Виды серверного программного обеспечения АИС. Управляющие серверы

1. Локальные вычислительные сети

Локальная вычислительная сеть (ЛВС) представляет совокупность компьютеров, расположенных на ограниченной территории и объединенных каналами связи для обмена информацией и распределенной обработки данных.

Организация ЛВС позволяет решать следующие задачи:

- **обмен информацией между абонентами сети, что позволяет сократить бумажный документооборот и перейти к электронному документообороту;**

- **обеспечение распределенной обработки данных, связанное с объединением АРМ всех специалистов данной организации в сеть. Несмотря на существенные различия в характере и объеме расчетов, проводимых на АРМ специалистами различного профиля, используемая при этом информация в рамках одной организации находится в единой базе данных, поэтому объединение таких АРМ в сеть является целесообразным и эффективным решением;**
- **поддержка принятия управленческих решений, предоставляющая руководителям и управленческому персоналу организации достоверную и оперативную информацию, необходимую для оценки ситуации и принятия правильных решений;**

- **организация собственных информационных систем, содержащих автоматизированные банки данных;**
- **коллективное использование ресурсов, таких, как высокоскоростные печатающие устройства, запоминающие устройства большой емкости, мощные средства обработки информации, прикладные программные системы, базы данных, базы знаний.**

При этом эффективность функционирования локальной вычислительной сети характеризуется:

Производительностью	**Производительность ЛВС оценивается:** **• временем реакции на запросы клиентов ЛВС;** **• пропускной способностью, равной количеству данных, передаваемых за единицу времени;** **• задержкой передачи пакета данных устройствами сети**
Надежностью	**Для оценки надежности ЛВС вводятся такие характеристики, как коэффициент готовности и устойчивости к отказам, т. е. способность работать при отказе части устройств. Сюда же относят и безопасность, т. е. способность ЛВС защищать данные от несанкционированного доступа к ним**
Расширяемостью	**Расширяемость характеризует возможность добавления новых элементов и узлов в ЛВС**
Управляемостью	**Управляемость - это возможность контролировать состояние узлов ЛВС, выявлять и разрешать проблемы, возникающие при работе сети, анализировать и планировать работу ЛВС**
Совместимостью	**Совместимость - это возможность компоновки ЛВС на основе разнородных**

ЛВС включает следующие основные компоненты, представленные на рис. 1.

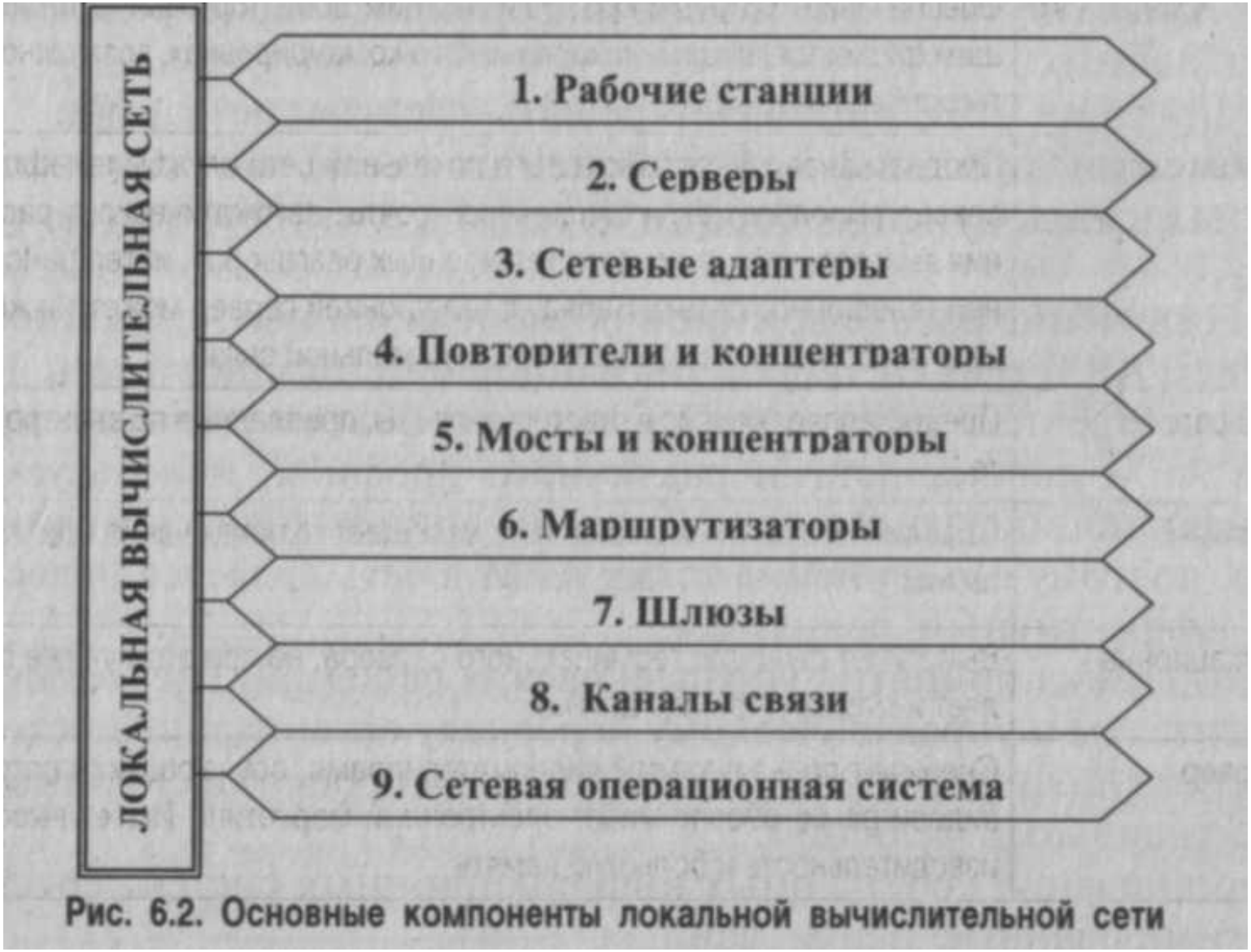


Рис. 6.2. Основные компоненты локальной вычислительной сети

1. Рабочая станция — это персональный компьютер, подключенный к сети, через который пользователь получает доступ к сетевым ресурсам. Рабочая станция функционирует как в сетевом, так и в локальном режиме и обеспечивает пользователя всем необходимым инструментарием для решения прикладных задач.

2. Сервер — это компьютер, выполняющий функции управления сетевыми ресурсами общего доступа: осуществляет хранение данных, управляет базами данных, выполняет удаленную обработку заданий, обеспечивает печать заданий и др.

Виды серверов и их назначение

Таблица 1

Вид сервера	Назначение
Универсальный сервер	Предназначен для выполнения несложного набора различных задач обработки данных в локальной сети
Сервер базы данных	Выполняет обработку запросов, направляемых базе данных
Прокси-сервер (Proxy-сервер)	Обеспечивает подключение рабочих станций локальной сети к глобальной сети Internet
Web-сервер	Предназначен для работы с web-ресурсами глобальной сети Internet
Файловый сервер	Обеспечивает функционирование распределенных ресурсов, включая файлы и программное обеспечение
Сервер приложений	Предназначен для выполнения прикладных процессов. С одной стороны, взаимодействует с клиентами, получая задания, а с другой стороны, работает с базами данных, подбирая данные, необходимые для обработки
Сервер удаленного доступа	Обеспечивает сотрудникам, работающим дома торговым агентам, служащим филиалов, лицам, находящимся в командировках, возможность работы с данными сети
Телефонный сервер	Предназначен для организации в локальной сети службы телефонии. Этот сервер выполняет функции речевой почты, автоматического распределения вызовов, учет стоимости телефонных разговоров, интерфейса с внешней телефонной сетью. Наряду с телефонией сервер может также передавать изображения и сообщения факсимильной связи
Почтовый сервер	Предоставляет сервис в ответ на запросы, присланные по электронной почте
Терминальный сервер	Объединяет группу терминалов и упрощает переключения при их перемещении
Коммуникационный сервер	Выполняет функции терминального сервера, но при этом также осуществляет и маршрутизацию данных
Видео-сервер	Снабжает пользователей видеоматериалами, обучающими программами, видеоиграми, обеспечивает электронный маркетинг. Имеет высокую производительность и большую память
Факс-сервер	Обеспечивает передачу и прием сообщений в стандартах факсимильной связи
Сервер защиты данных	Содержит широкий набор средств обеспечения безопасности данных и, в первую очередь, идентификации паролей

3. Сетевой адаптер (сетевая карта) относится к периферийным устройствам персонального компьютера, непосредственно взаимодействующим со средой передачи данных, которая прямо или через другое коммуникационное оборудование связывает его с другими компьютерами. Сетевые адаптеры вместе с сетевым программным обеспечением способны распознавать и обрабатывать ошибки, которые могут возникнуть из-за электрических помех, коллизий или плохой работы оборудования.

4. Повторители и концентраторы. Основная функция повторителя (repeater), как это следует из его названия, — повторение сигналов, поступающих на его порт. Повторитель улучшает электрические характе ристики сигналов и их синхронность, и за счет этого появляется возможность увеличивать общую длину кабеля между самыми удаленными в сети узлами.

Многопортовый повторитель часто называют *концентратором* (concentrator) или *хабом* (hub), что отражает тот факт, что данное устройство реализует не только функцию повторения сигналов, но и концентрирует в одном центральном устройстве функции объединения компьютеров в сеть.

Концентратор может выполнять следующие дополнительные функции:

- **объединение сегментов сети с различными физическими средами в единый логический сегмент;**
- **автосегментация портов — автоматическое отключение порта при его некорректном поведении (повреждение кабеля, интенсивная генерация пакетов ошибочной длины и т.п.);**
- **поддержка между концентраторами резервных связей, которые используются при отказе основных;**
- **защита передаваемых по сети данных от несанкционированного доступа (например, путем искажения поля данных в кадрах, повторяемых на портах, не содержащих компьютера с адресом назначения) и др.**

Основные операции, выполняемые сетевыми адаптерами

Таблица 2

Наименование операции	Характеристика операции
Прием и передача данных	**Данные передаются из ОЗУ ПК в адаптер или из адаптера в память ПК через программируемый канал ввода/вывода, канал прямого доступа или разделяемую память**
Буферизация	**Для согласования скорости обработки различными компонентами ЛВС используются буфера. Буфер позволяет адаптеру осуществлять доступ ко всему пакету данных**
Формирование пакета данных	**Сетевой адаптер делит данные на блоки в режиме передачи и оформляет в виде кадра определенного формата или соединяет их в режиме приема данных. Кадр включает несколько служебных полей, среди которых имеется адрес компьютера назначения и контрольная сумма кадра, по которой сетевой адаптер станции назначения делает вывод о корректности доставленной по сети информации**
Доступ к каналу связи	**Сетевой адаптер использует набор правил, обеспечивающих доступ к среде передачи и позволяющих выявить конфликтные ситуации и контроль состояния сети**
Идентификация адреса	**Сетевой адаптер идентифицирует свой адрес в принимаемом пакете. Физический адрес адаптера может определяться установкой переключателей, храниться в специальном регистре или ПЗУ адаптера**
Кодирование и декодирование данных	**Сетевой адаптер формирует электрические сигналы, используемые для представления данных в процессе передачи их по каналам связи**
Передача и прием импульсов	**В режиме передачи сетевой адаптер передает закодированные электрические импульсы данных в канал связи, а при приеме направляет импульсы на декодирование**

5. Мосты и коммутаторы делят общую среду передачи данных на логические сегменты. Логический сегмент образуется путём объединения нескольких физических сегментов (отрезков кабеля) с помощью одного или нескольких концентраторов. Каждый логический сегмент подключается к отдельному порту моста или коммутатора. При поступлении кадра на какой-либо из портов мост или коммутатор повторяет этот кадр, но не на всех портах, как это делает концентратор, а только на том порту, к которому подключен сегмент, содержащий компьютер-адресат.

Основное отличие мостов и коммутаторов состоит в том, что мост обрабатывает кадры последовательно (один за другим), а коммутатор — параллельно (одновременно между всеми парами своих портов).

Мост (bridge) – ретрансляционная система, соединяющая каналы передачи данных.

Коммутатор (switching hub) — это многопортовый и многопроцессорный мост, обрабатывающий кадры со скоростью, значительно превышающей скорость работы моста.

Маршрутизатор ***(router) - ретрансляционная система соединяющая две коммуникационные сети либо их части.***

Шлюз ***(gateway) — ретрансляционная система, обеспечивающая взаимодействие информационных сетей.***

Каналы связи ***— это физическая среда для передачи информации между рабочими станциями или узлами сети.***

6. Маршрутизаторы обмениваются - информацией об изменениях структуры сетей, трафике и их состоянии. Благодаря этому выбирается оптимальный маршрут следования блока данных в разных сетях от абонентской системы отправителя к системе-получателю. Маршрутизаторы обеспечивают также соединение административно независимых коммуникационных сетей.

7. Шлюз является наиболее сложной - ретрансляционной системой, обеспечивающей взаимодействие сетей с различными наборами протоколов всех семи уровней модели открытых систем. Шлюзы оперируют на верхних уровнях модели OSI (сеансовом, представительском и прикладном) и представляют наиболее развитый метод подсоединения сетевых сегментов и компьютерных сетей. Необходимость в сетевых шлюзах возникает при объединении двух систем, имеющих различную архитектуру, т.к. в этом случае требуется полностью переводить весь поток данных, проходящих между двумя системами.

В качестве шлюза обычно используется выделенный компьютер, на котором запущено программное обеспечение шлюза и производятся преобразования, позволяющие взаимодействовать нескольким системам в сети.

8. Каналы связи позволяют быстро надежно передавать информацию между различными устройствами локальной вычислительной сети.

Выделяют следующие виды каналов связи, представленные на рис. 2.


КАНАЛЫ СВЯЗИ
Кабельные
Беспроводные
Оптические
Электрические
Радиосвязь
Оптоволокон-ный кабель
Витая пара
Связь в микроволно-вом диапазоне
Коаксиальный кабель
Инфракрасная связь

2. Файловые серверы

Файловый сервер предоставляет центральный ресурс в сети для хранения и обеспечения совместного доступа к файлам пользователям сети. При необходимости использования важного файла, к которому должны иметь доступ многие пользователи, например плана проекта, пользователи могут получать удаленный доступ к нему на файловом сервере, а не перемещать его с компьютера на компьютер. Если пользователям сети требуется доступ к одним и тем же файлам и приложениям или если в организации необходимо централизованное управление файлами и резервным копированием, этот компьютер следует настроить как файловый сервер.

В роли файлового сервера используются многие функции, уже установленные в составе операционной системы Windows Server 2003, например файловая система NTFS, дефрагментатор диска и DFS-имена. По умолчанию в роли файлового сервера устанавливаются следующие функции.

- **Управление файловым сервером**

Консоль управления файловым сервером служит центральным средством управления файловым сервером. Средства управления файловым сервером обеспечивают создание общих ресурсов и управление ими, настройку пределов квоты, создание отчетов об использовании хранилищ, репликацию данных на файловый сервер и с сервера, управление сетями SAN (Storage Area Network) и совместное использование файлов с системами UNIX и Macintosh.

- **Отчеты хранилища**

Отчеты хранилища позволяют анализировать степень использования дискового пространства на сервере. Например, предусмотрено создание отчетов по требованию и запланированных отчетов для идентификации дублированных файлов. После этого дублированные файлы можно удалить, чтобы возвратить дисковое пространство.

- **Квоты и фильтрация файлов**

Квоты позволяют ограничивать размер тома или поддерева папки. Windows можно настроить таким образом, чтобы выводилось оповещение о достижении предела квоты. Фильтрация файлов позволяет предотвратить сохранение в папке или на томе файлов определенных типов. Фильтрация файлов гарантирует, что пользователи не будут сохранять на сервере второстепенные файлы или файлы, при сохранении которых могут нарушаться законы об охране интеллектуальной собственности.

- **Управление DFS**

Оснастка управления DFS обеспечивает управление репликацией данных с серверов подразделений на серверы центров обработки данных. Данные могут архивироваться централизованным образом, благодаря чему отпадает необходимость выполнять архивирование на местах. С помощью оснастки управления DFS можно группировать общие папки, расположенные на разных компьютерах, и представлять их пользователю в виде виртуального дерева папок, так называемого *пространства имен*.

- **Репликация DFS**

Репликация DFS — модуль репликации по состояниям в режиме с несколькими хозяевами, поддерживающий настройку расписания репликации и регулировку полосы пропускания. При репликации DFS используется алгоритм сжатия, известный как RDC (Remote Differential Compression), который может эффективно применяться для обновления файлов по сетям, имеющим ограниченную пропускную способность. При его использовании выполняется репликация только измененных блоков обновляемых файлов.

- **Диспетчер хранилища для сетей SAN**

Диспетчер хранилища для сетей SAN позволяет обеспечить хранилище одним или несколькими оптическими каналами или подсистемами хранилища iSCSI в сети SAN (Storage Area Network).

- **Службы MSNFS (Microsoft Services for Network File System)**

Службы MSNFS обеспечивают общий доступ к файлам в смешанной среде компьютеров, операционных систем и сетей.

- **Службы для Macintosh**

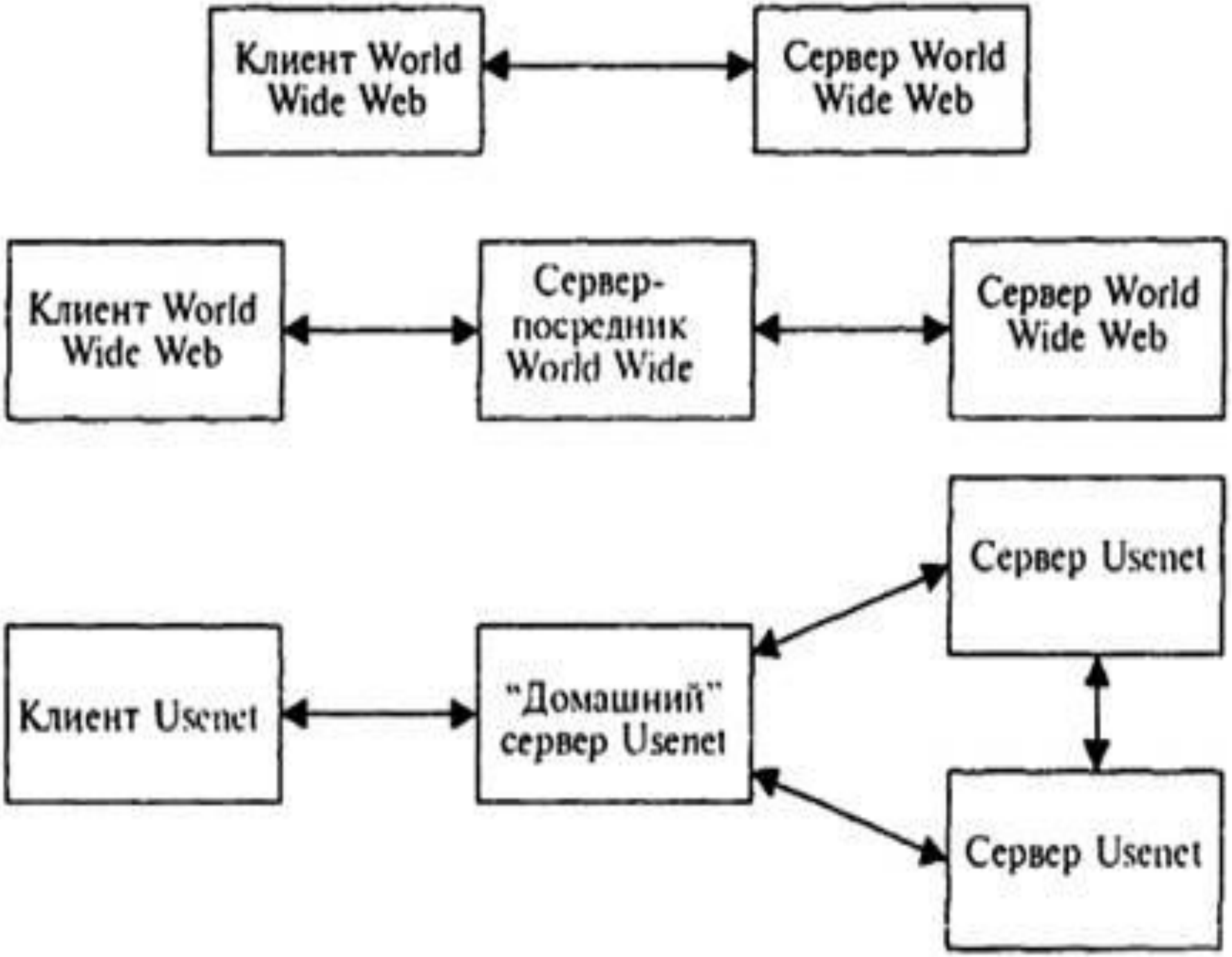
Службы Macintosh позволяют клиентским компьютерам с операционными системами Windows и Macintosh совместно использовать файлы и принтеры и осуществлять удаленный доступ к сети Microsoft.

3. Сетевое взаимодействие «клиент-сервер»

В основу взаимодействия компонентов информационных сервисов Сети в большинстве случаев положена модель «клиент-сервер». Как правило, в качестве клиента выступает программа, которая установлена на компьютере пользователя, а в качестве сервера - программа, установленная у провайдера. В данном контексте под провайдером понимаем организацию или частное лицо, которые ведут (поддерживают) информационные ресурсы.

При этом возможны два варианта организации самой информационной системы, которая обеспечивает доступ к информационному ресурсу. Большинство систем Интернет построены по принципу взаимодействия "каждый с каждым", например система World Wide Wed, т.е. каждый пользователь может напрямую взаимодействовать с каждым сервером без посредников. Такой подход позволяет упростить всю технологическую схему построения системы, однако приводит к порождению большого трафика в Сети. Альтернативный вариант построения системы, например системы Usenet, когда пользователь может взаимодействовать только со «своим» сервером и не может обратиться к произвольному серверу в Сети.

В ряде случаев возможен выбор между первым способом реализации информационного обслуживания и вторым, например, это возможно в службе доменных имен DNS. Администратор сервера может настроить его для работы через другой сервер или непосредственно с программами-клиентами. Аналогично настраиваются и специальные серверы-посредники для различных информационных серверов Интернет. Несколько таких схем показано на рис. 3.


Клиент World Wide Web
Сервер World Wide Web
Клиент World Wide Web
Сервер-посредник World Wide
Сервер World Wide Web
Сервер Usenet
Клиент Usenet
"Домашний" сервер Usenet
Сервер Usenet

Недостатки протоколов

Проблемы защиты информации являются "врожденными" практически для всех протоколов и служб Интернет.

Система имен доменов (Domain Name System - DNS) представляет собой распределенную базу данных, которая преобразует имена пользователей и хостов в IP-адреса и наоборот. DNS также хранит информацию о структуре сети компании, например количестве компьютеров с IP-адресами в каждом домене. Одной из проблем DNS является то, что эту базу данных очень трудно "скрыть" от неавторизованных пользователей. В результате, DNS часто используется хакерами как источник информации об именах доверенных хостов.

FTP (File Transfer Protocol) обеспечивает передачу текстовых и двоичных файлов, поэтому его часто используют в Интернет для организации совместного доступа к информации. На FTP-серверах хранятся документы, программы, графика и любые другие виды информации. Некоторые FTP-серверы ограничивают доступ пользователей к своим архивам данных с помощью пароля, другие же предоставляют свободный доступ (так называемый анонимный FTP-сервис). Если вы используете опцию анонимного FTP для своего сервера, то должны быть уверены, что на нем хранятся только файлы, предназначенные для свободного распространения.

Sendmail - популярная в Интернет программа электронной почты, использующая для своей работы некоторую сетевую информацию, такую как IP-адрес отправителя. Перехватывая сообщения, отправляемые с помощью Sendmail, хакеры могут использовать эту информацию для нападений, например для спуфинга (подмены адресов).

SMTP (Simple Mail Transfer Protocol) - протокол, позволяющий осуществлять почтовую транспортную службу Интернет. Одна из проблем безопасности, связанная с этим протоколом, состоит в том, что пользователь не может проверить адрес отправителя в заголовке сообщения электронной почты. В результате хакер может послать в вашу сеть большое количество почтовых сообщений, что приведет к перегрузке и блокированию работы вашего почтового сервера.

Telnet - сервис Интернет, при осуществлении которого пользователи должны регистрироваться на сервере Telnet, вводя свое имя и пароль. После аутентификации пользователя его рабочая станция функционирует в режиме «тупого» терминала, подключенного к внешнему хосту. С этого терминала пользователь может вводить команды, которые обеспечивают ему доступ к файлам и возможность запуска программ. Подключившись к серверу Telnet, хакер может сконфигурировать его программу таким образом, чтобы она записывала имена и пароли пользователей.

Как уже говорилось, стек TCP/IP представляет собой набор протоколов, которые используются в Интернет и интрасетях для передачи пакетов между компьютерами. При передаче информация заголовков пакетов может подвергнуться нападению хакеров. Например, хакеры могут подменить адрес отправителя в своих пакетах, после чего они будут смотреться как пакеты, передаваемые авторизованным клиентом.

Серверы, предоставляющие свои аппаратные ресурсы. Сервер печати, почтовый сервер. Основные принципы работы

1. Серверы, предоставляющие свои аппаратные ресурсы

Термин "сервер" толкуют по-разному. Иногда его относят к оборудованию, а иногда - к ПО. В некотором смысле оба этих определения относятся к архитектуре, подготовленной к получению запросов извне и отвечающей на эти запросы путем выдачи информации заданного типа. Разумеется, в обоих случаях ядром системы является соответствующее ПО.

Когда об оборудовании говорят как о сервере, обычно имеют в виду, что на нем работает одна или более серверных программ, что он может быть предназначен для той или иной роли и, возможно, состоит из компонентов, обеспечивающих высокую степень готовности. Вообще говоря, слово "сервер" имеет тот же корень, что и "сервис". Таким образом, с точки зрения аппаратных средств сервер - это компьютер, который способен оказывать некоторые услуги другим, подсоединенным к нему компьютерам. Подразумевается, что компьютеры каким-то образом связаны с сервером и друг с другом (рис. 1).

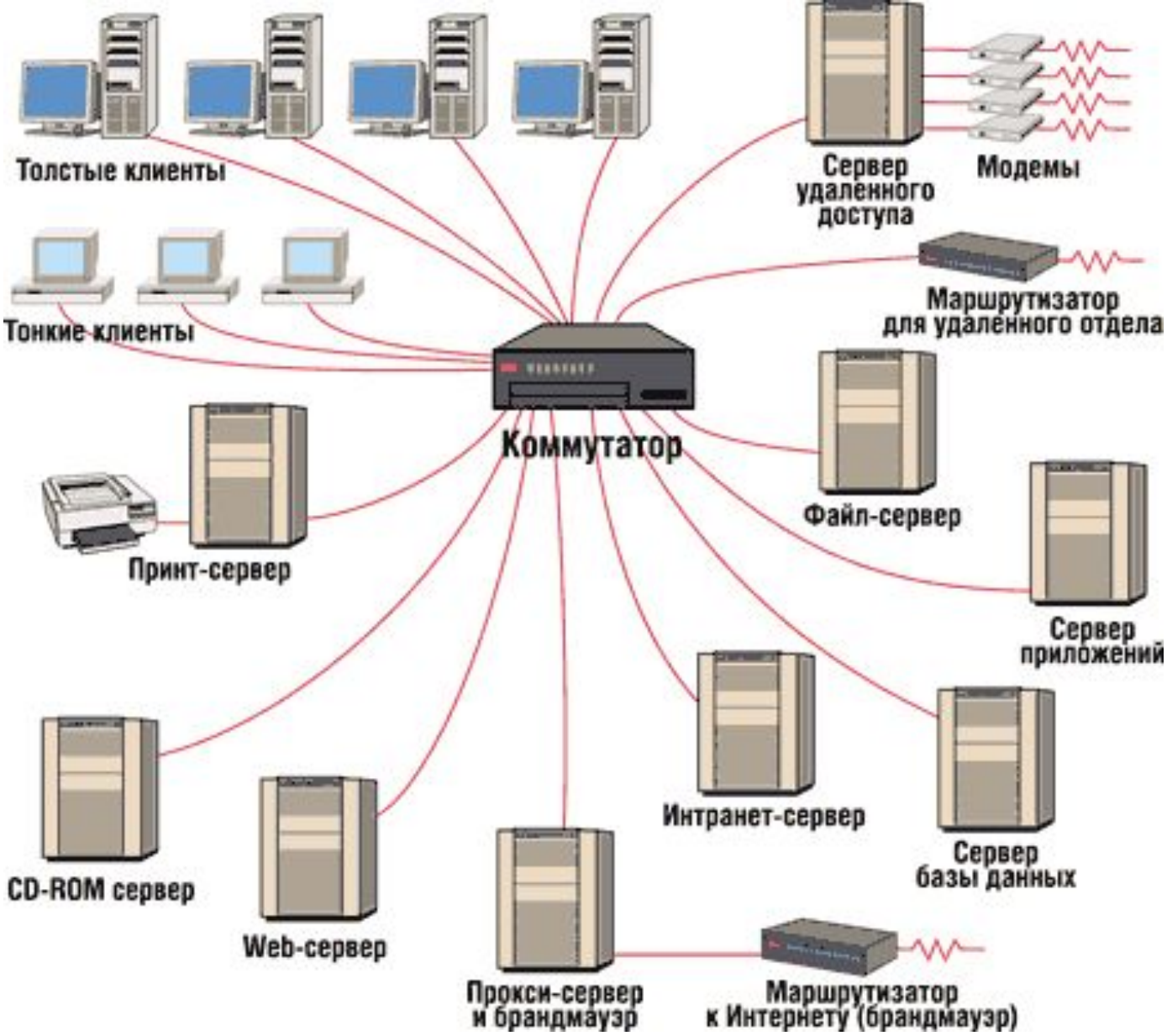


Рис. 1. Различные серверы в локальной сети

Правильный выбор сервера для организации - нелегкая задача. Широкий ассортимент серверных систем требует от руководителей ИТ-служб реалистичной оценки требований к их вычислительной мощности, масштабируемости, надежности и степени готовности. Они должны четко сформулировать, каковы будут требования к серверам, изучить возможности сервисной поддержки, а также определить будущие затраты на модернизацию. Кроме того, надо хорошо ориентироваться в разнообразии предлагаемой на рынке продукции.

Некоторые сегодняшние серверы берут свое начало в моделях, использовавшихся на протяжении уже многих лет; иные же представляют собой результат их развития, часто будучи отмечены новыми, удобными в маркетинге именами. В верхнем сегменте этого рынка царит атмосфера конкуренции и новаций, а нижний может привести в замешательство множеством имен категорий, которые иногда изобретаются лишь для того, чтобы дифференцировать продукт от его ближайшего конкурента.

Серверы можно классифицировать, например, по классу задач, который на нем выполняется, или по количеству обслуживаемых клиентов. В соответствии со вторым методом различают серверы масштаба:

- **рабочей группы (workgroup);**
- **отдела (department);**
- **средних организаций (midrange);**
- **предприятия (enterprise).**

Нужно сказать, что, поскольку в рамках каждого типа конфигурация серверов довольно значительно варьируется, четких границ между ними установить нельзя. Мощные компьютеры младшего класса могут выполнять роль серверов начального уровня в старшем смежном классе и наоборот. Тенденция к размыванию границ в последнее время настолько усилилась, что чаще всего рассматриваются серверы только трех классов: для рабочих групп, отделов и предприятий. Кроме того, по стоимости серверы можно подразделить на системы высокого, среднего и начального класса.

Надо отметить, что классификаций серверов существует довольно много, причем все они в той или иной степени перекрываются. Так, фирмы-производители часто подразделяют выпускаемые серверы по типу исполнения: сверхтонкие (blade), классические напольные (tower), оптимизированные для установки в стойку (rack) и с высокой степенью масштабируемости (super scalable).

Перевод слова blade как "лезвие", безусловно, не совсем точен. Видимо, этот образ навеян кухонными ножами, хранимыми в специальной подставке. Часто применяют также термин "сверхтонкий". Идея подобных компьютеров (рис. 2) заключается в том, чтобы упаковать в одной стойке как можно больше независимых систем, - по сути, это логическое развитие подхода, начало которому было положено просто тонкими серверами высотой 1U. В этом случае не только экономится место, отводимое под каждый сервер, но и уменьшается энергопотребление.

Рис. 2. Blade-сервер

Напольные серверы (рис. 3) обычно представляют собой самодостаточную систему (all-in-one, "все в одном"). Они обеспечивают высокую гибкость при размещении компонентов в корпусе и легко наращиваемы. Серверы для установки в стойку (рис. 4) предназначены для консолидации серверов в центрах обработки данных и использования их с внешними подсистемами памяти. Они могут эффективно применяться для кластерных решений, когда сами серверы, внешняя память и дополнительные устройства размещаются в тех же стойках. Серверы с высокой степенью масштабируемости обычно предназначены для крупных предприятий и способны обеспечить решение практически любых задач корпорации.

Рис. 3. Напольный сервер

Рис. 4. Сервер для установки в стойку

2. Принт-серверы, общие сведения

Сегодня принтеры, непосредственно подключенные к сети, стали традиционным офисным оборудованием. Они обеспечивают быструю печать и простую установку независимо от местоположения, что так важно администраторам сети. Однако в организациях с локальной сетью малого и среднего масштаба самым распространенным вариантом совместной сетевой печати остается персональный принтер, подключенный к одному из ПК (рис. 5). Главное достоинство этого варианта - экономичность. Однако следует помнить, что при таком способе печати задействуются ресурсы того ПК, к которому подсоединен принтер, и производительность компьютера значительно снижается.

Рис. 5. Использование локального принтера

Так что при больших объемах сетевой печати использовать данный ПК по прямому назначению становится весьма затруднительно. Выделение одного компьютера лишь для сетевой печати, равно как и приобретение специального сетевого принтера, часто неоправданно с финансовой точки зрения, особенно если объемы печати не превышают возможностей используемого печатающего устройства. В решении этой проблемы и призваны помочь так называемые принт-серверы, или серверы печати.

Принт-серверы представляют собой небольшие сетевые устройства, к которым могут подключаться один или несколько принтеров (рис. 6). Они бывают двух типов: внешние и внутренние (рис. 7). Последние выполнены в виде платы и, как правило, могут использоваться только с определенными печатающими устройствами. Наиболее заметное ограничение внутренних сетевых плат заключается в том, что их должен поддерживать конкретный принтер. А это означает, что если принтер изготовлен до появления таких устройств или если принтер новый, но внутренние карты он не поддерживает, то придется применить внешнее устройство сетевой печати.

Рис. 6. Принт-сервер на два принтера

Рис. 7. Внешний и встраиваемый принт-серверы

Кроме того, внутренняя плата обслуживает только один принтер. Внешние серверы печати способны обслуживать одновременно несколько принтеров, и таким образом удается сэкономить на дополнительных портах. Когда единое устройство работает со многими принтерами, это также сокращает потребности в конфигурировании и необходимые ресурсы. Например, для каждого из принтеров достаточно одного IP-адреса, в то время когда принтерам с внутренней платой придется каждому выделить по отдельному адресу.

Часто внутренние сетевые платы имеют ограничения по числу поддерживаемых сетевых протоколов и их одновременной поддержке: бывает, что нужный протокол не поддерживается или два протокола не могут работать одновременно.

Внешний сервер печати обеспечивает прозрачное, совместное использование сетевого принтера. Так же, как и сетевой компьютер с подключенным к нему принтером, внешний сервер печати обладает собственным сетевым подключением, а принтер (или принтеры) подключается непосредственно к серверу.

Преимущества внешних принт-серверов

Принт-серверы обычно рекомендуется устанавливать в средних и крупных сетях, где требуется удаленное управление печатающими устройствами. Особенно эффективно их применение в тех случаях, когда в сети есть много принтеров, размещенных на разных этажах и даже в разных зданиях, и необходимо предоставить администратору максимум возможностей для управления, а пользователю - максимум удобств для работы с ними.

Экономия средств

Для печати с компьютера или файлового сервера с подключенным принтером нужно иметь этот компьютер или сервер, а они достаточно дороги. Затраты существенно возрастают, если множество принтеров, нуждающихся в доступе к сети, распределены по всему офису. Даже самый простой компьютер существенно дороже, чем мощный многопротокольный сервер печати, причем последний поддерживает одновременно несколько принтеров. Кроме того, сервер печати позволяет сократить время, затрачиваемое на поддержку и сервис (обслуживание компьютера требует существенно больше времени).

Разгрузка основного процессора

Обработка заданий печати занимает много процессорного времени компьютера, к которому подключен принтер. Если при этом ПК выполняет другие задания или предоставляет файлы в общий доступ, печать способна существенно повлиять на его способность выполнять другие задачи. Производительность печати также оставляет желать лучшего, если компьютер одновременно выполняет несколько заданий, существенно нагружающих процессор.

Причина большой нагрузки на процессор заключается в так называемом механизме программного посимвольного ввода-вывода. Суть его в том, что, когда сервер отправляет принтеру задание на печать, для каждого символа инициируется процессорное прерывание. При этом не имеет значения, какой компьютер используется - устаревшая XT-шка или самая быстрая машина на базе Pentium 4. Требования к серверу существенно возрастают, если он обслуживает несколько принтеров.

В сервере печати вместо программного посимвольного ввода-вывода используется прямой доступ к памяти DMA (Direct Memory Access). В этом случае процессорное прерывание инициируется лишь для целых пакетов данных. Таким образом, сервер печати не только разгружает компьютер или файловый сервер, повышая эффективность выполнения ими других задач, но и позволяет добиться более быстрой сетевой печати.

Удобное размещение

При подключении принтеров к компьютеру или файловому серверу их местоположение выбирать не приходится. А поскольку серверы, например, обычно располагаются в безопасном, специально кондиционируемом помещении или поближе к администратору системы, они чаще всего недоступны большинству пользователей, которым приходится совершать дальние "прогулки", чтобы забрать отправленные на печать документы. Сетевые серверы печати, напротив, позволяют размещать принтеры в любом месте сети и именно там, где они больше всего нужны. Благодаря небольшому размеру такие устройства можно подключать непосредственно к принтеру или размещать в таких местах, где они никому не мешают.

3. Почтовый сервер – основные принципы работы

Почтовые серверы – это серверы, получающие и отправляющие электронные сообщения.

Сервер, получающий электронные сообщения, работает по протоколу POP (Post Office Protocol).

Сервер, отправляющий электронные сообщения работает по протоколу SMTP (Simple Mail Transfer Protocol).

Почтовый сервер, сервер электронной почты, мейл-сервер — в системе пересылки электронной почты так обычно называют агент пересылки сообщений (англ. *mail transfer agent, MTA*). Это компьютерная программа, которая передаёт сообщения от одного компьютера к другому. Обычно почтовый сервер работает «за кулисами», а пользователи имеют дело с другой программой — клиентом электронной почты (англ. *mail user agent, MUA*).

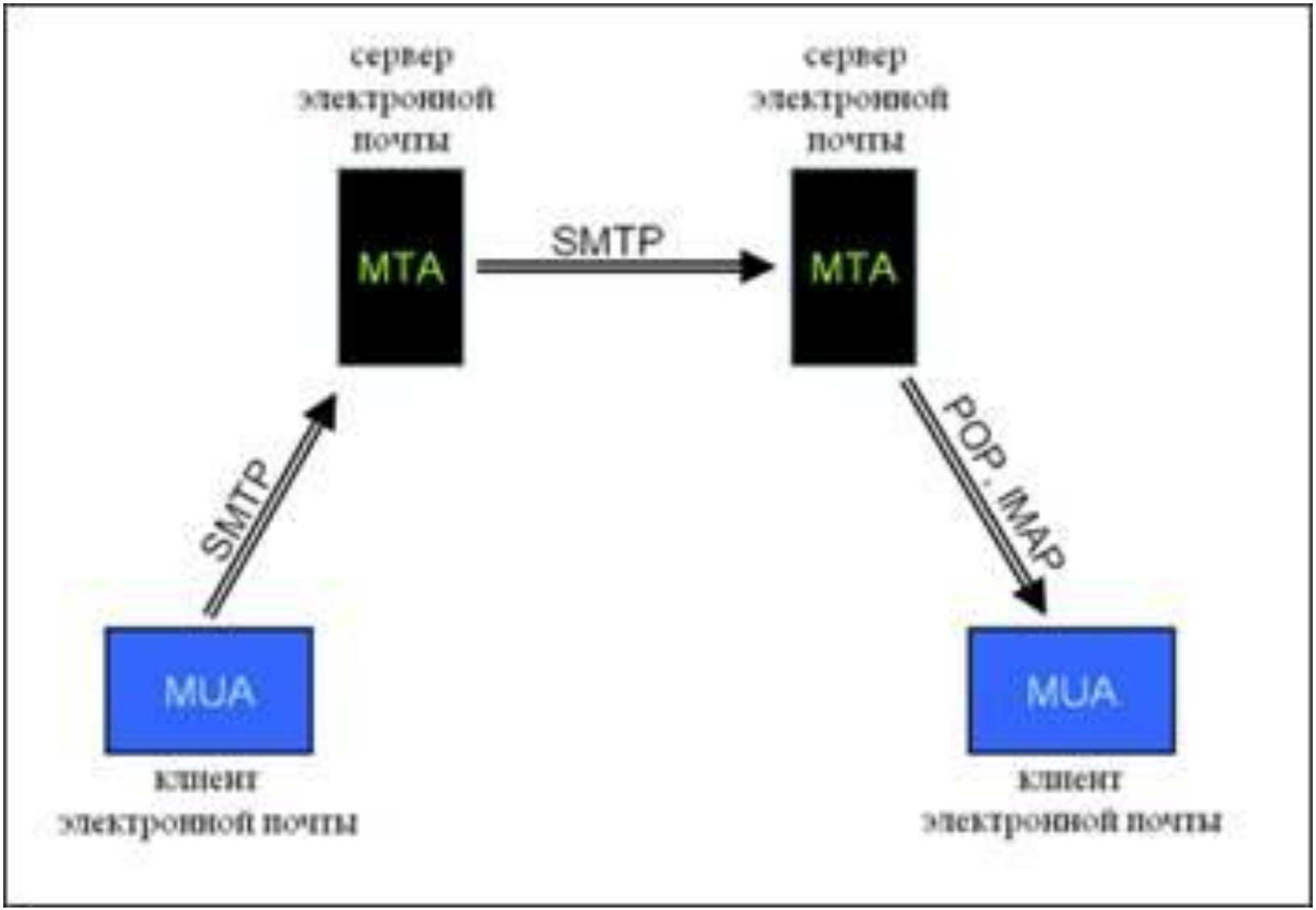
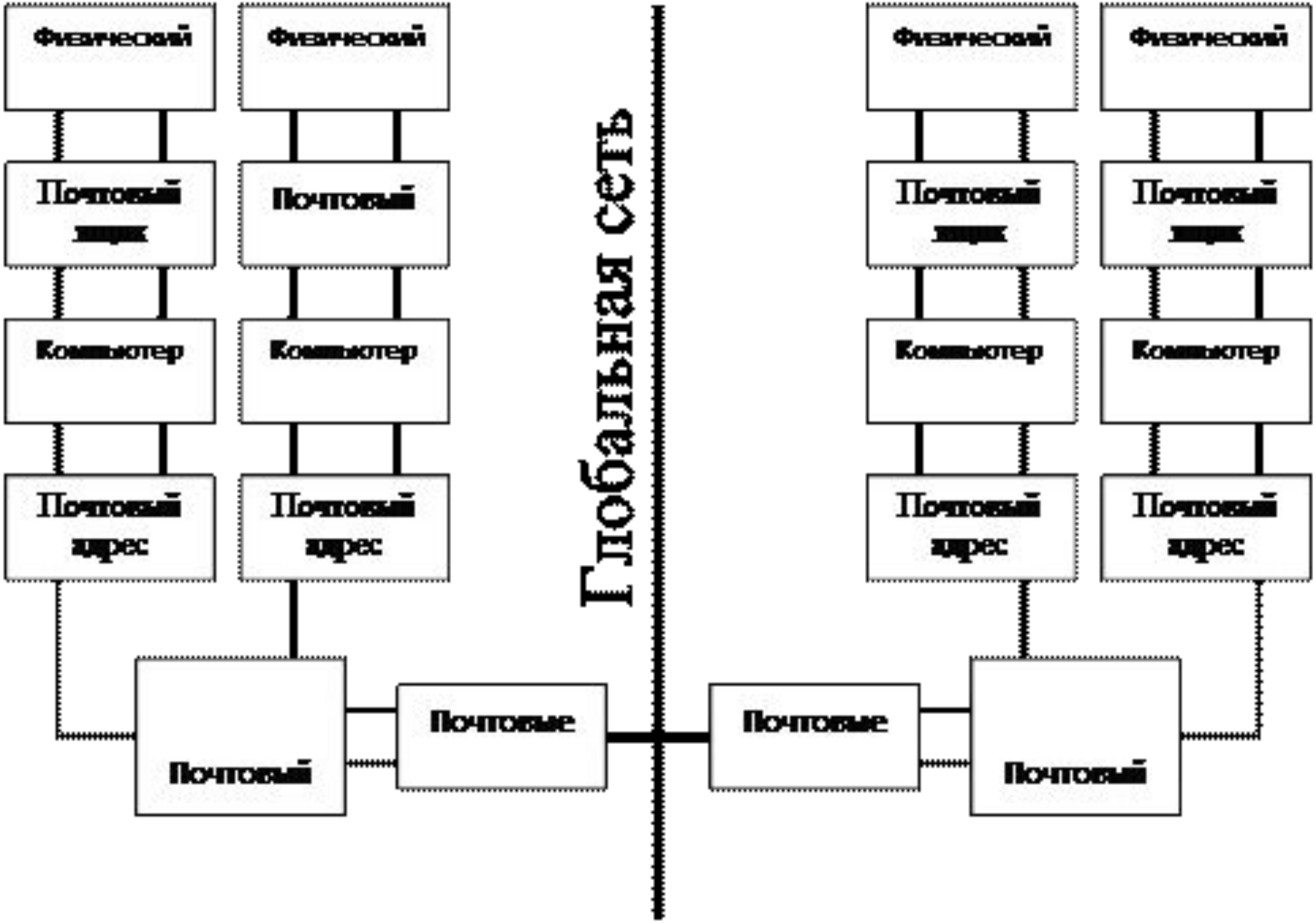


Рис. 8. Схема взаимодействия

К примеру, в распространённой конфигурации агентом пользователя является Outlook Express. Когда пользователь набрал сообщение и посылает его получателю, почтовый клиент взаимодействует с почтовым сервером, используя протокол SMTP. Почтовый сервер отправителя взаимодействует с почтовым сервером получателя (напрямую или через промежуточный сервер — релей). На почтовом сервере получателя сообщение попадает в почтовый ящик, откуда при помощи агента доставки сообщений (mail delivery agent, MDA) доставляется клиенту получателя. Часто последние два агента совмещены в одной программе, хотя есть специализированные MDA, которые в том числе занимаются фильтрацией спама.

Электронная почта

Электронная почта, как и обычная, работает с системой электронных "почтовых отделений" – почтовых серверов, которые обеспечивают пересылку писем по глобальным сетям. Они взаимодействуют с помощью почтовых протоколов, обеспечивающих пересылку и распознавание передаваемой в сети информации. Компьютеры-клиенты почтовых серверов обслуживают пользователей электронной почты. Каждый получает свой почтовый адрес и свой "почтовый ящик" на этом компьютере, т.е. область памяти, а также пароль для доступа к нему.


Физический
Физический
Физический
Физический
Почтовый ящик
Почтовый
Почтовый ящик
Почтовый ящик
Компьютер
Компьютер
Компьютер
Компьютер
Почтовый адрес
Почтовый адрес
Почтовый адрес
Почтовый адрес
Глобальная сеть
Почтовый
Почтовые
Почтовые
Почтовый

С помощью почтовой программы можно создавать сообщения, считывать их с почтового сервера, работать с адресной книгой, хранить и организовывать письма в папках "почтового ящика", готовить файлы для пересылки и преобразовывать их в нужный формат после получение и др.

С помощью почтовой программы пользователь создает сообщение адресату, задает адрес, отправляет сообщение, для чего соединяется с почтовым сервером. Во время соединения почтовый сервер запрашивает имя пользователя и его пароль. Иначе сеанс связи не состоится. После соединения подготовленная почта автоматически отправляется на сервер и далее через передачу от одного к другому почтовому серверу достигает адресата.

Сразу после отправки корреспонденции автоматически происходит получение почты клиентом. Считанные в его область памяти сообщения и файлы сортируются и раскладываются по почтовым ящикам пользователей. Адресат при загрузке своего ящика видит разложенные по папкам сообщения: новые, старые, отправленные. Он может удалять, сортировать и классифицировать их по своему разумению.

Структура адреса электронной почты

При пересылке информации большое значение имеет адресация, поскольку без нее не может быть найден получатель. Все знают печальную историю Ваньки Жукова, который отправил письмо "на деревню дедушке". Адрес обычной почты оформляются по определенным почтовым правилам.

Существующие правила оформления электронных адресов иные. Адреса электронной почты имеют более четкую логическую структуру.

Они состоят из иерархической последовательности доменов – частей, например:

user@gym2.spb.su

kimmeria@sch.spb.ru

sviend@comp.kiev.ua

Все адреса состоят из двух частей, разделенных символом @ (читается "эт"). При прочтении слева направо до этого знака отображаются имена пользователей (получателей). Часть адреса, находящаяся справа от @, определяет компьютер, подключенный к сети, город и страну или название сети, в которой пользователь зарегистрирован. Адреса делятся на части, которые называются доменами.

Что такое домен

Рассматривая домен справа налево и разбив его по точкам на отдельные слова, получим поддомены, поочередно уточняющие, где этот почтовый ящик искать. В аналогии с обычной почтой домен – это адрес (строка "Куда" на конверте), а поддомены - название страны, города, улицы, номер дома.

Домен не описывает путь, по которому следует передавать сообщение, а только объясняет, где находится адресат; точно так же адрес на почтовом конверте - это не описание дороги, по которой должен идти почтальон, чтобы доставить письмо, а место, в которое он должен в конце концов его принести.

Самый правый поддомен (в нашем случае ru) называется доменом верхнего уровня и чаще всего обозначает код страны, в которой находится сервер. Код ru - это Россия, kz – Казахстан. Каждый код состоит из двух латинских букв. Например, код uk обозначает Великобританию, и почтовый ящик с адресом mathew@montis.co.uk следует искать в английской сети JANET.

Домен верхнего уровня - не всегда код страны. В Соединенных Штатах встречаются такие, например, домены верхнего уровня, как edu - научные и учебные организации, или gov – правительственные учреждения:

lamaster@geo rge.arc.nasa.gov

Если почтовая служба видит в правой части домена поддомен такого вида, она уже знает, что адресат находится в США, поэтому код страны us не нужен. Такие обозначения сложились в американской научной сети ARPANET еще до того, как ее связали с сетями в других странах, а сейчас они сохраняются только по привычке. Как правило, во все места, которые адресуются по типу организации, можно добраться и используя код страны. Из соображений простоты и единообразия лучше пользоваться адресами с кодами стран.

Обычно такие адреса используются, если эта сеть понимает адреса в формате, отличном от RFC822. Тогда Вы пишите адрес типа имя@машина.сеть, а мост между Вашей сетью и сетью адресата преобразует его к нужному виду.

Поддомены, расположенные правее домена верхнего уровня, уточняют положение адресата внутри этого домена (внутри России для ru, среди военных организаций США для mil, или в сети BITNET для bitnet). К примеру, в адресе avg@hq.demos.ru поддомен demos обозначает организацию внутри России, а hq – группу машин внутри demos.

В адресе lamaster@george.arc.nasa.gov домен верхнего уровня gov означает, что адресат находится в одном из правительственных учреждений США, первый поддомен nasa уточняет, в каком именно - NASA, второй поддомен arc называет подразделение NASA - Ames Research Center, а george указывает на конкретную машину в этом подразделении.

Если письмо адресуется по имени сети, в которую его надо послать, адрес (домен) состоит только из домена верхнего уровня - имени сети и еще одного поддомена - имени машины в этой сети. Разбираться, где находится данная машина, выпадает на долю почтовых служб этой сети.

Когда необходимо достичь адреса, например, ux . cso . uiuc . edu , компьютер должен преобразовать его в адрес. Чтобы это сделать, Ваш компьютер начинает просить помощи у серверов (компьютеров) DNS, начиная с правой части имени и двигаясь влево. Сначала она просит локальные серверы DNS найти адрес.

Здесь существуют три возможности:

– Локальный сервер знает адрес, потому что этот адрес находится в той части всемирной базы данных, которую курирует данный сервер.

– Локальный сервер знает адрес, потому что кто-то недавно уже спрашивал о нём. Когда Вы спрашиваете об адресе, сервер DNS(Domain Name System) некоторое время держит его «под рукой» на тот случай, если чуть позже о нём спросит ещё кто-нибудь. Это значительно повышает эффективность работы системы.

– Локальный сервер не знает адрес, но знает, как его определить.

Серверы приложений. Двухзвенная и трехзвенная архитектура клиент-сервер. Общая схема сервера приложений. Интерфейс сервера приложений. Хранимые процедуры сервера приложений.

1. Двухзвенная и трехзвенная архитектура клиент-сервер

Архитектура распределенных систем определяет распределение компонентов и способы их связи. Современные распределенные системы весьма разнообразны по своей архитектуре. Рассмотрим только основные из них.

Архитектура *клиент-сервер* - это базовая модель организации распределенных систем, реализованная во всех сетевых операционных системах. В этой модели все процессы в РС делятся на две возможно перекрывающиеся группы. Процессы, реализующие некоторую службу, например, службу файловой системы или БД, называются серверами. Процессы, запрашивающие службы у серверов путем посылки запроса и последующего ожидания ответа от сервера, называются клиентами.

Рис. 9. Модель «Клиент-сервер»

Эта модель всегда была предметов споров среди разработчиков ПО, касающихся в, первую очередь, разделения функций системы между клиентом и сервером. Обычно четкого различия нет. Например, сервер распределенных баз данных (РБД) может выступать клиентом, передающим запросы на файловые серверы, отвечающие за реализацию таблиц этой БД.

Однако, рассматривая разные приложения типа клиент-сервер, для работы с БД, многие рекомендовали разделять их на три логических уровня.

- **уровень интерфейса пользователя;**
- **уровень обработки;**
- **уровень данных.**

Уровень интерфейса. Обычно реализуется на клиенте, что вполне естественно.

Уровень обработки. На этом уровне обычно реализуется основная логика приложения (функциональность).

Уровень данных. Содержит программы, которые предоставляют данные обрабатывающим их приложениям. Особым свойством этого уровня является требование сохранности (persistence).

Рис. 10. Логические уровни приложения

Кроме хранения данных, уровень данных обеспечивает поддержку целостности данных для разных приложений. Для БД поддержание целостности означает, что схемы БД, специфические условия приложений (триггеры БД), хранимые процедуры и прочее также хранятся на этом уровне.

Обычно уровень данных реализуется в форме реляционной БД. Это обеспечивает независимость данных от приложений. Применение реляционных баз данных в модели клиент-сервер помогает отделить уровень обработки от уровня данных, рассматривая данные и их обработку независимо друг от друга.

Варианты архитектуры клиент-сервер

Разделение системы на три логических уровня приводит к проблеме физического распределения приложений по отдельным компьютерам в модели клиент-сервер. Самая простая организация предполагает использование двух типов машин:

1. Клиентские машины (рабочие станции), на которых устанавливаются программы-клиенты, реализующие интерфейс.

2. Серверы, реализующие все остальное, то есть уровни обработки и данных.

Двухзвенная архитектура клиент-сервер

Один из подходов – распределение программ на два типа машин, клиенты и серверы, что приводит к физически двухзвенной архитектуре (two-tiered architecture).

Программные решения варьируются от минимизации функций интерфейса пользователя на клиенте (тонкий клиент) до передачи клиенту всей работы с пользовательским интерфейсом (толстый клиент). В обоих случаях мы отделяем от приложения графический внешний интерфейс, связанный с остальной частью приложения (находящейся на сервере) с помощью конкретного для данного приложения протокола. В этом подходе внешний интерфейс делает только то, что нужно для предоставления интерфейса приложения (рис. 11).

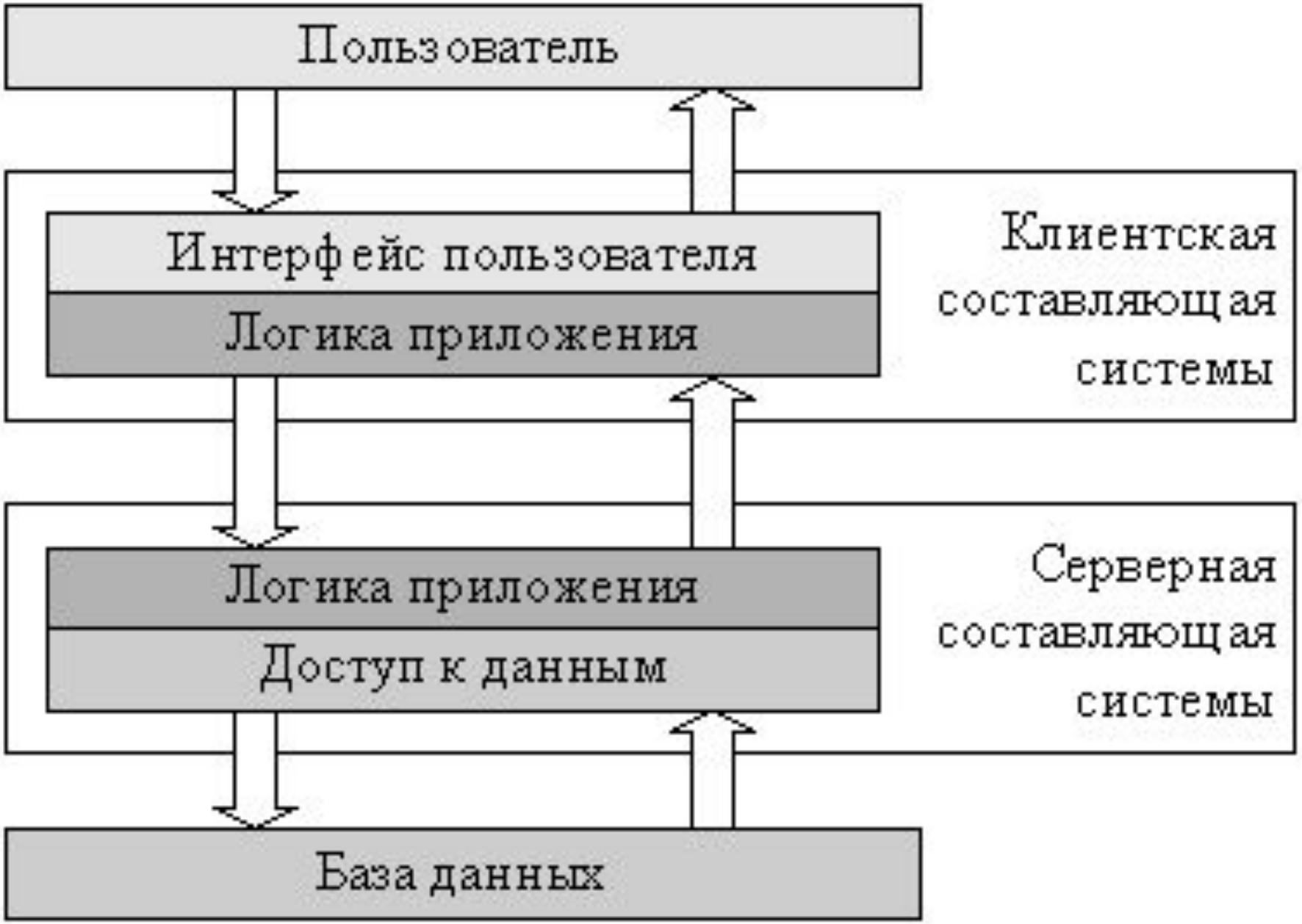


Рис. 11. Двухзвенная архитектура

Архитектуру построенных по такому принципу приложений называют клиент серверной или двухзвенной. На практике подобные системы часто не относят к классу распределенных, но формально они могут считаться простейшими представителями распределенных систем.

Трехзвенная архитектура клиент-сервер

Рассматривая только клиенты и серверы, мы упускаем то, что серверу иногда может понадобиться работать в качестве клиента. Такая ситуация приводит нас к физически трехзвенной архитектуре (three-tiered architecture).

В такой архитектуре программы, составляющие часть уровня обработки, выносятся на отдельный сервер (*сервер приложений*), но дополнительно могут частично находиться на машинах клиентов и серверов.

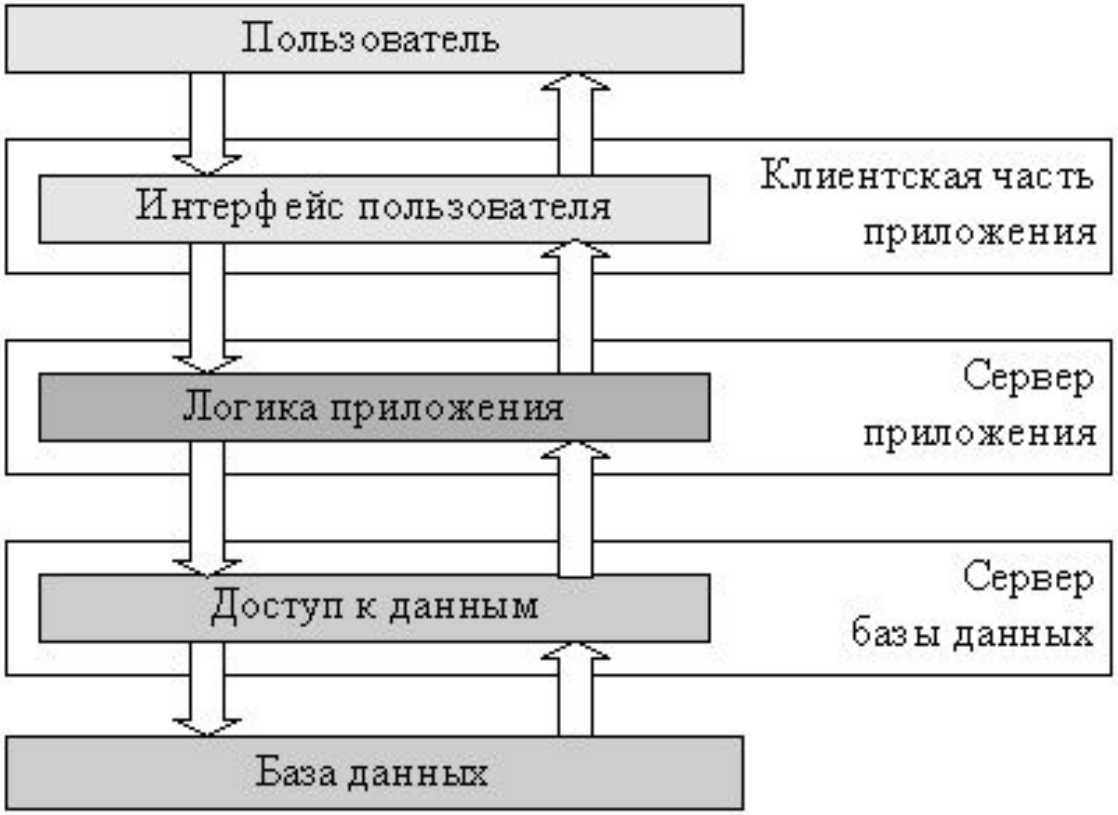


Рис. 12. Трехзвенная архитектура

2. Общая схема сервера приложений

В архитектуре "клиент/сервер" функции приложения распределены между двумя (или более) компьютерами. В соответствии с тем, каким образом это сделано, выделяются три модели архитектуры "клиент/сервер":

- **Модель доступа к удаленным данным (Remote Data Access - RDA);**
- **Модель сервера базы данных (DataBase Server - DBS);**
- **Модель сервера приложений (Application Server - AS).**

RDA-модель

В RDA-модели (рис. 13) коды компонента представления и прикладного компонента совмещены и выполнятся на компьютере-клиенте. Последний поддерживает как функции ввода и отображения данных, так и прикладные функции ("толстый" клиент).

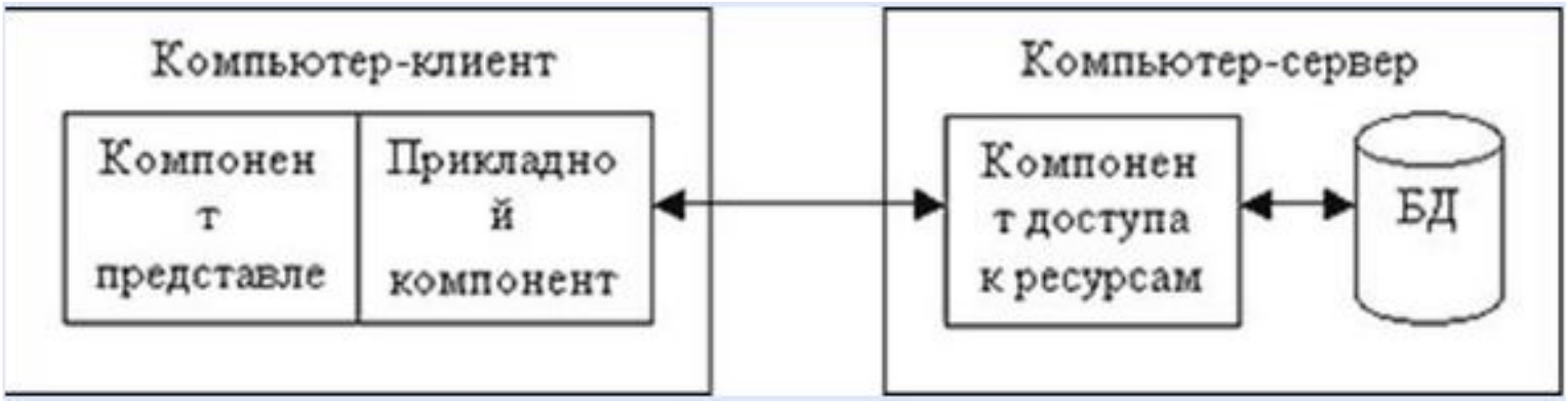


Рис. 13. RDA-модель сервера приложений

Доступ к информационным ресурсам обеспечивается, как правило, операторами специального языка (например, SQL) или вызовами функций специальной библиотеки (если имеется соответствующий API). Запросы к информационным ресурсам направляются по сети удаленному компьютеру-серверу базы данных. Последний обрабатывает и выполняет запросы и возвращает клиенту блоки данных. Говоря об архитектуре "клиент/сервер", в большинстве случаев имеют в виду именно эту модель.

DBS-модель

В DBS-модели (рис. 14) процесс, выполняемый на компьютере-клиенте, ограничивается функциями представления ("тонкий" клиент), а прикладные функции реализованы в хранимых процедурах (stored procedure), которые также называют компилируемыми резидентными процедурами, или процедурами базы данных. Они хранятся непосредственно в базе данных и выполняются на компьютере-сервере базы данных, где функционирует и компонент, управляющий доступом к данным, то есть ядро СУБД.

Рис. 14. DBS-модель сервера приложений

AS-модель

В AS-модели (рис. 15) процесс, выполняющийся на компьютере-клиенте, отвечает, как обычно, за ввод и отображение данных (то есть реализует функции первой группы). Прикладные функции выполняются группой процессов (серверов приложений), функционирующих на удаленном компьютере (или нескольких компьютерах). Доступ к информационным ресурсам, необходимым для решения прикладных задач, обеспечивается таким же способом, что и в RDA-модели.

Серверы приложений выполняются, как правило, на том же компьютере, где функционирует менеджер ресурсов, однако могут выполняться и на других компьютерах.

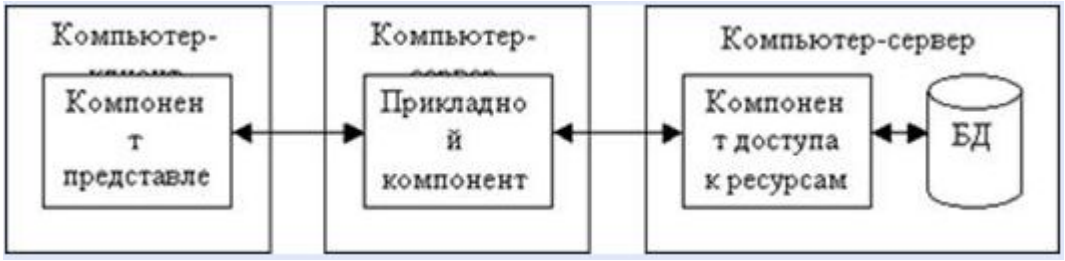


Рис. 15. AS-модель сервера приложений

Архитектура клиент-сервер предназначена для разрешения проблем файл-серверных приложений путем разделения компонентов приложения и размещение их там, где они будут функционировать более эффективно.

Особенностью архитектуры клиент-сервер является использование выделенных серверов баз данных, понимающих запросы на языке структурированных запросов SQL и выполняющих поиск, сортировку и агрегирование информации на месте без излишней перекачки данных на рабочие станции.

Большинство конфигураций клиент-сервер использует двухзвенную модель, состоящую из клиента, который обращается к услугам сервера. Для эффективной реализации такой схемы часто применяют неоднородную сеть. Как минимум, предполагается, что диалоговые компоненты PS и PL размещаются на клиенте, что позволяет обеспечить графический интерфейс.

Далее возможно разместить компоненты управления данными DS и FS на сервере, а диалог (PS, PL), логику BL и DL на клиенте. Типовое определение архитектуры клиент-сервер - приложение на клиенте, СУБД - на сервере - использует эту схему.

Переместив с клиента часть логики приложения на сервер, получим систему клиент-сервер с разделенной логикой. Часть прикладной логики может быть реализована на клиенте, а другая часть логики - в виде обработчиков событий (триггеров) и хранимых процедур на сервере БД. Такая схема при удачном разделении логики приводит к сбалансированной загрузке клиентов и сервера, но при этом затрудняется сопровождение приложений.

3. Интерфейс сервера приложений

Взаимодействие клиента и сервера приложений базируется на протоколе TCP/IP и организуется в виде обмена текстовыми строками (пакетами). Интерфейс клиента с сервером приложений является двухуровневым:

- **на нижнем уровне клиент самостоятельно устанавливает TCP/IP-соединение с сервером приложений и обменивается с ним мнемоническими командами;**
- **на верхнем уровне взаимодействие производится с помощью средств конкретной системы разработки приложений.**

Для каждой команды клиента сервер приложений возвращает подтверждение о ее завершении, что является признаком окончания выполнения команды, или сообщение об ошибке. Сервер приложений поддерживает асинхронное выполнение команд клиента.

СТРУКТУРА КЛИЕНТСКОГО ПАКЕТА

В общем виде структура клиентского пакета может быть записана в виде:

Рис. 16. Общий формат пакета клиента

Сигнатура пакета - стандартный 4-байтовый неизменяемый префикс пакета ("ASIF").

Длина пакета - беззнаковое длинное целое, показывающее размер тела пакета в байтах.

Тип пакета - однобайтовый символ, который принимает значение 'C' или 'D'.

Тело пакета - произвольная строка, содержащая команду или данные для сервера приложений (в общем виде - поток байт).

Пакеты клиента делятся на 2 вида:

- **Пакеты с данными (тип пакета - 'D'), которые содержат информацию для клиентского процесса, работающего на сервере приложений.**
- **Командные пакеты (тип пакета - 'C') содержат команды администрирования и организации работы клиента с сервером приложений".**

ЯЗЫК КОМАНД СЕРВЕРА ПРИЛОЖЕНИЙ

Элементы языка - это команды, которые делятся на следующие классы:

- **команды поддержки связи и авторизации клиента, обеспечивающие организацию сессий для работы клиентов с сервером приложений:**
  - **Команда для идентификации и аутентификации клиента при работе с сервером приложений**

LI <имя_клиента> <пароль>

  - **Команда "начать сессию"**

BS

или

BS <имя запускаемого файла>

- **Команда ES - закончить сессию.**
- **Команда DS - отключение сессии с сохранением ее идентификатора.**
- **Команда для включения сессии с соответствующим идентификатором:**

AS <идентификатор_сессии>

- **команды удаленного администрирования сервера приложений позволяют получать информацию о клиентах и их задачах на сервере приложений:**
  - **Команда LU - показать список клиентов.**
  - **Команда LT - показать список задач.**

СТРУКТУРА ОТВЕТНОГО ПАКЕТА

В общем виде структура ответного пакета сервера приложений может быть записана следующим образом:

Рис. 17. Общий формат ответного пакета сервера приложений

Сигнатура ответного пакета - стандартный 4-байтовый неизменяемый префикс пакета ("ASIF").

Длина ответного пакета - беззнаковое длинное целое, показывающее размер тела ответного пакета в байтах.

Тип ответного пакета - однобайтовый символ, который принимает одно из следующих значений: 'D', 'R' или 'E'.

Тело ответного пакета - произвольная строка, содержащая ответ сервера приложений.

Данные, получаемые в ответном пакете, представляют собой неформатированные ("сырые") данные.

4. Хранимые процедуры сервера приложений

Хранимые процедуры сервера приложений являются основным инструментом, реализующим логику прикладной системы. В рамках трехзвенной архитектуры "клиент-сервер приложений-сервер" такой подход позволяет максимально облегчить создание клиентских приложений и добиться создания так называемого "тонкого клиента". С другой стороны, достигается определенная гибкость в реализации правил обработки данных и их распределении между СУБД, которая может их поддержкой не обладать, и самими хранимыми процедурами сервера приложений.

В идеале бизнес-логика приложения реализуется частично на СУБД и частично в процедурах сервера приложений. СУБД следует сделать "ответственной" за целостность данных, максимально используя средства, предоставляемые конкретной реализацией SQL для этой СУБД, а на долю хранимых процедур сервера приложений останутся те функции, которые выходят за пределы возможностей самой СУБД и ее механизмов (триггеров, хранимых процедур БД и т.д.).

Хранимые процедуры сервера приложений создаются в рамках сервера приложений и делятся на два класса:

- **внутренние хранимые процедуры (текстовые, скриптовые);**
- **внешние хранимые процедуры (двоичные, библиотечные).**

Далее, если это специально не оговорено, под термином хранимая процедура понимается внутренняя хранимая процедура. Термин внешняя хранимая процедура всегда записывается полностью.

Внутренние хранимые процедуры

Внутренние хранимые процедуры создаются на языке хранимых процедур сервера приложений ASPL .

Для каждого пользователя сервера приложений существует три класса доступных ему функций и процедур:

- **стандартные функции сервера приложений (математические и пр.);**
- **хранимые процедуры общего использования (общие хранимые процедуры);**
- **пользовательские хранимые процедуры (частные хранимые процедуры).**

Стандартные функции реализованы стандартной библиотекой на ASPL, хранимые процедуры и функции общего назначения создаются разработчиками (могут быть внешними и внутренними), пользовательские хранимые процедуры (могут быть внешними и внутренними) создаются для каждого пользователя конкретно и могут переопределять хранимые процедуры общего назначения.

Подробнее язык хранимых процедур описан в разд. "Язык хранимых процедур сервера приложений".

Внешние хранимые процедуры

Внешние хранимые процедуры сервера приложений внедряются в архитектуру сервера приложений администратором и разделяются на пользовательские процедуры и процедуры общего использования. Они представляют собой отдельные функции динамически подключаемых (разделяемых) библиотек той операционной системы, в которой выполняется сервер приложений.

При указании имени процедуры, содержащейся в пользовательской библиотеке и библиотеке общего назначения, преимущество имеет пользовательская библиотека, т.е. будет запущена процедура именно из нее.

Внешняя хранимая процедура сервера приложений может быть создана с использованием любого языка программирования, компилятор которого позволяет создавать разделяемые библиотеки операционной системы. Для UNIX такие библиотеки, как правило, имеют расширение .so, для Windows .dll.