# БЕЗОПАСНОСТЬ СИСТЕМ БАЗ ДАННЫХ

## 8. Безопасность распределенных систем

# Принципы построения распределенных КС и БД

- Неуклонная тенденция к интеграции информационных ресурсов в компьютерные сети разных уровней в сочетании с утвердившейся концепцией БД закономерно привели к появлению систем распределенных баз данных (РБД, англ. Distributed DataBase - DDB).
- РБД включает в себя фрагменты из нескольких БД, располагающихся на различных узлах компьютерной сети, и, возможно, управляющихся различными СУБД. Для пользователей и прикладных программ РБД выглядит как обычная локальная БД. В этом смысле слово термин «распределенная» отражает способ организации БД, но не внешнюю ее характеристику.

# Принципы построения распределенных КС и БД

- Основной целью системы РБД является обеспечение управляемого доступа и независимого обращения к данным, распределенным в компьютерной сети.
- Основная задача СУРБД состоит в обеспечении средств интеграции локальных БД, располагающихся в узлах компьютерной сети, с тем, чтобы пользователь, работающий в любом узле сети, имел доступ ко всем этим БД как к единой БД.
- При этом должны обеспечиваться:
  - простота использования системы;
  - возможности автономного функционирования при нарушениях связности сети или при административных потребностях;
  - высокая степень эффективности.

# Принципы построения распределенных КС и БД

- В основе РБД лежат две основные идеи:
  - много организационно и физически распределенных пользователей могут одновременно работать с общими данными (общей БД);
  - логически и физически распределенные данные (таблицы, записи и даже поля) составляют и образуют единое взаимосогласованное целое – общую БД.
- Три основных принципа построения РБД:
  - прозрачность (невидимость) расположения данных для пользователя (должна выглядеть как обычная БД);
  - изолированность пользователей друг от друга;
  - синхронизация и согласованность состояния данных в любой момент времени.

# Принципы построения распределенных КС и БД

- 12 дополнительных принципов построения РБД:
  - локальная автономия;
  - отсутствие центральной ВУ;
  - непрерывность функционирования;
  - прозрачность расположения данных;
  - прозрачная фрагментация данных;
  - прозрачное реплицирование данных;
  - распределенная обработка запросов;
  - распределенная обработка транзакций ;
  - независимость от оборудования;
  - независимость от типа ОС;
  - независимость от коммуникационной среды;
  - независимость от типа СУБД.

# Проблемы РБД

- Основные проблемы РБД:
  - проблема размещения системного каталога РБД;
  - проблема поддержания копий данных в нескольких узлах сети;
  - проблема обеспечения изолированности пользователей при работе с общими данными;
  - проблема необходимости распараллеливания выполнения запроса к БД в сочетании с фрагментацией объектов РБД и расширением пространства поиска вариантов выполнения запросов.

# Принципы решения проблем РБД

- Для решения многочисленных проблем РБД при их практической реализации идут по пути умышленных отступлений от сформулированных К. Дейтом принципов.
- Это приводит к тому, что в зависимости от того, какой из принципов «приносится в жертву», возникает свое самостоятельное направление в технологиях РБД.
- В настоящее время наиболее обозначились три таких технологии:
  - ***технологии «клиент-сервер»;***
  - ***технологии объектного связывания данных;***
  - ***технологии реплицирования данных.***

# Технологии «клиент-сервер»

- В технологиях «клиент-сервер» отступают от одного из главных принципов РБД – отсутствие центральной ВУ. в В основе клиент-серверных технологий лежат две основные идеи:
  - общие для всех пользователей данные располагаются на одном или нескольких серверах;
  - много пользователей (клиентов) на различных ВУ совместно (параллельно и одновременно) обрабатывают общие данные.
- Т.е. технологии «клиент-сервер» распределены только в отношении пользователей. Поэтому их чаще не относят к «настоящим» РБД, а выделяют в отдельный самостоятельный класс многопользовательских систем.

# Технологии «клиент-сервер»

- Система К-С имеет две части – клиентскую и серверную, которые могут выполняться в разных узлах сети. ПП или пользователи взаимодействуют с клиентской частью, которая в простейшем случае обеспечивает надсетевой интерфейс. Клиентская часть системы при потребности обращается по сети к серверной части.
- Интерфейс серверной части определен и фиксирован. Поэтому возможно создание новых клиентских частей существующей системы (обеспечение интероперабельности на системном уровне).
- Основной проблемой систем К-С является то, что от них требуется мобильность в как можно более широком классе аппаратно-программных решений открытых систем.

# Технологии «клиент-сервер»

- Общим решением проблемы мобильности систем К-С является опора на программные пакеты, реализующие протоколы удаленного вызова процедур (RPC - Remote Procedure Call). При использовании таких средств обращение к сервису в удаленном узле выглядит как обычный вызов процедуры.
- При вызове удаленной процедуры программы RPC производят преобразование форматов данных клиента в промежуточные машинно-независимые форматы, и затем преобразование в форматы данных сервера. При передаче ответных параметров производятся аналогичные преобразования в обратном порядке.

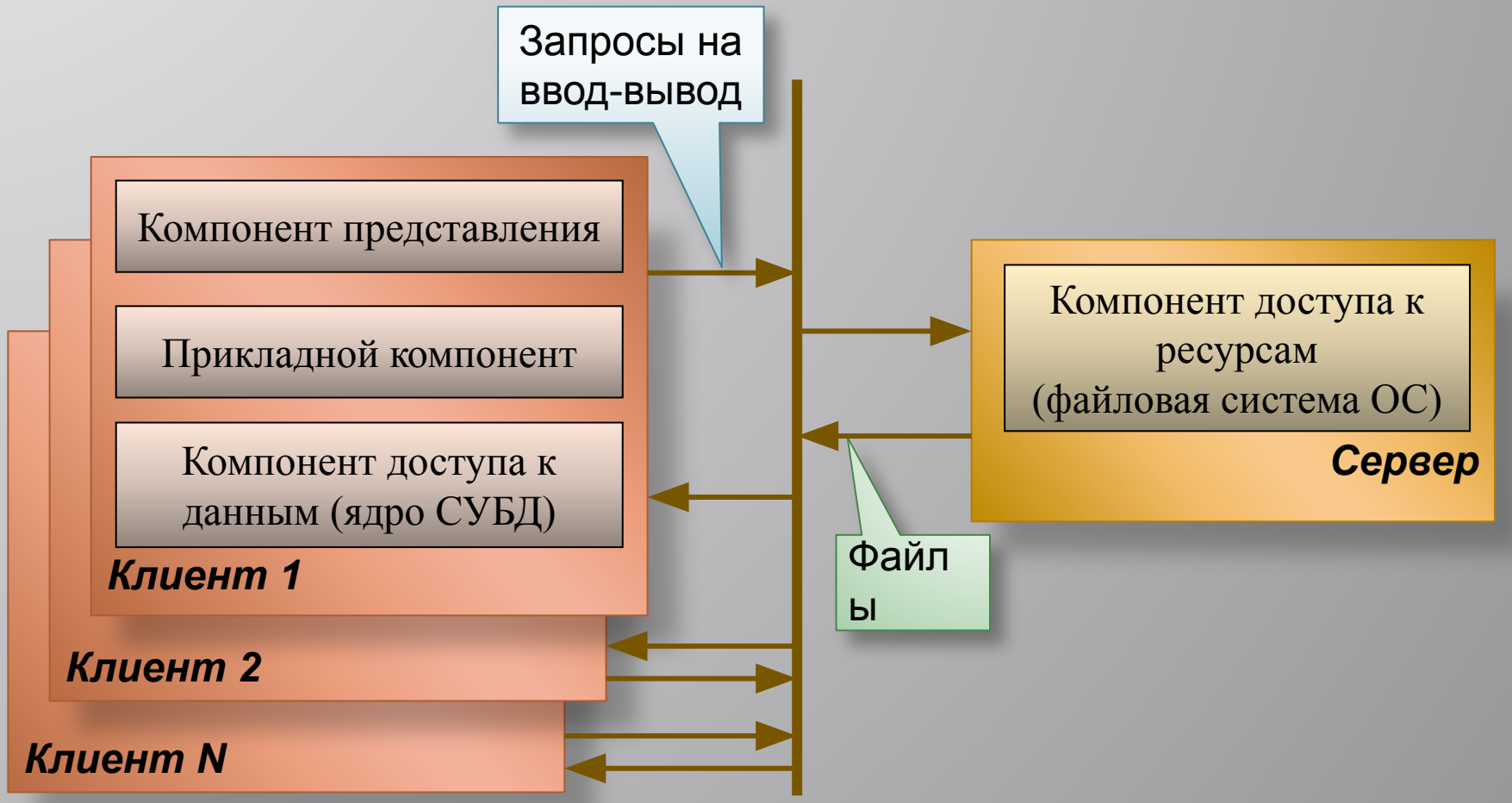
# Модели систем «клиент-сервер»

- Применительно к системам БД различия в реализациях систем К-С основана на разделении структуры СУБД на 3 компонента:
  - ***компонент представления***, реализующий функции ввода и отображения данных (интерфейс пользователя);
  - ***прикладной компонент***, включающий набор запросов, событий, правил, процедур и других функций и реализующий предназначение ИС в конкретной предметной области;
  - ***компонент доступа к данным***, реализующий функции хранения, извлечения, физического обновления и изменения данных (машина данных).

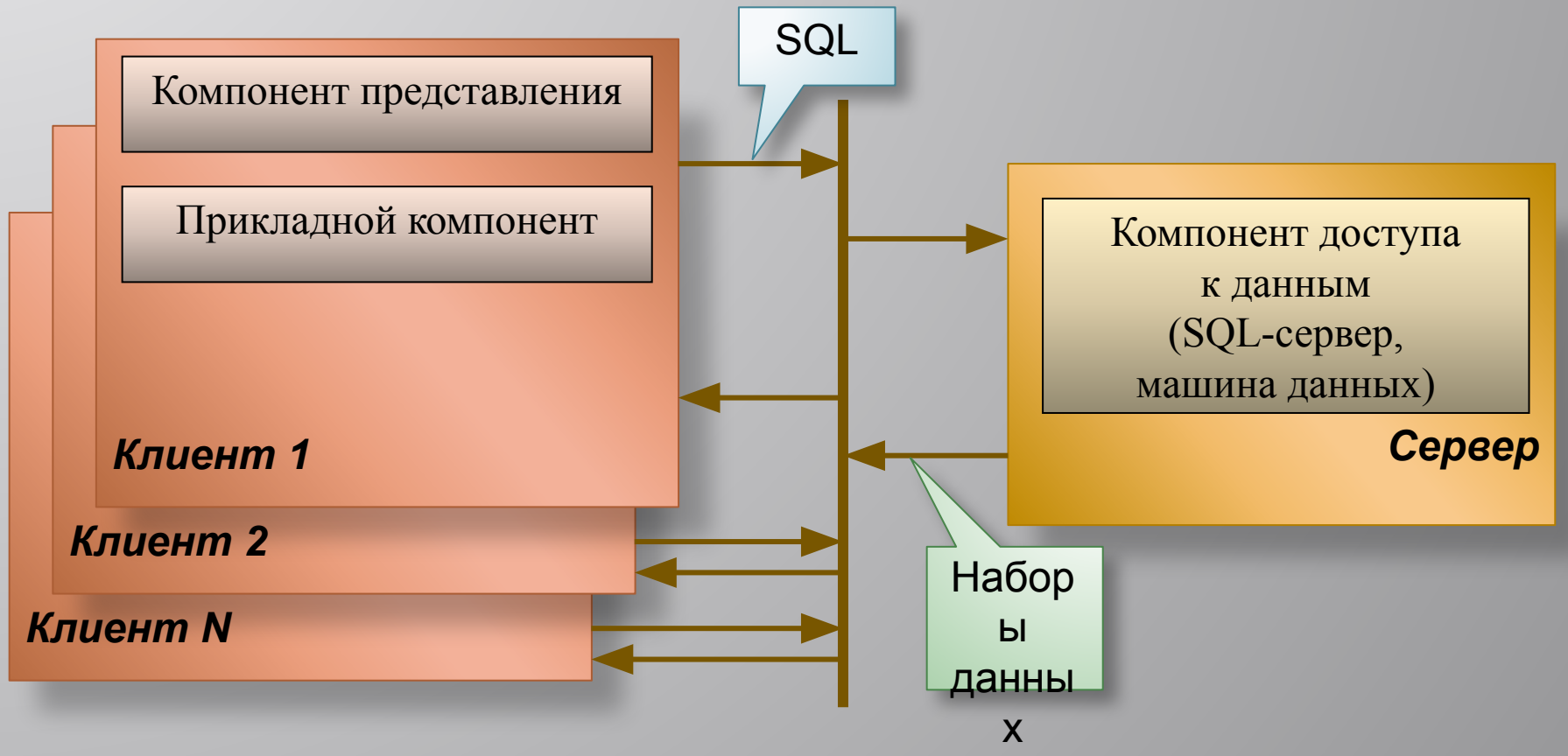
# Модели систем «клиент-сервер»

- В зависимости от особенностей реализации и распределения в системе указанных трех компонентов СУБД, различают четыре модели технологий К-С:
  - ***Модель файлового сервера*** (File Server – FS);
  - ***Модель удаленного доступа к данным*** (Remote Data Access RDA);
  - ***Модель сервера базы данных*** (Data Base Server - DBS);
  - ***Модель сервера приложений*** (Application Server – AS).

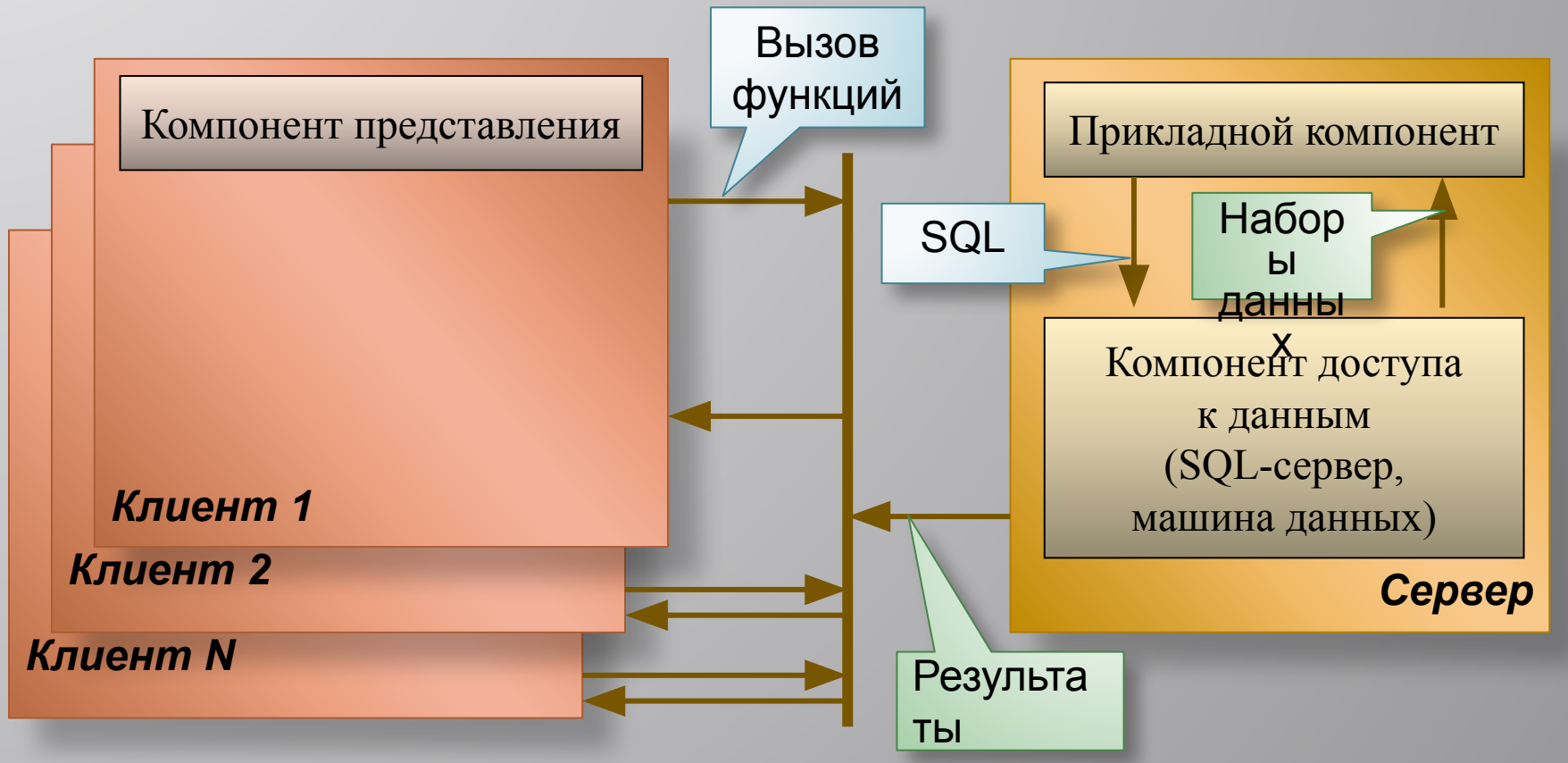
# Модель файлового сервера – FS-модель



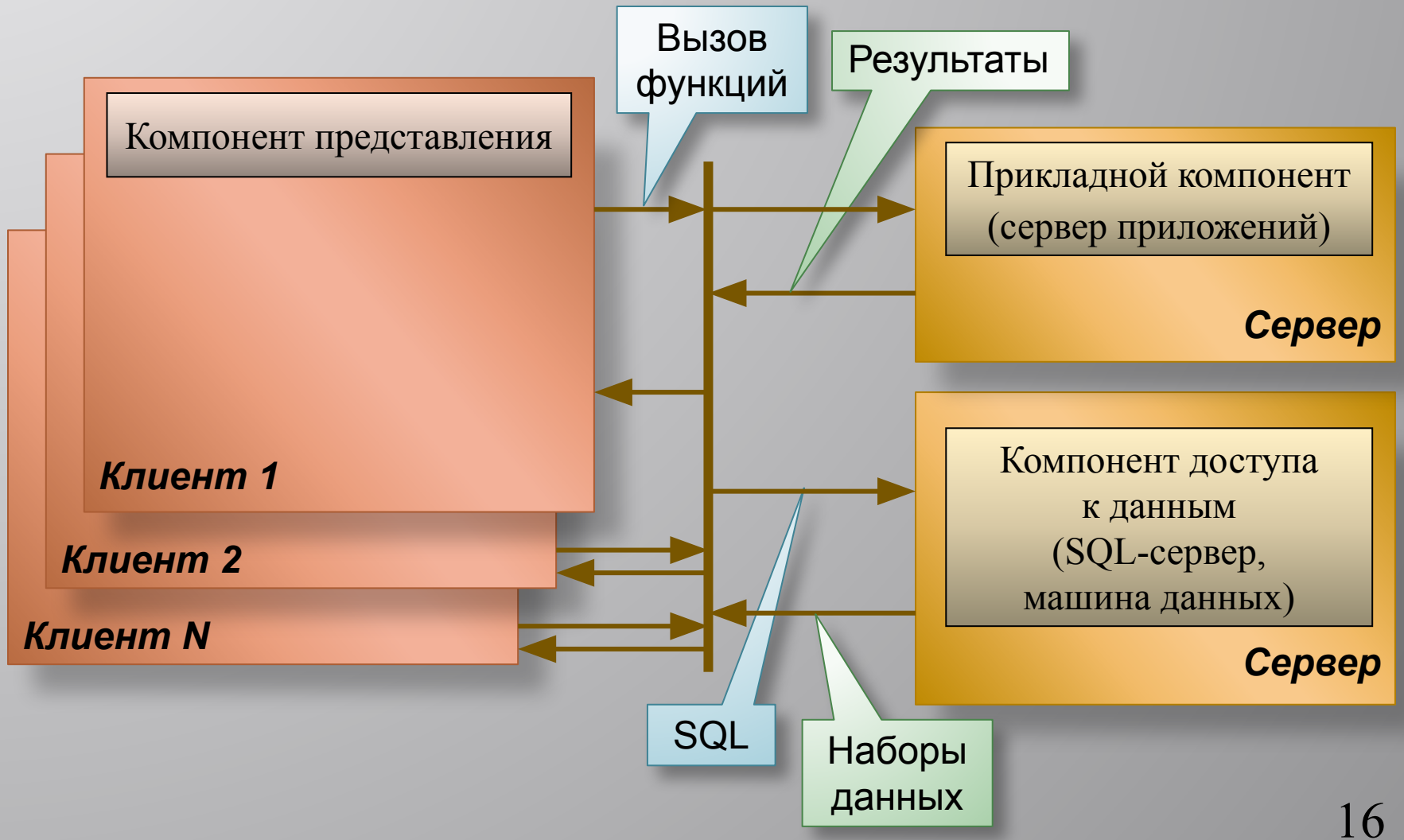
# Модель удаленного доступа к данным – RDA-модель



# Модель сервера базы данных – DBS-модель



# Модель сервера приложений – AS-модель



# Технологии объектного связывания данных

- ***Технология объектного связывания данных*** предназначена для интеграции разрозненных локальных БД под управлением «настольных» СУБД в сложные децентрализованные гетерогенные распределенные системы.
- С узкой точки зрения, технология объектного связывания данных решает задачу обеспечения доступа из одной локальной БД, открытой одним пользователем, к данным другой локальной БД, возможно находящейся на другой ВУ, открытой другим пользователем.
- Решение этой задачи основывается на поддержке современными «настольными» СУБД (MS Access, MS FoxPro, dBase и др.) ***технологии – DAO*** (Data Access Object), в переводе «объекты доступа к данным».

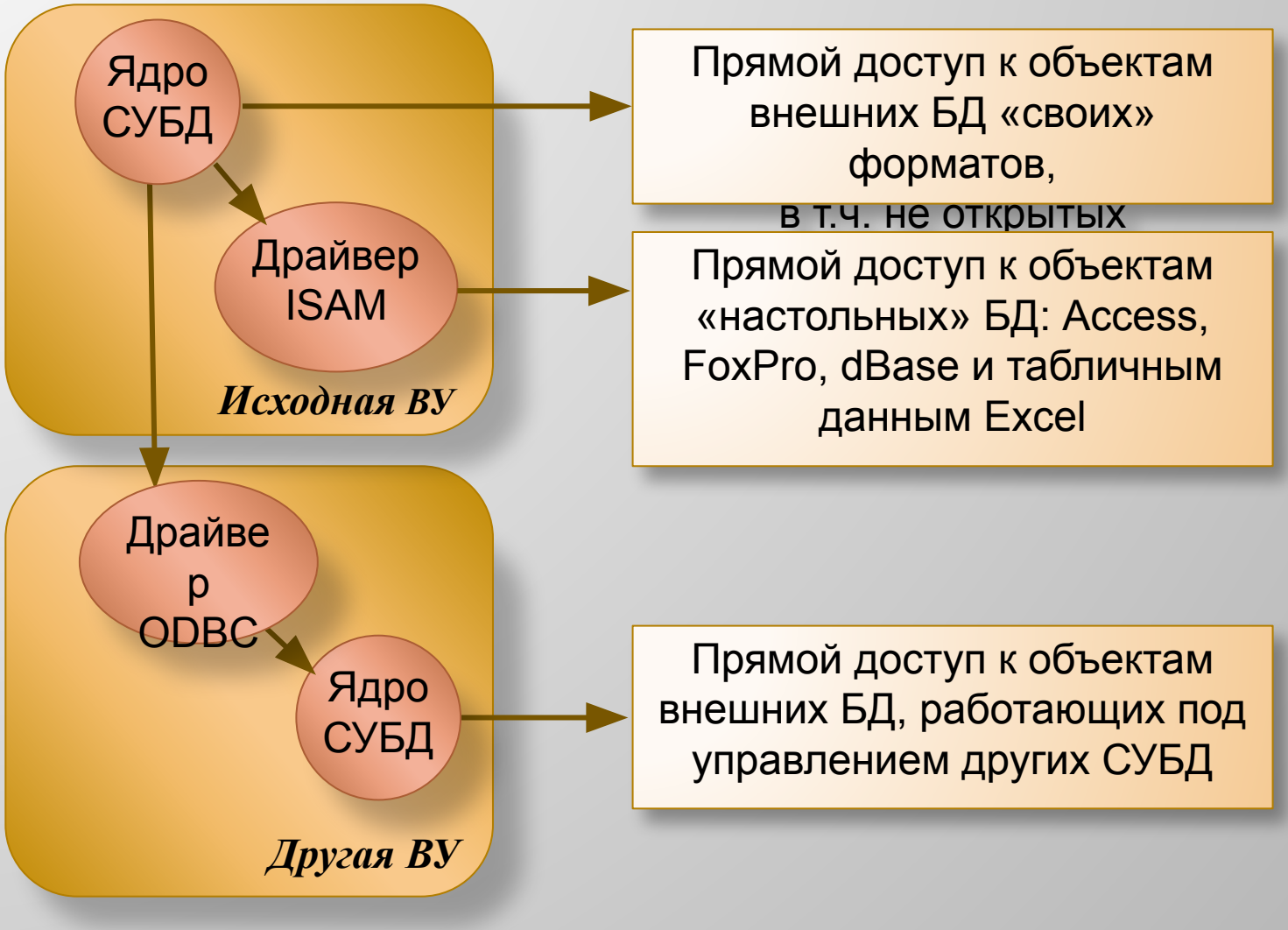
# Технологии объектного связывания данных

- Технически технология DAO основана на ***протоколе ODBC*** (Open Database Connectivity), в переводе – «открытый доступ к БД». ODBC – это общее определение языка и набора протоколов и является стандартом для обеспечения доступа к любым данным, находящимся под управлением реляционных СУБД.
- Непосредственно для доступа к данным на основе протокола ODBC используются инициализируемые на тех установках, где находятся данные, специальные программные компоненты - ***драйверы ODBC***, или инициализируемые ядра тех СУБД, под управлением которых были созданы и эксплуатируются внешние БД.

# Принципы доступа к данным на основе протокола ODBC

- Современные «настольные» СУБД обеспечивают различны виды доступа к внешним данным:
  - Прямой доступ к объектам (таблицам, запросам, формам) внешних БД «своих» форматов, в т.ч. не открытых;
  - Прямой доступ к объектам «настольных» БД, находящихся под управлением других СУБД наиболее распространенных форматов: Access, FoxPro, dBase, а также – к табличным данным электронных таблиц Excel;
  - Прямой доступ к объектам внешних БД, работающих под управлением других СУБД.

# Принципы доступа к данным на основе протокола ODBC



# Технологии реплицирования данных

- Альтернативу построения быстродействующих РБД предоставляют ***технологии реплицирования данных*** (ТРД).
- ***Репликой*** называют особую копию БД для размещения на другом компьютере сети с целью автономной работы пользователей с одинаковыми (согласованными) данными.
- Идея ***реплицирования*** заключается в том, что пользователи работают автономно с одинаковыми (общими) данными, растиражированными по локальным БД.
- В ТРД ПО СУБД дополняется функциями тиражирования (реплицирования) БД, включая тиражирование как самих данных и их структуры, так и системного каталога с информацией о размещении реплик (о конфигурации РБД).

# Технологии реплицирования данных

- При использовании ТРД возникают две проблемы обеспечения непрерывности согласованного состояния данных:
  - обеспечение согласованного состояния во всех репликах количества и значений общих данных;
  - обеспечение согласованного состояния во всех репликах структуры данных.
- Обеспечение согласованного состояния общих данных, в свою очередь, основывается на реализации одного из двух принципов:
  - принципа непрерывного размножения обновлений;
  - принципа отложенных обновлений.

# Принцип непрерывного размножения обновлений реплик

- Принцип ***непрерывного размножения обновлений реплик*** применяется при построении т. н. «систем реального времени» (системы управления воздушным движением, системы бронирования билетов пассажирского транспорта и т. п.), где требуется непрерывное и точное соответствие реплик или других растиражированных данных во всех узлах и компонентах РБД.
- Реализация принципа непрерывного размножения обновлений заключается в том, что любая транзакция считается успешно завершенной, если она успешно завершена на всех репликах системы.
- Реализация этого принципа встречает затруднения, связанные с тупиками

# Принцип отложенных обновлений реплик

- ИС с невысокой динамикой обновления данных можно строить на основе ***принципа отложенных обновлений***.
- Накопленные в какой-либо реплике изменения данных специальной командой пользователя направляются для обновления всех остальных реплик системы. Такая операция называется ***синхронизацией реплик***.
- Возможность конфликтов и тупиков при синхронизации реплик в этом случае существенно снижается, а немногочисленные конфликтные ситуации легко разрешаются организационными мерами.
- Решение проблемы согласованности структуры данных основывается на технике ***главной*** реплики.

# Принцип отложенных обновлений реплик

- Суть этой техники заключается в том, что одна из реплик РБД объявляется главной. При этом изменять структуру БД можно только в главной реплике. Эти изменения тиражируются на основе принципа отложенных обновлений, т. е. через специальную синхронизацию реплик.
- Частичность отступления от принципа отсутствия центральной установки заключается в том, что в отличие от чисто централизованных систем, выход из строя главной реплики не влечет сразу гибель всей распределенной системы, так как остальные реплики продолжают функционировать автономно.

# Синхронизация реплик

- Процесс синхронизации реплик в современных СУБД включает обмен не всеми, а только теми данными, которые были изменены или добавлены в разных репликах. С этой целью в системном каталоге БД создаются специальные таблицы текущих изменений и организуется система ***глобальной идентификации*** всех объектов распределенной системы, включая раздельное именование одинаковых объектов (вплоть до записей таблиц) в разных репликах (это т.н. техника глобальных уникальных идентификаторов – GUID). Такой подход несколько увеличивает объем БД, но позволяет существенно ограничить транспортные расходы на синхронизацию реплик.

# Частичные реплики

- В технологиях реплицирования имеется возможность создания т.н. ***частичных реплик*** и включения в реплики как реплицируемых, так и не реплицируемых объектов.
- Частичной репликой называется БД, содержащая ограниченное подмножество записей полной реплики.
- Распространенным способом создания частичных реплик является установка фильтров на таблицы полной (главной) реплики.
- Частичные реплики позволяют решить некоторые проблемы по разграничению доступа к данным, повышают производительность РБД, снижают затраты на синхронизацию реплик.

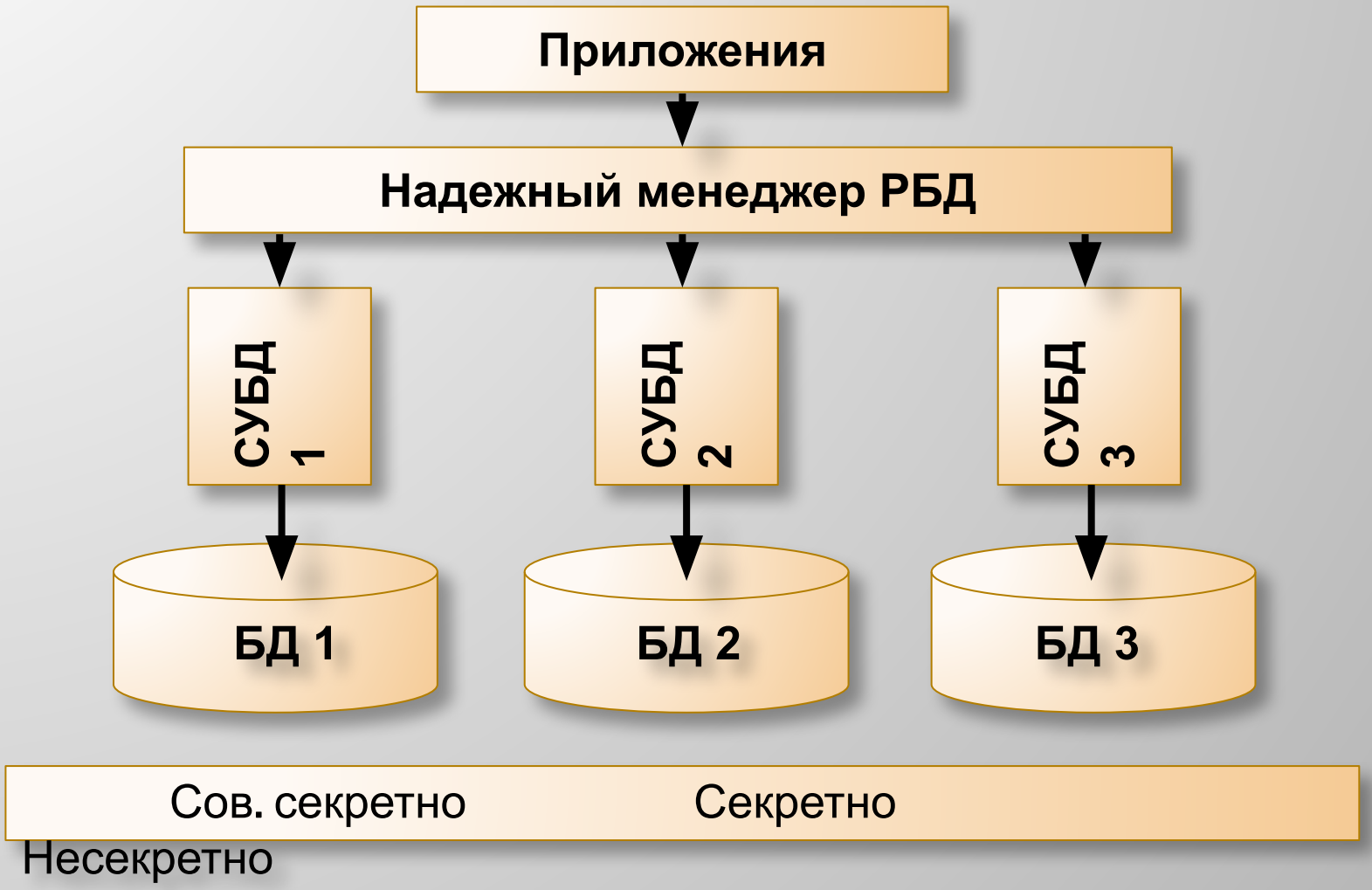
# Смешанные технологии РБД

- На практике для коллективной обработки данных применяются смешанные технологии, включающие элементы объектного связывания данных, репликации и клиент-серверных решений.
- В таких технологиях дополнительно к проблеме логического проектирования, добавляется проблема транспортно-технологического проектирования информационных потоков, разграничения доступа и т. д.
- Пока не проработаны теоретико-методологические и инструментальные подходы для автоматизации проектирования РБД с учетом логики, инфраструктуры предметной области и информационной безопасности.

# РБД с мандатной защитой

- Истоки исследований безопасных РБД относятся к 1982 г., к проекту Woods Hole американских ВВС, в котором были определены две альтернативные архитектуры распределенных СУБД (СУРБД). Общей чертой обеих архитектур было наличие надежного менеджера, подключаемого к раздельным БД.
- **В первой архитектуре** использованы три БД по степени секретности. Менеджер в соответствии  с уровнем секретности приложений направляет выполнение операций (в том числе запросов) над соответствующие БД.
- Запросы на доступ к СС-информации с необходимым допуском направляются на СУБД высокого уровня, а многоуровневые запросы подвергаются декомпозиции и соответствующим образом распределяются.

# РБД с мандатной защитой



# РБД с мандатной защитой

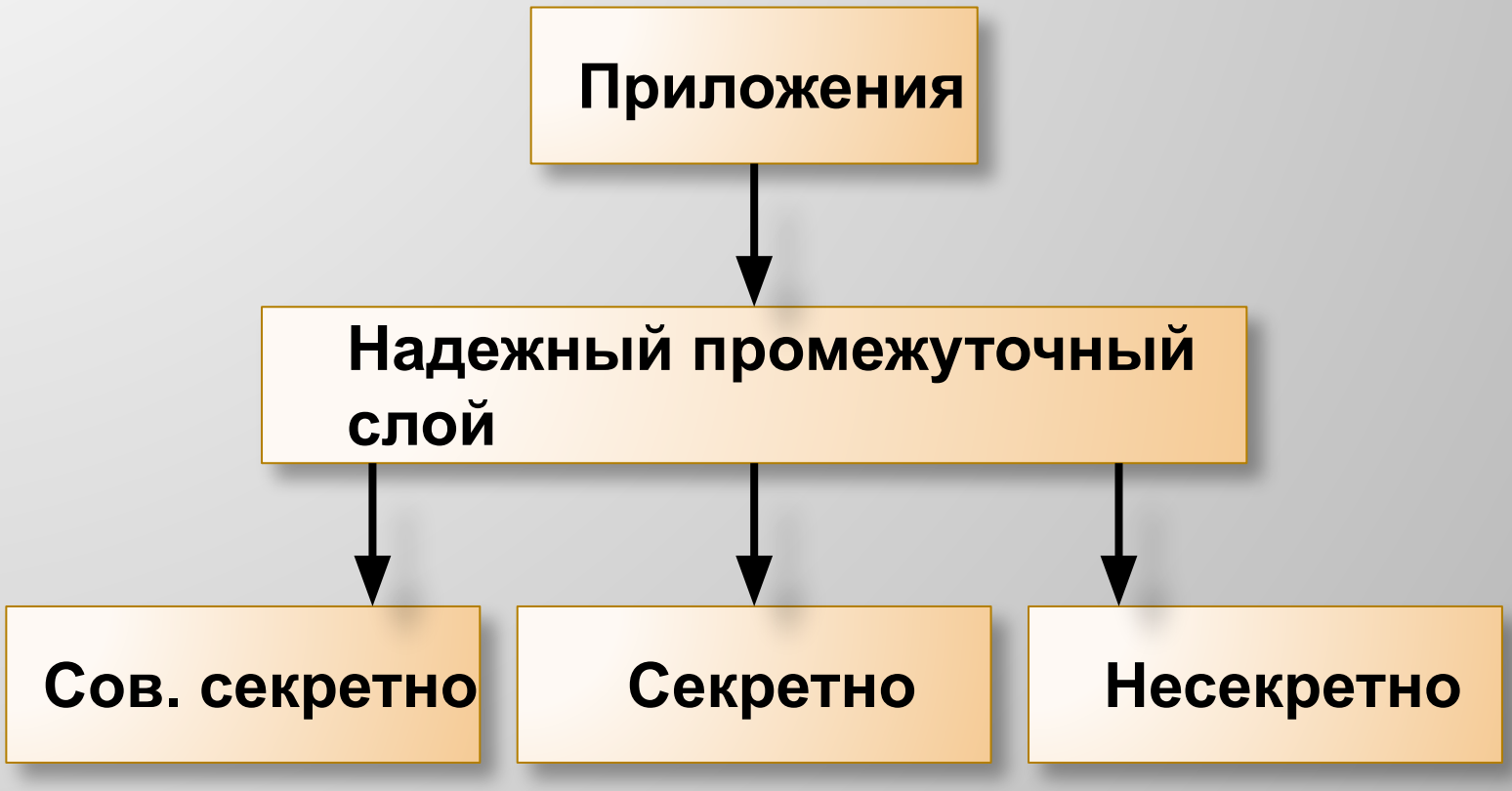
- **Вторая архитектура** несколько сложнее и рассчитана на наличие нескольких уровней в пределах одной СУБД.
- Операции низкого уровня обрабатываются так же, как в первой архитектуре.
- Операции высокого уровня и многоуровневые операции обрабатываются СУБД, на которой хранятся СС-данные и копии данных низших степеней секретности.
- Такая архитектура подразумевает использование средств тиражирования и, следовательно, в ней должны присутствовать менеджеры тиражирования и средства соответствующая политика обеспечения целостности данных.

# РБД с мандатной защитой

# Проект SD-DBMS ВВС США

- SD-DBMS следует базовой модели Белл-ЛаПадула. Система обеспечивает добровольное управление доступом посредством представлений доступа (access views) в виде виртуальных отношений, производных от базовых и подобных традиционным SQL-представлениям.
- Субъектам (пользователям и процессам) не разрешается обращаться непосредственно к базовым отношениям, они имеют доступ к данным только через посредство представлений с контролируемым доступом.
- Каждое вновь создаваемое многоуровневое отношение разбивается на одноуровневые фрагменты, хранимые на разных узлах заднего плана в соответствии со своим классом безопасности.

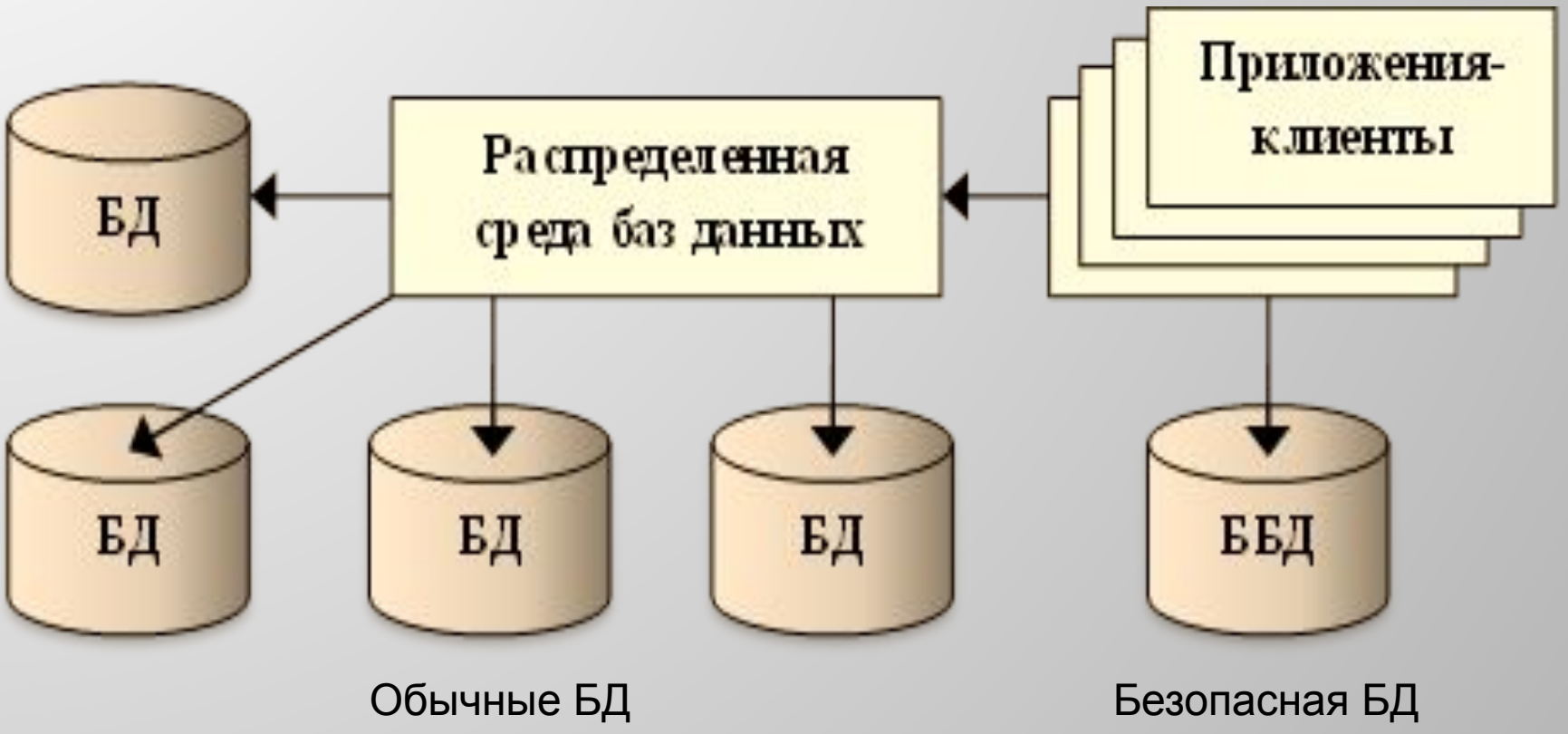
# Проект SD-DBMS ВВС США



# Системы мультибаз данных

- Системы МБД ориентированы на то, что в корпорации будут существовать островки безопасной обработки, включающие безопасные БД и информационные менеджеры, отделенные от мероприятий всеобщей интеграции.
- Пользователь соответствующего класса благонадежности может иметь доступ к среде безопасной обработки, возможно, включающей защищенную СУРБД, находящуюся за рамками программы интеграции корпоративной информационной среды, независимо от реализации, уровня прозрачности и других аспектов организации МБД.

# Системы мультибаз данных



# Политики и модели безопасности в распределенных КС и БД

- В части безопасности РКС можно выделить 3 аспекта:
  - для ПО современных РКС характерен модульный принцип реализации ядра системы и, в т. ч. монитора безопасности (МБ), и отдельные функции безопасности распределены между модулями системы;
  - в процессах управления доступом МБ использует объект, содержащий информацию по политике разграничения доступа в конкретной КС, который может реализовываться как отдельный объект, или как распределенная структура;
  - современные РКС представляют собой системы, объединяющие физически распределенные ВУ.

# Политики и модели безопасности в распределенных КС и БД

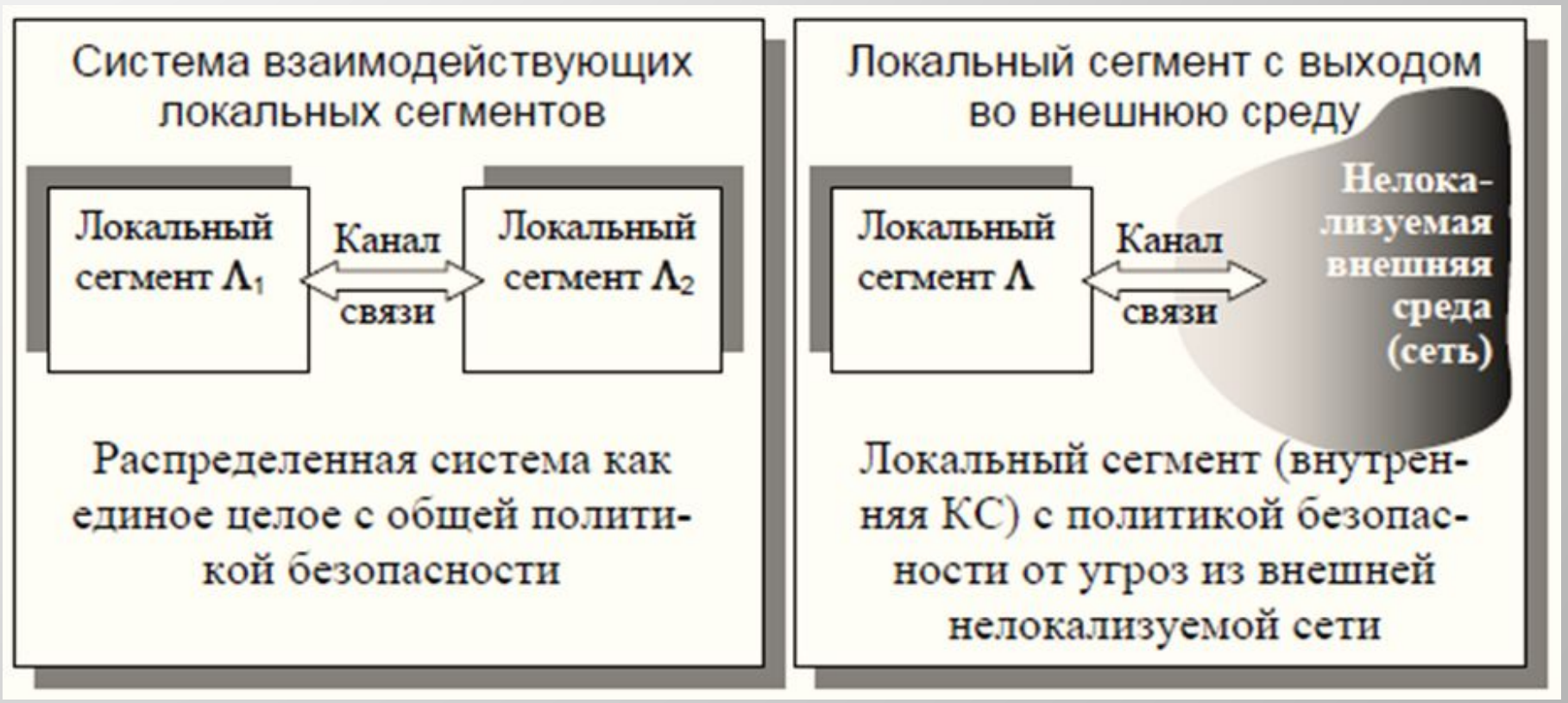
- ***Определение.*** РКС называется система, состоящая более чем из одного локального сегмента, представляющего обособленную совокупность субъектов и объектов доступа.
- Выделяют три способа обособления подмножества субъектов и объектов в локальный сегмент:
  - группирование некоторого подмножества субъектов доступа;
  - локализация некоторого подмножества субъектов и объектов доступа;
  - присвоение всем субъектам и объектам некоторого уникального идентификатора (адреса).

# Политики и модели безопасности в распределенных КС и БД

- Принципиальным во всех трех подходах является то, что локализуемый сегмент с точки зрения безопасности и разграничения доступа рассматривается как монолитная (нераспределенная) КС.
- ***Предустановка1.*** В локальном сегменте РКС функционирует единый для всех субъектов и объектов монитор безопасности, реализующий локальную политику безопасности (политику разграничения доступа).
- Предметом политики разграничения доступа в РКС является рассмотрение принципов организации и механизмов доступа, обеспечивающих как внутрисегментную, так и межсегментную безопасность.

# Политики и модели безопасности в распределенных КС и БД

- Две основные разновидности архитектуры РКС:



# Политики и модели безопасности в распределенных КС и БД

- ***Предустановка 2.*** Множество пользователей $U$ распределенной системы взаимодействующих локальных сегментов $\{\Lambda 1, \Lambda 2, \ldots, \Lambda_K\}$ является объединением непересекающихся подмножеств $\{U1, U2, \ldots, U_K\}$ пользователей соответствующих локальных сегментов.
- Пользователей, осуществляющих доступ из одного локального сегмента к объектам другого локального сегмента, часто называют удаленными пользователями.
- Для обеспечения удаленных доступов выделяется самый верхний слой политики безопасности – политика отношений между локальными сегментами, которая выражается парадигмой доверия (доверительными отношениями).

# Политики и модели безопасности в распределенных КС и БД

- ***Определение.*** Доверительными отношениями между локальными сегментами $\Lambda_1$ и $\Lambda_2$ называется составная часть общесистемной политики безопасности РКС, определяющая возможность осуществления удаленных доступов пользователей одного локального сегмента РКС к объектам другого локального сегмента.
- Выделяют следующие виды доверительных отношений:
  - одноуровневые отношения доверия, подразделяемые, в свою очередь, на:
    - отношения одностороннего доверия;
    - отношения двустороннего доверия;
  - иерархические отношения доверия.

# Политики и модели безопасности в распределенных КС и БД

- Можно сформулировать два направления обеспечения безопасности в РКС:
  - выделение специального системного субъекта, обеспечивающего "внешнюю" безопасность;
  - реализация общего МБ, обеспечивающего единую (согласованную) политику безопасности РКС.
- Первое направление, получило распространение в виде широко известных в настоящее время межсетевых экранов.
- Задачи второго направления были поставлены еще в сетевой интерпретации "Оранжевой книги в виде требований по реализации сетевого монитора безопасности– NTCB (Network Trusted Computer Base).

# Модель безопасности РКС Варадхараджана

- Предложена в 1990 г. Модель основана на сочетании дискреционного, мандатного и ролевого принципа разграничения доступа.
- Множество объектов системы включает помимо информационных объектов и множество сетевых компонент, имеющих, в том числе, свою классификацию.
- В состав множества операций включены операции по установлению связи субъектов с удаленными компонентами сети.
- Выполнение операций сопровождается прохождением системой двух фаз – фазы доступа к системе и фазы установления связи.

# Зональная модель разграничения доступа в РКС

- Исходным мотивом для данных решений послужил анализ наработанных во внекомпьютерной сфере подходов к разграничению доступа, в частности, организационно-режимных схем управления доступом на территорию и в помещения сотрудников предприятий и учреждений.
- Применяется рубежно-иерархический принцип построения и дифференциация режимных мер путем выделения так называемых ***зон*** – открытых, режимных (закрытых), производственно-технологических, складских и т. п.
- Каждая зона характеризуется своим режимом (политикой).

# Зональная модель разграничения доступа в РКС

1. РКС представляется совокупностью наборов сущностей:
   - множества объектов доступа $O(o_1, o_2,\ldots, o_M)$;
   - множества пользователей $U(u_1, u_2,\ldots, u_N)$;
   - множества физических объектов системы $V(v_1,v_2,\ldots, v_L)$ – вычислительных установок, принтеров, коммуникационного оборудования и т. п.;
   - множества зон системы $Z(z_1, z_2,\ldots, z_K)$.

- ***Определение.*** Зоной в РКС называется совокупность подмножества пользователей, подмножества объектов доступа и подмножества физических объектов, обособленных в локальный сегмент с отдельной (внутризональной) политикой безопасности.

# Зональная модель разграничения доступа в РКС

- Внутризональная политика безопасности реализуется внутризональным монитором безопасности, который обеспечивает весь набор функций безопасности (аутентификация и порождение первичных субъектов доступа, управление доступом, аудит процессов).
- Ввиду того, что зона может быть образована на нескольких ВУ, возможны два подхода реализации единого внутризонального монитора безопасности:
  - по принципу "Клиент-сервер";
  - по принципу реплицирования.

# Зональная модель разграничения доступа в РКС

- ***Предустановка.*** Зонально-распределенная система строится на безопасном канале связи, который удовлетворяет следующим требованиям:
  - устойчивость к НСД или модификации передаваемой ценной информации (безопасность связи);
  - невозможность отказов от доставки сообщения, неправильной доставки, доставки ошибочных данных (надежность связи);
  - невозможность изменений в критической информации (имитозащита);
  - отсутствие скрытых каналов утечки информации за счет модуляции параметров канала.

# Зональная модель разграничения доступа в РКС

- Любые субъекты любых пользователей в своих зонах находятся под полным контролем соответствующих внутризональных мониторов безопасности, санкционирующих только подмножества безопасных (разрешенных) доступов.
- Иначе говоря, внутризональные мониторы безопасности, организующие процедуры вхождения пользователей в свои зоны, гарантируют отсутствие в своих зонах субъектов, осуществляющих доступы вне подмножеств разрешенных внутризональных доступов.

# Зональная модель разграничения доступа в РКС

- Субъекты доступа к объектам зон от удаленных пользователей также находятся под полным контролем соответствующих внутризональных МБ, так как процедурами вхождения в зоны пользователей других зон (удаленных пользователей) управляют МБ зоны вхождения.
- При этом процедуры вхождения удаленных пользователей и инициализация в зонах их субъектов производится при выполнении условия:
  - зона вхождения доверяет зоне удаленного пользователя;
  - пользователь уполномочен работать в данной зоне, входящей в подмножество доверенных зон.

# Зональная модель разграничения доступа в РКС

1. Общезональная безопасность гарантируется на основе априорно заданной внутризональной безопасности и определенных процедур, отношений, регламентаций по взаимодействию зон. Отсюда следует, что изъяны и бреши в системе безопасности всей РКС могут возникнуть либо на основе изъянов внутризональной безопасности, либо на основе ошибок при реализации правил и регламентаций межзонального взаимодействия.
2. Процедуры вхождения удаленных пользователей в "не свои" зоны не требуют дополнительной аутентификации. МБ доменов доверяют друг другу по процедурам аутентификации пользователей, что создает ряд сетевых угроз безопасности.

# Управление доступом в Java-среде

- В настоящее время широко используется подход к безопасной распределенной обработки данных в сети, основанный на использовании языка Java.
- Java – это ОО-язык программирования, разработанный компанией Sun Microsystems (поглощена Oracle). Приложения Java обычно транслируются в специальный байт-код – апплет (applet).
- Java-апплет может исполняться на любых локальных компьютерах и других устройствах, поддерживающих Java.
- Апплеты используются для предоставления интерактивных возможностей веб-приложений, которые не могут быть предоставлены HTML.

# Управление доступом в Java-среде

- Поскольку байт-код Java платформенно-независим, то Java-апплеты могут выполняться с помощью плагинов браузерами многих платформ, включая Microsoft Windows, UNIX, Apple Mac OS и GNU/Linux.
- Т.е. Java-апплеты могут работать на любой виртуальной Java-машине (JVM) вне зависимости от компьютерной архитектуры.
- JVM – это, по существу, программа, обрабатывающая байтовый код и передающая инструкции оборудованию как интерпретатор (инструкции направлены на непосредственное выполнение программы).

# Управление доступом в Java-среде

- **Важной особенностью технологии Java является возможность обеспечения РКС гибкой системой безопасности, благодаря тому, что исполнение программы полностью контролируется виртуальной машиной. Любые операции, которые превышают установленные полномочия программы (например, попытка НСД к данным или, – соединения с другим компьютером) вызывают немедленное прерывание.**
- К недостаткам концепции JVM относят то, что исполнение байт-кода виртуальной машиной может приводить к увеличению потребления памяти и процессорного времени КС.

# Модели безопасности Java

- МБ Java претерпели несколько этапов эволюции.
- Для разработки приложений на языке Java компанией Oracle бесплатно распространяется комплект разработчика приложений Java Development Kit (JDK). JDK включает в себя компилятор, стандартные библиотеки классов, примеры, документацию, различные утилиты и исполнительную систему Java Runtime Environment (сокр. JRE), представляющую собой минимальную реализацию JVM.
- Все современные интегрированные среды разработки приложений на Java (JDeveloper, NetBeans IDE, Sun Java Studio Creator, IntelliJ IDEA, Borland JBuilder, Eclipse) опираются на сервисы, предоставляемые JDK.

# Модели безопасности Java

- **В версии JDK 1.0** была предложена концепция "песочницы" (sandbox) – замкнутой среды, в которой выполняются потенциально ненадежные апплеты, поступившие по сети. При этом программы, располагающиеся на локальном компьютере, считались абсолютно надежными, и им было доступно все, что доступно виртуальной Java-машине.
- В число ограничений, налагаемых "песочницей", входит запрет на доступ к локальной файловой системе, на сетевое взаимодействие со всеми хостами, кроме источника апплета, и т.п.
- Такие ограничения сводили к минимуму возможности для содержательных действий у апплетов.

# Модели безопасности Java

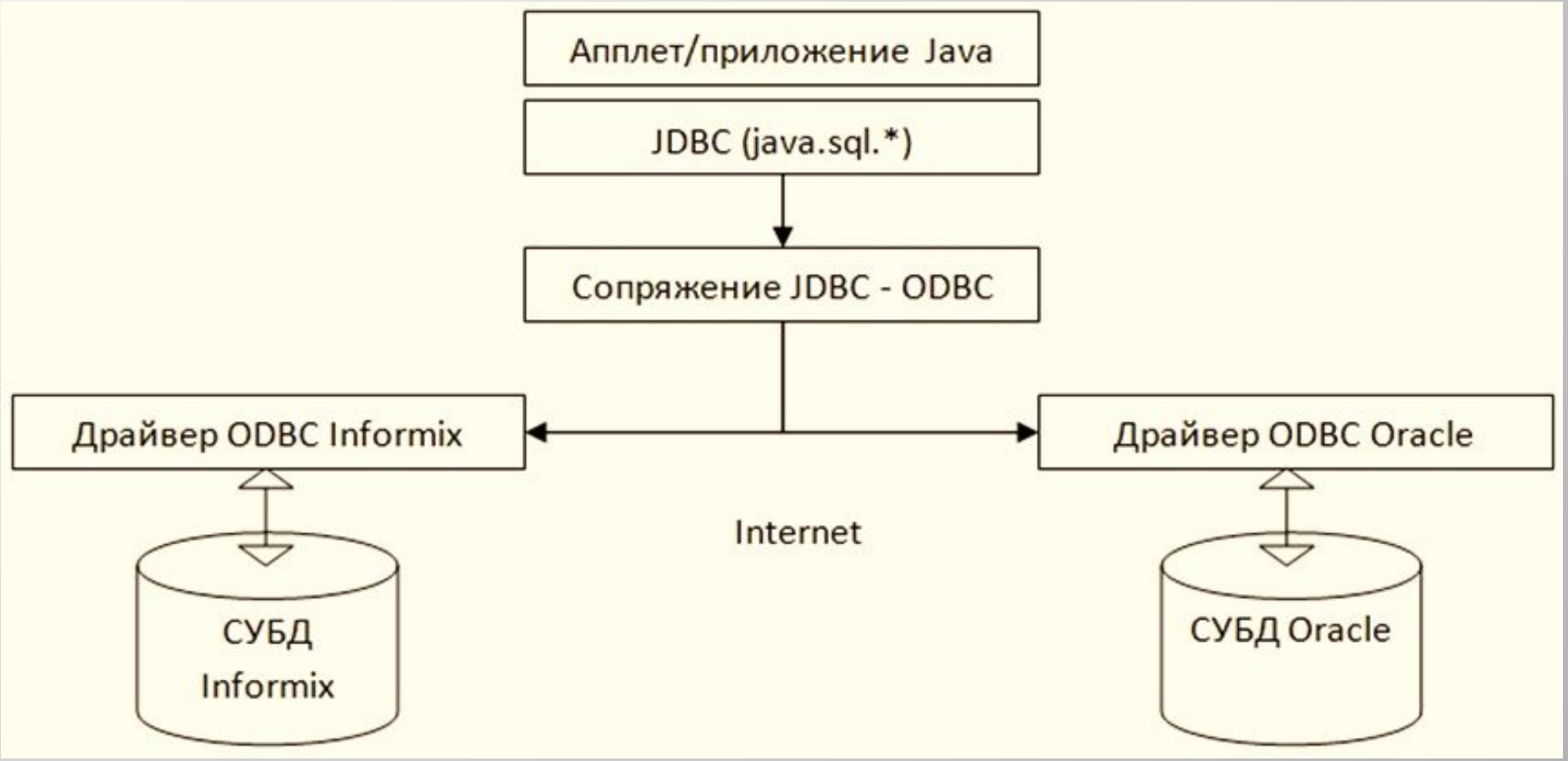
- В следующей **версии JDK 1.**1 ввели деление источников апплетов на надежные и ненадежные. Надежность источника определялся по электронной подписи. Надежные апплеты приравнивались в правах к "родному" коду. Это решило проблемы тех, кому прав не хватало, но защита осталась неполной.
- В **версии JDK 1.2** сформировалась модель безопасности, используемая в Java 2. В этой модели от концепции "песочницы" отказались. Ввели 3 основных понятия:
  - источник программы;
  - право и множество прав;
  - политика безопасности.

# Модели безопасности Java

- В результате **в модели JDK 1.2** получился традиционный для современных ОС и СУБД механизм прав доступа со следующими особенностями:
  - Java-программы выступают не от имени пользователя, их запустившего, а от имени источника программы;
  - нет понятия владельца ресурсов, который мог бы менять права; последние задаются исключительно политикой безопасности (формально можно считать, что владельцем всего является тот, кто формирует политику безопасности);
  - механизмы безопасности снабжены объектной оберткой.

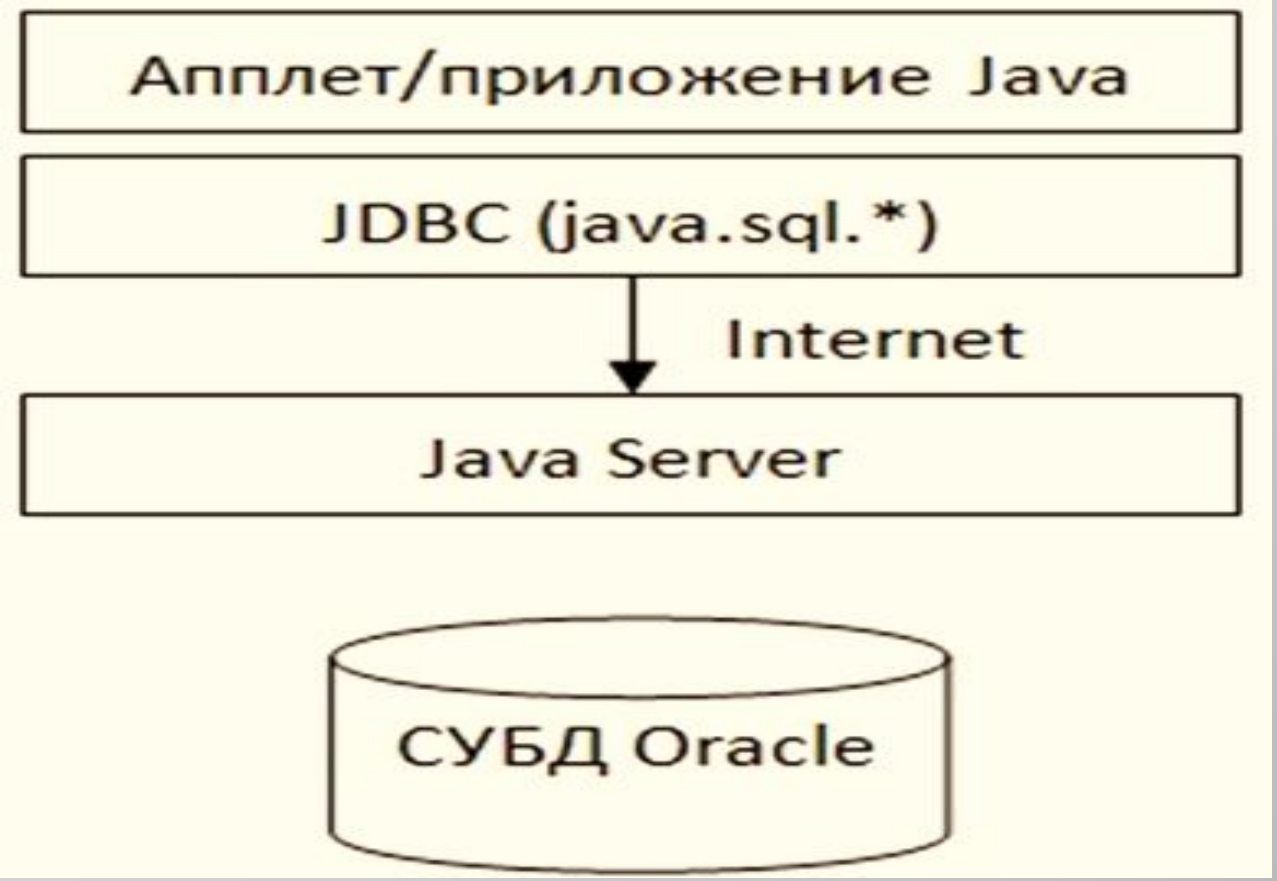
# Взаимодействия апплета с БД

Первый вариант схемы соединения апплета с БД



# Взаимодействия апплета с БД

- Второй вариант схемы соединения апплета с БД



# Достоинства апплетов

- Кроссплатформенность.
- Апплет может работает на «всех» версиях Java.
- Апплет поддерживается большинством браузеров.
- Апплет кэшируется в большинстве браузеров.
- Апплет может иметь полный доступ к машине, на которой выполняется, если пользователь согласен на это.
- Апплет может улучшить использование браузера.
- Апплет может запуститься с сопоставимой скоростью на других компилируемых языках.
- Апплет может перенести работу с сервера к клиенту, делая Интернет-решение с большим числом клиентов.

# Недостатки апплетов

- Апплет требует установки Java-расширения (plug-in).
- Апплет не может запуститься до тех пор, пока не запустится виртуальная Java-машина.
- Апплет небезопасен по своей природе.
- Создание пользовательского интерфейса с использованием апплетов более сложно чем в HTML.
- Проблемы, связанные с идентификацией / аутентификацией пользователей и проверкой доступа к БД.
- Для безопасности апплета, ограничивается доступ к пользовательской системе.
- Если ПО установлено администратором, то пользователи не могут видеть апплеты по умолчанию.
- Апплеты могут потребовать определенную версию JRE.

# Перспективы развития безопасности распределенных систем

- В настоящее время проблема управления доступом существует в трех почти не связанных между собой проявлениях:
  - традиционные модели (дискреционная и мандатная);
  - модель "песочница" (предложенная для Java-среды и близкой ей системы Safe-Tcl);
  - модель фильтрации (используемая в межсетевых экранах).
- Высказываются предложения объединить эти существующие подходы на основе их развития и обобщения (В. Галатенко).