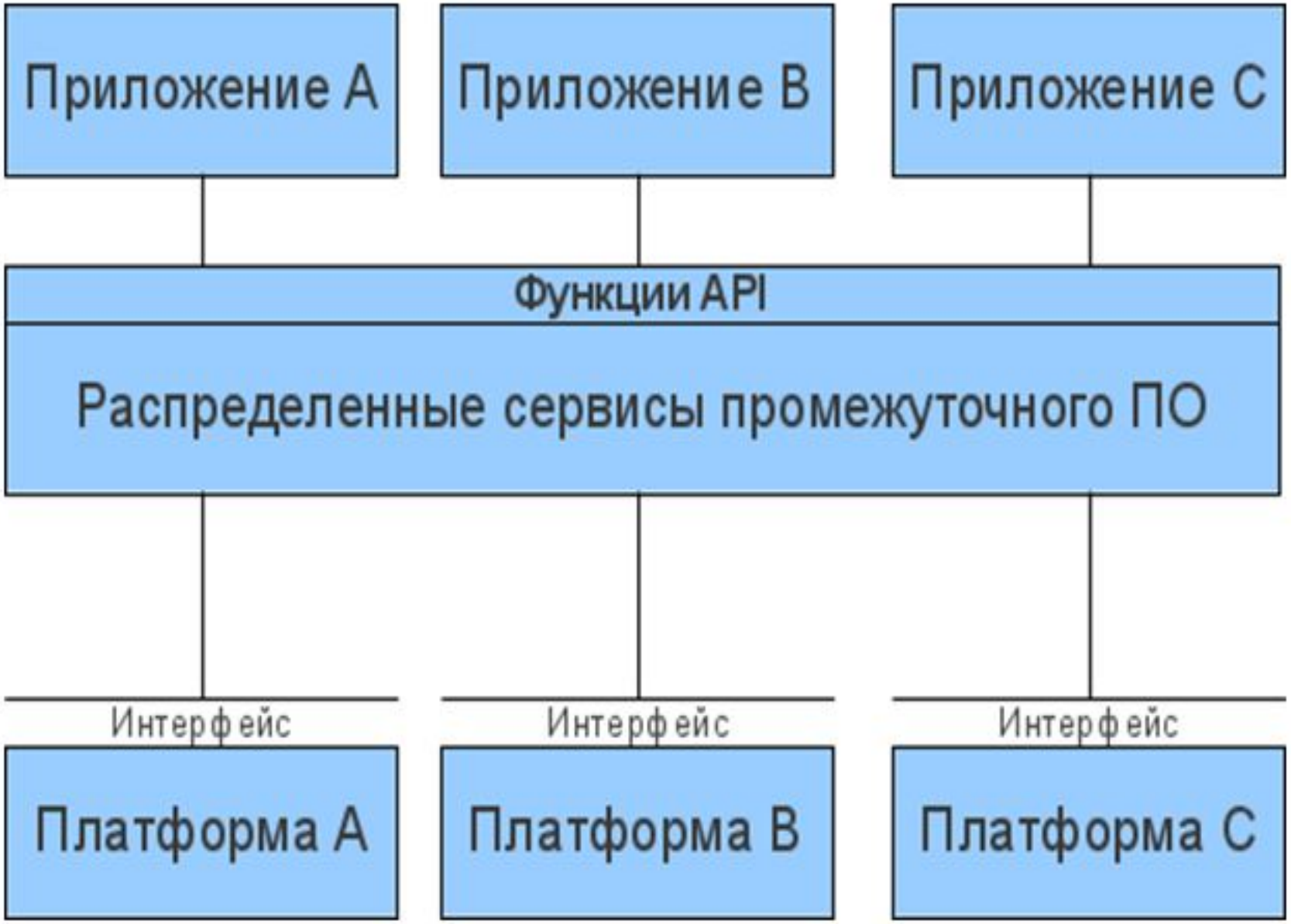
UDDI - спецификация, включающая набор XML-файлов и ассоциированные схемы, которые содержат описания предоставляемых Web-сервисами услуг. В качестве примера UDDI-реестров можно привести следующие:

- Реестр фирмы IBM - UDDI Business Registry;
- UDDI-реестр фирмы Microsoft;
- UDDI-реестр фирмы Hewlett-Packard.

Вопросы

1. В чем отличия между понятиями «клиент-серверная архитектура» и «клиент-серверная технология»?
2. Опишите технологию клиент-сервер.

3. **Промежуточное программное обеспечение** (middleware) — это класс программного обеспечения, предназначенного для объединения компонентов распределенного клиент-серверного приложения или целых сетевых приложений в единую информационную систему.


Приложение A
Приложение B
Приложение C
Функции API
Распределенные сервисы промежуточного ПО
Интерфейс
Интерфейс
Интерфейс
Платформа A
Платформа B
Платформа C

Функции middleware

- прозрачный доступ к сетевым сервисам и приложениям;
- независимость от других сетевых сервисов;
- высокая надежность и доступность.

Виды промежуточного ПО:

- ***Программное обеспечение для межпрограммного взаимодействия (основными технологиями являются***: RPC, MOM, TPM и ORB).
- ***Программное обеспечение доступа к базам данных***.

Концепция **вызова удаленных процедур** (*remote procedure call — RPC*) была разработана и реализована в компанией XEROX еще начале 80-х годов XX века.

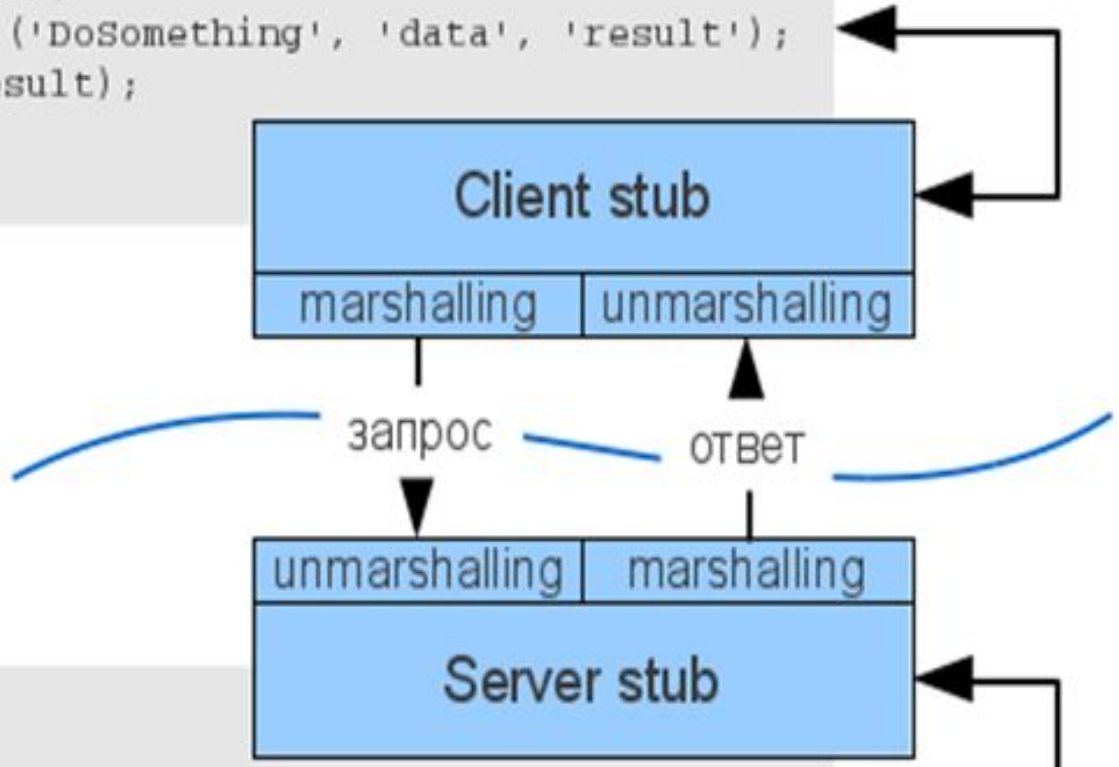
Общий смысл RPC : программа может выполнять не только собственные (скомпилированные) процедуры и функции, но и обращаться к процедурам удаленного сервера.

Клиент

```
...

Init(data);
RPC_call('DoSomething', 'data', 'result');
print(result);

...
```



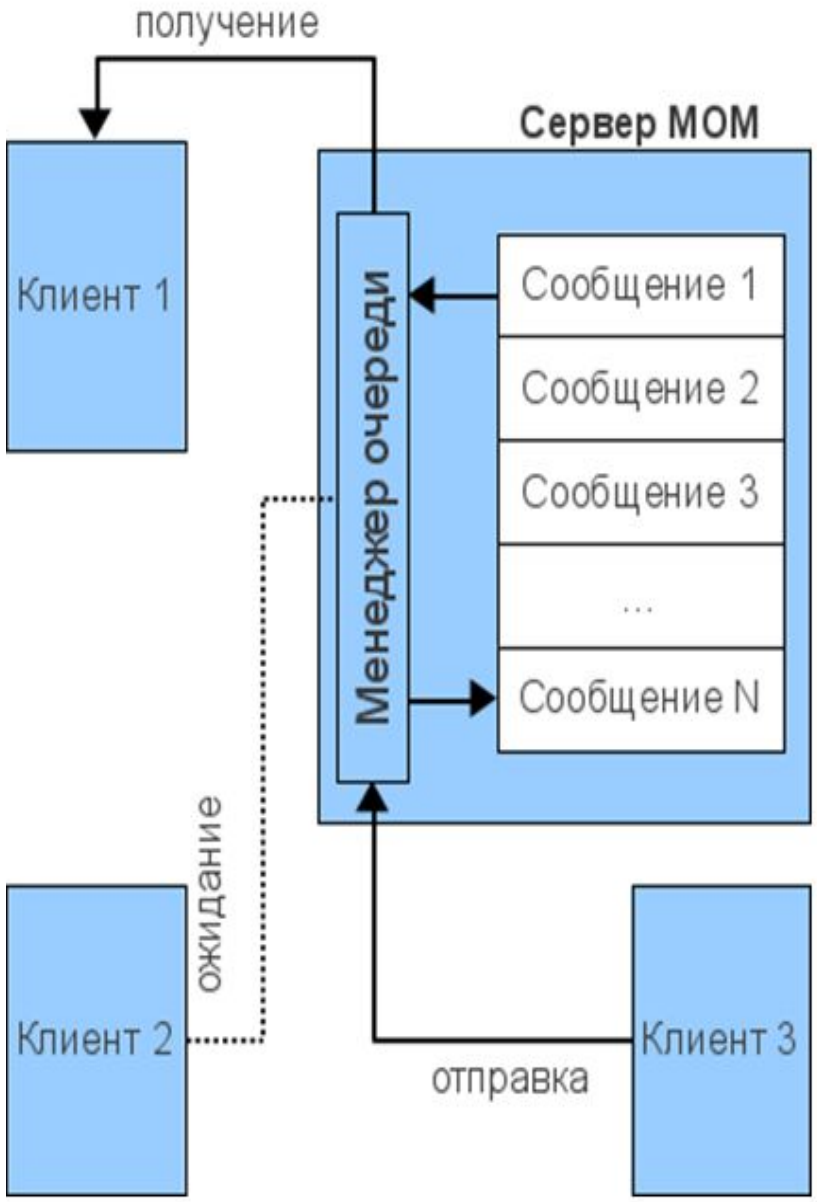

```
...

DoSomething (data) {
  ...
  return result;
  };
...
```

Сервер

Сервисы обработки сообщений (*MOM — message-oriented middleware*) —это системы, как правило асинхронные, в которых взаимодействие между клиентом и сервером основано на обмене сообщениями.

Для передачи сообщений используются байт-ориентированные протоколы, такие как HTTP, POP/SMTP и т.п.


получение
Сервер МОМ
Клиент 1
Менеджер очереди
Сообщение 1
Сообщение 2
Сообщение 3
...
Сообщение N
ожидание
Клиент 2
Клиент 3
отправка

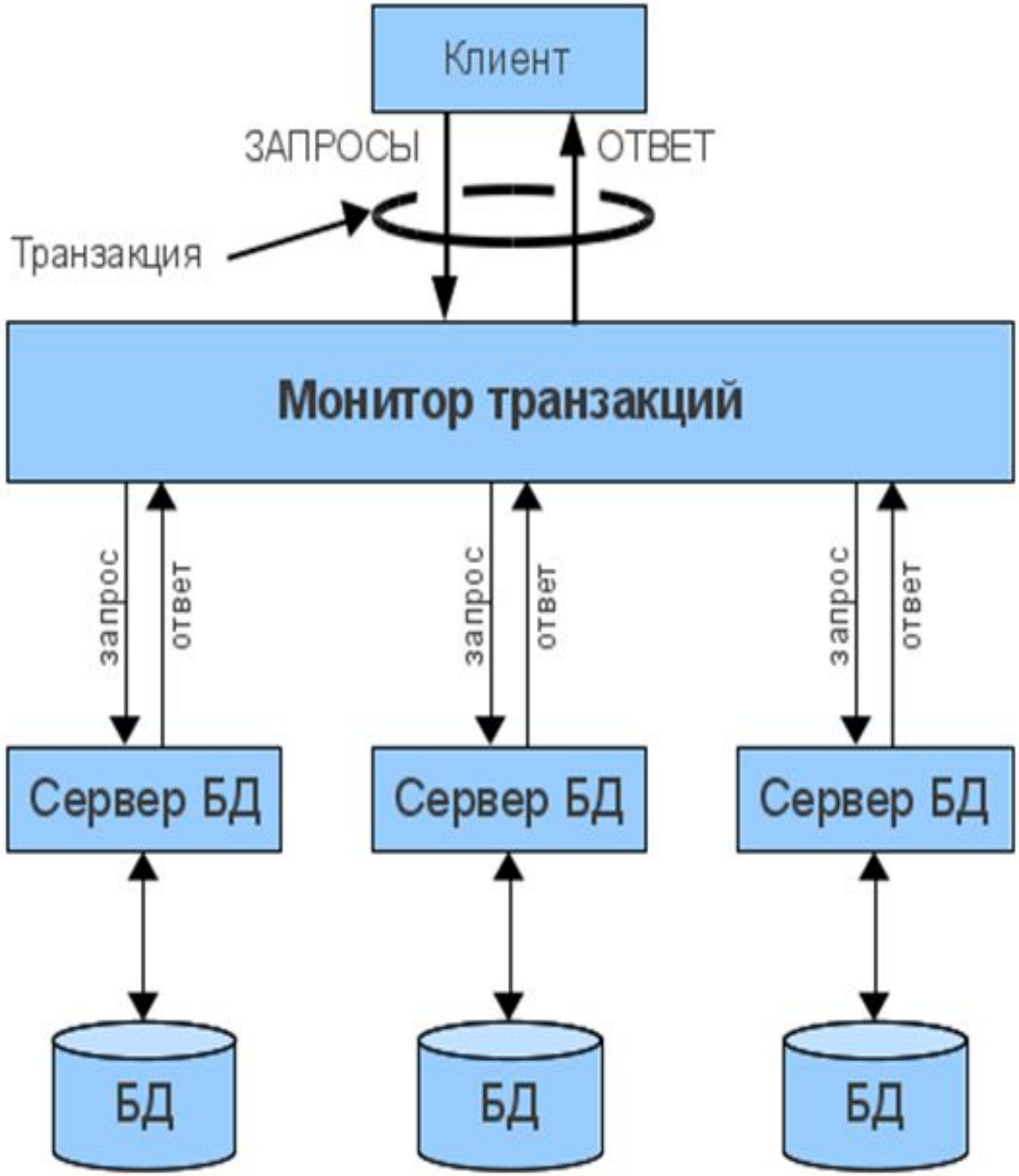
Запросы сервисов ставятся в *очередь сообщений,* *где м.б.гарантирована* **:**

- надежная доставка сообщений;
- гарантированная доставка сообщений;
- застрахованная доставка сообщений.

Другие сервисы МОМ:

- ***Непосредственная передача сообщений*** (логические сетевые соединения).
- На основе ***подписки*** (по принципу, почтовой рассылки).

- **Мониторы обработки транзакций** (*Transaction Processing monitors*, TP-monitors) — это промежуточное программное обеспечение, обеспечивающее контроль передачи данных от клиента при работе с распределенными базами данных.


Клиент
ЗАПРОСЫ
ОТВЕТ
Транзакция
Монитор транзакций
запрос
ответ
запрос
ответ
запрос
ответ
Сервер БД
Сервер БД
Сервер БД
БД
БД
БД

Промежуточное ПО доступа к базам данных

- *Собственное промежуточное ПО СУБД* — это встроенные механизмы доступа для конкретного сервера баз данных.
- *Основное промежуточное ПО баз данных* — к этому типу относится, например, интерфейс Open Database Connectivity (ODBC), который позволяет программам «общаться» на разных диалектах SQL через общие интерфейсы.

Собственное промежуточное ПО СУБД:

- поддержка стандартов языка SQL;
- встроенные средства СУБД, позволяющие выполнять импорт или экспорт данных в сторонние форматы (например, CSV или XML).

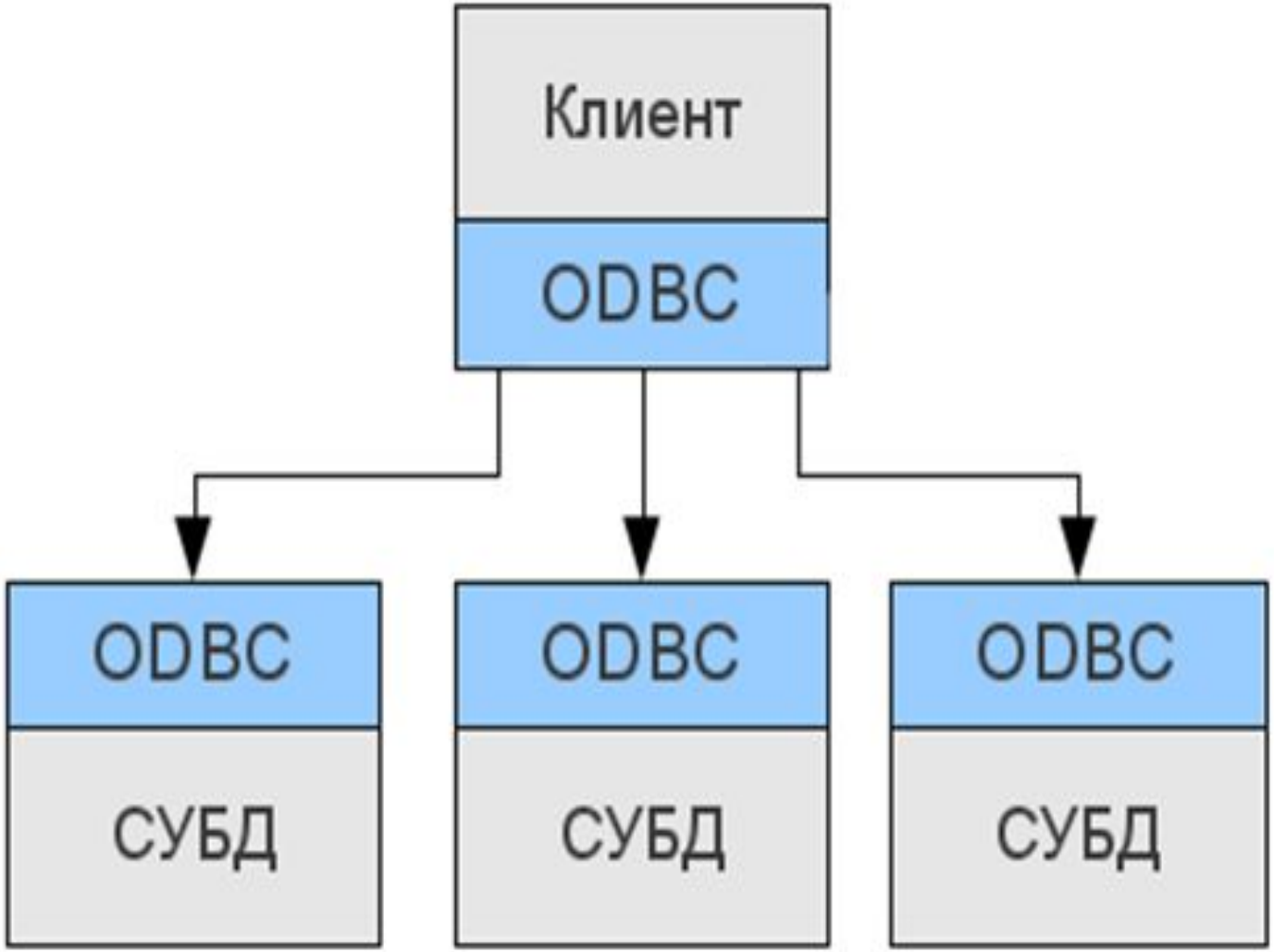
Основное промежуточное ПО доступа к БД:

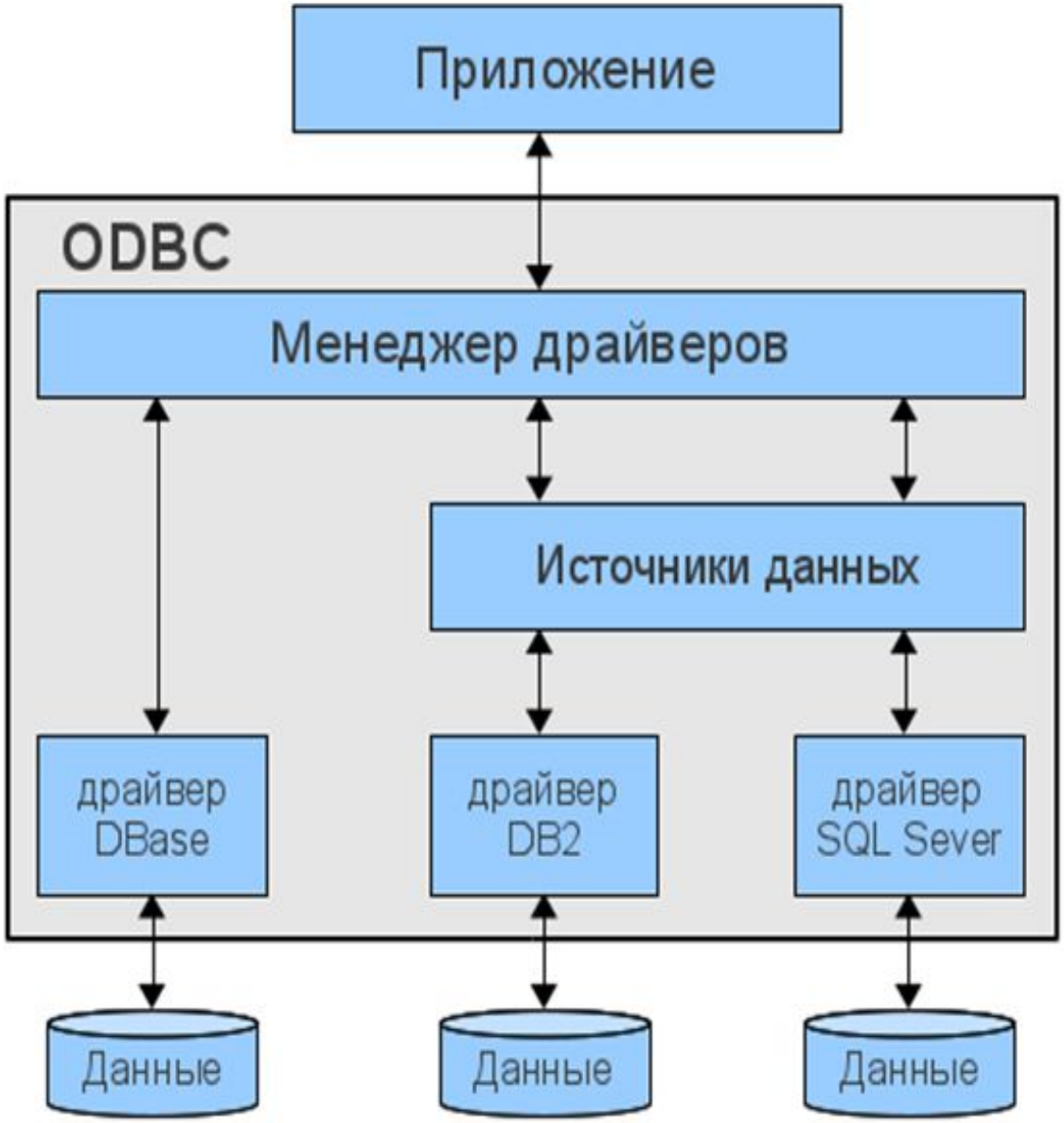
- средства, специально разработанные для обращения к базам данных.

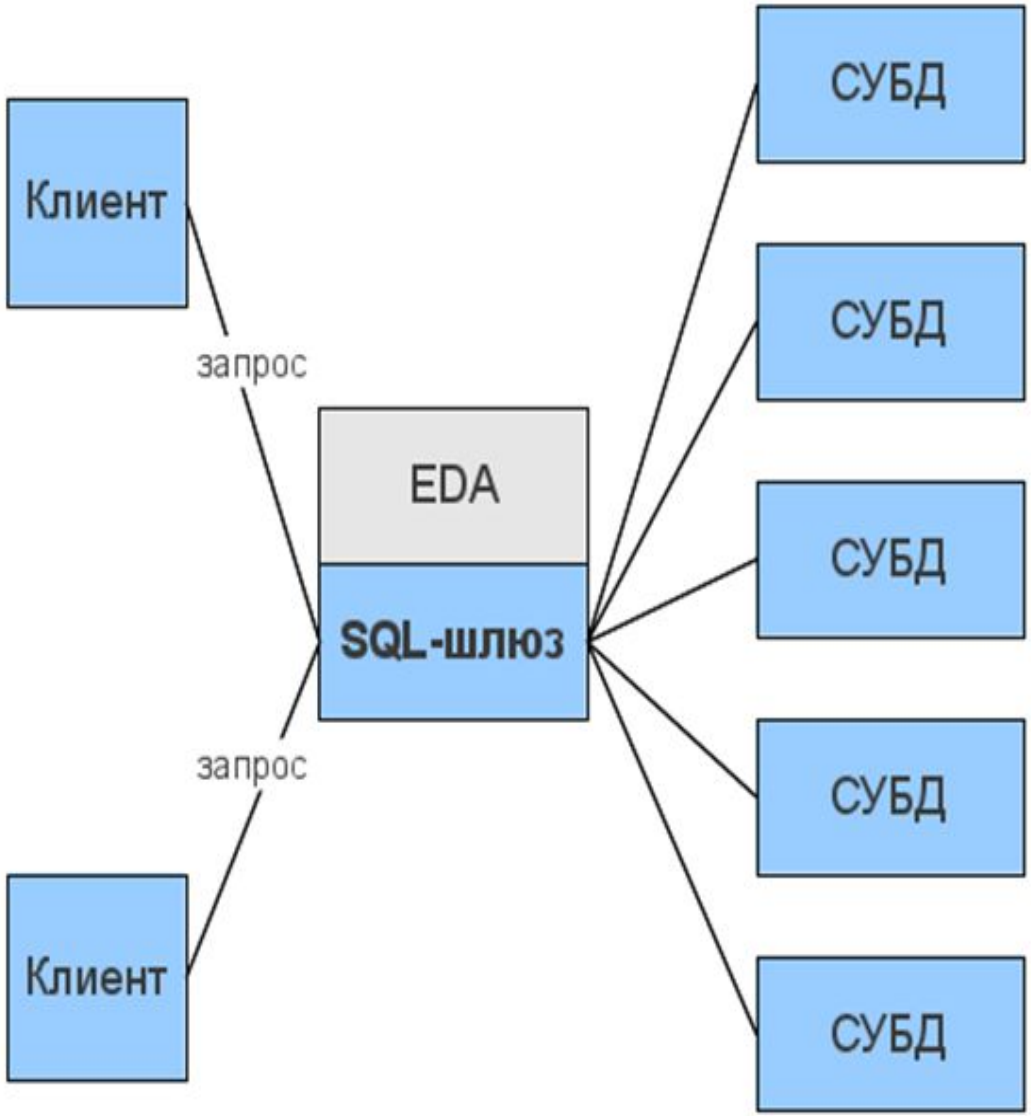
  - *технологические решения* (например, ODBC и JDBC),

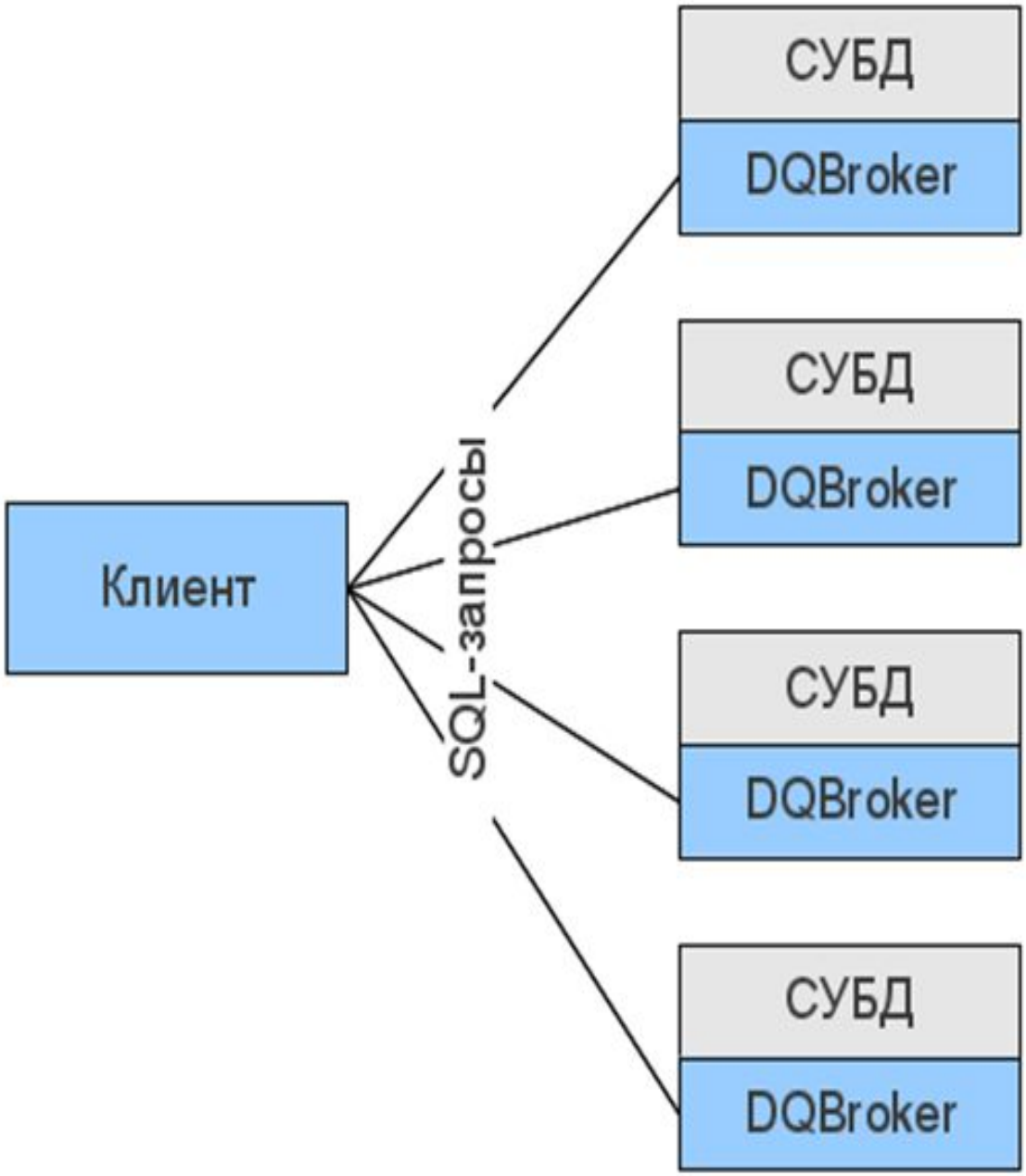
  - *концептуальные* (например, EDA или DQB).

- **ODBC** (*Open DataBase Connectivity*) —интерфейс доступа к базам данных (Microsoft).
- **JDBC** (*Java DataBase Connectivity*) — это прикладной программный интерфейс для обращения к базам данных из Java-приложений.
- **EDA** (*Event-Driven Architecture*, событийно-управляемая архитектура) — концепция управления корпоративной информационной системой на основе событий
- **DQB** (*Distributed Query Broker*, брокер распределенных запросов) — децентрализованное (в отличие от EDA) решение для доступа к БД на основе ORB.

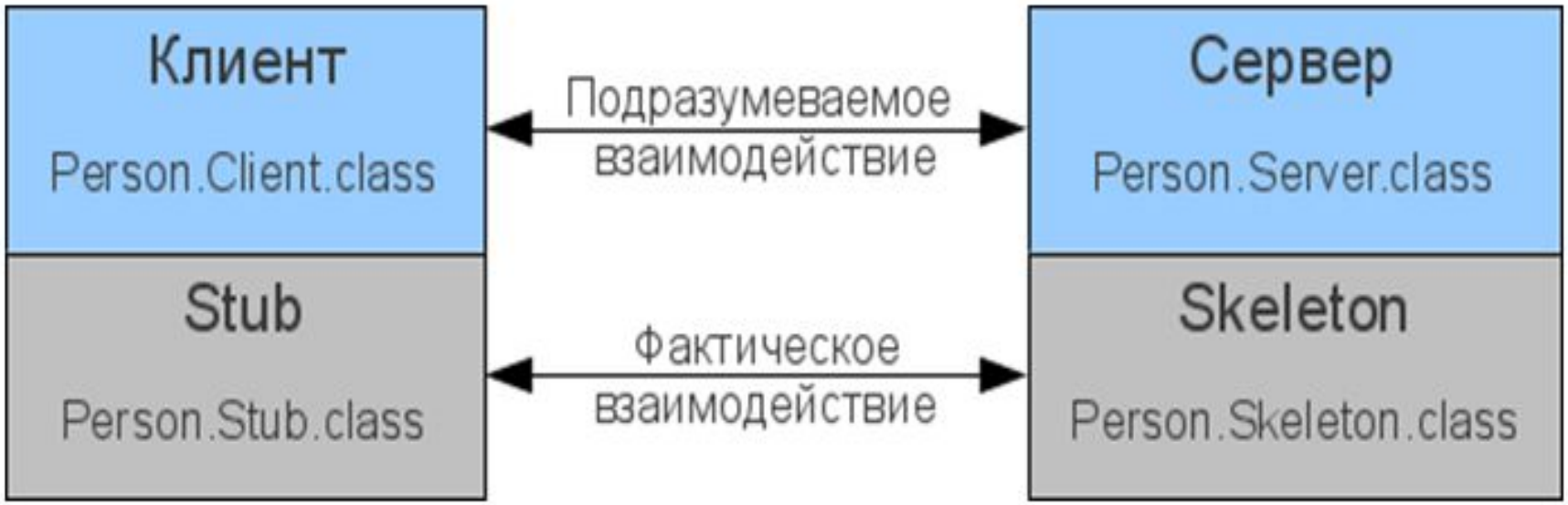

Клиент
ODBC
ODBC
ODBC
ODBC
СУБД
СУБД
СУБД


Приложение
ODBC
Менеджер драйверов
Источники данных
драйвер DBase
драйвер DB2
драйвер SQL Sever
Данные
Данные
Данные


Клиент
запрос
Клиент
запрос
EDA
SQL-шлюз
СУБД
СУБД
СУБД
СУБД
СУБД


Клиент
SQL-запросы
СУБД
DQBroker
СУБД
DQBroker
СУБД
DQBroker
СУБД
DQBroker

- **Распределенные объектные системы** (Distributed object systems) — это промежуточное программное обеспечение, реализованное в виде взаимодействующих друг с другом программных объектов.


Клиент
Person.Client.class
Подразумеваемое
взаимодействие
Сервер
Person.Server.class
Stub
Person.Stub.class
Фактическое
взаимодействие
Skeleton
Person.Skeleton.class

Архитектура **распределенных объектных систем** стандартизована и наиболее распространены спецификации CORBA, COM/DCOM и EJB.

- **CORBA** (*Common Object Request Broker Architecture*, типовая архитектура брокера объектных запросов) — открытый стандарт, определяет интерфейсы между сетевыми объектами, позволяющие им работать совместно.
- Microsoft **COM** (*Component Object Model*, компонентная объектная модель) — это семейство технологий, предназначенных для организации взаимодействия Windows-приложений.
- **EJB** (Enterprise JavaBeans) — технология, разработанная Sun Microsystems для платформы Java.

ЗАКЛЮЧЕНИЕ

Web-сервисы - это развивающаяся технология, и многие стандарты, которые мы здесь обсудили, будут со временем меняться. Важно отметить, что ни одна технология для обеспечения Web-сервисов не должна быть монополизирована какой-либо одной компанией - только в этом случае можно гарантировать успех Web-сервисов.