***Департамент общего и профессионального образования Брянской области***

***Государственное образовательное учреждение***

***Клинцовский текстильный техникум***

**ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ АВТОМАТИЗИРОВАННЫХ ИНФОРМАЦИОННЫХ СИСТЕМ**

 **РЕФЕРАТ НА ТЕМУ:**

**Технология «Клиент – сервер»**

**Студент гр. А-90\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(Петроченко А.О.)**

**Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Широкова А.Л.)**

**Клинцы – 2011**

**Содержание:**

1. **Серверы. Основные понятия серверов**
2. **Модель клиент-сервер**
3. **Классификация стандартных серверов**
4. **Вывод**

**1. Серверы. Основные понятия серверов**

Сервер (от англ. server, обслуживающий). В зависимости от предназначения существует несколько определений понятия сервер.

1. Сервер (сеть) — логический или физический узел сети, обслуживающий запросы к одному адресу и/или доменному имени (смежным доменным именам), состоящий из одного или системы аппаратных серверов, на котором выполняются один или система серверных программ

2. Сервер (программное обеспечение) — программное обеспечение, принимающее запросы от клиентов (в архитектуре клиент-сервер).

3. Сервер (аппаратное обеспечение) — компьютер (или специальное компьютерное оборудование) выделенный и/или специализированный для выполнения определенных сервисных функций.

3. Сервер в информационных технологиях — программный компонент вычислительной системы, выполняющий сервисные функции по запросу клиента, предоставляя ему доступ к определённым ресурсам.

Взаимосвязь понятий. Серверное приложение (сервер) запускается на компьютере, так же называемом "сервер", при этом при рассмотрении топологии сети, такой узел называют "сервером". В общем случае может быть так, что серверное приложение запущено на обычной рабочей станции, или серверное приложение, запущенное на серверном компьютере в рамках рассматриваемой топологии выступает в роли клиента (т.е. не является сервером с точки зрения сетевой топологии).

**2. Модель клиент-сервер**

Клиент-серверная система характеризуется наличием двух взаимодействующих самостоятельных процессов - клиента и сервера, которые, в общем случае, могут выполняться на разных компьютерах, обмениваясь данными по сети.

Процессы, реализующие некоторую службу, например службу файловой системы или базы данных, называются **серверами** (servers). Процессы, запрашивающие службы у серверов путем посылки запроса и последующего ожидания ответа от сервера, называются **клиентами** (clients) .

По такой схеме могут быть построены системы обработки данных на основе СУБД, почтовые и другие системы. Мы будем говорить о базах данных и системах на их основе. И здесь удобнее будет не просто рассматривать клиент-серверную архитектуру, а сравнить ее с другой - файл-серверной.

В файл-серверной системе данные хранятся на файловом сервере (например, Novell NetWare или Windows NT Server), а их обработка осуществляется на рабочих станциях, на которых, как правило, функционирует одна из, так называемых, "настольных СУБД" - Access, FoxPro, Paradox и т.п..

Приложение на рабочей станции "отвечает за все" - за формирование пользовательского интерфейса, логическую обработку данных и за непосредственное манипулирование данными. Файловый сервер предоставляет услуги только самого низкого уровня - открытие, закрытие и модификацию файлов. Обратите внимание - файлов, а не базы данных. Система управления базами данных расположена на рабочей станции.

Таким образом, непосредственным манипулированием данными занимается несколько независимых и несогласованных между собой процессов. Кроме того, для осуществления любой обработки (поиск, модификация, суммирование и т.п.) все данные необходимо передать по сети с сервера на рабочую станцию (*см. рис. Сравнение файл-серверной и клиент-серверной моделей)*

**Рис. Сравнение файл-серверной и клиент-серверной моделей**

В клиент-серверной системе функционируют (как минимум) два приложения - клиент и сервер, делящие между собой те функции, которые в файл-серверной архитектуре целиком выполняет приложение на рабочей станции. Хранением и непосредственным манипулированием данными занимается сервер баз данных, в качестве которого может выступать Microsoft SQL Server, Oracle, Sybase и т.п..

Формированием пользовательского интерфейса занимается клиент, для построения которого можно использовать целый ряд специальных инструментов, а также большинство настольных СУБД. Логика обработки данных может выполняться как на клиенте, так и на сервере. Клиент посылает на сервер запросы, сформулированные, как правило, на языке SQL. Сервер обрабатывает эти запросы и передает клиенту результат (разумеется, клиентов может быть много).

Таким образом, непосредственным манипулированием данными занимается один процесс. При этом, обработка данных происходит там же, где данные хранятся - на сервере, что исключает необходимость передачи больших объемов данных по сети.

**Что дает архитектура клиент-сервер?**

Посмотрим на данную архитектуру с точки зрения потребностей бизнеса. Какие же качества привносит клиент-сервер в информационную систему?

**Надежность**

Сервер баз данных осуществляет модификацию данных на основе механизма транзакций, который придает любой совокупности операций, объявленных как транзакция, следующие свойства:

* **атомарность** - при любых обстоятельствах будут либо выполнены все операции транзакции, либо не выполнена ни одна; целостность данных при завершении транзакции;
* **независимость** - транзакции, инициированные разными пользователями, не вмешиваются в дела друг друга;
* **устойчивость к сбоям** - после завершения транзакции, ее результаты уже не пропадут.

Механизм транзакций, поддерживаемый сервером баз данных, намного более эффективен, чем аналогичный механизм в настольных СУБД, т.к. сервер централизованно контролирует работу транзакций. Кроме того, в файл-серверной системе сбой на любой из рабочих станций может привести к потере данных и их недоступности для других рабочих станций, в то время, как в клиент-серверной системе сбой на клиенте, практически, никогда не сказывается на целостности данных и их доступности для других клиентов.

**Масштабируемость**

Масштабируемость - способность системы адаптироваться к росту количества пользователей и объема базы данных при адекватном повышении производительности аппаратной платформы, без замены программного обеспечения.

Общеизвестно, что возможности настольных СУБД серьезно ограничены - это пять-семь пользователей и 30-50 Мб, соответственно. Цифры, разумеется, представляют собой некие средние значения, в конкретных случаях они могут отклоняться как в ту, так и в другую сторону. Что наиболее существенно, эти барьеры нельзя преодолеть за счет наращивания возможностей аппаратуры.

Системы же на основе серверов баз данных могут поддерживать тысячи пользователей и сотни ГБ информации - дайте им только соответствующую аппаратную платформу.

**Безопасность**

Сервер баз данных предоставляет мощные средства защиты данных от несанкционированного доступа, невозможные в настольных СУБД. При этом, права доступа администрируются очень гибко - до уровня полей таблиц. Кроме того, можно вообще запретить прямое обращение к таблицам, осуществляя взаимодействие пользователя с данными через промежуточные объекты - представления и хранимые процедуры. Так что администратор может быть уверен - никакой слишком умный пользователь не прочитает то, что ему читать неположено.

**Гибкость**

В приложении, работающем с данными, можно выделить три логических слоя:

* **пользовательского интерфейса**;
* **правил логической обработки** (бизнес-правил);
* **управления данными** (не следует только путать логические слои с физическими уровнями, о которых речь пойдет ниже).

Как уже говорилось, в файл-серверной архитектуре все три слоя реализуются в одном монолитном приложении, функционирующем на рабочей станции. Поэтому изменения в любом из слоев приводят однозначно к модификации приложения и последующему обновлению его версий на рабочих станциях.

В двухуровневом клиент-серверном приложении, показанном на рисунке выше, как правило, все функции по формированию пользовательского интерфейса реализуются на клиенте, все функции по управлению данными - на сервере, а вот бизнес-правила можно реализовать как на сервере используя механизмы программирования сервера (хранимые процедуры, триггеры, представления и т.п.), так и на клиенте.

В трехуровневом приложении появляется третий, промежуточный уровень, реализующий бизнес-правила, которые являются наиболее часто изменяемыми компонентами приложения (*см. рис. Трехуровневая модель клиент-серверного приложения*)

**Рис. Трехуровневая модель клиент-серверного приложения**

Наличие не одного, а нескольких уровней позволяет гибко и с минимальными затратами адаптировать приложение к изменяющимся требованиям бизнеса.

Попробуем все вышеизложенное проиллюстрировать на маленьком примере. Предположим, в некоей организации изменились правила расчета заработной платы (бизнес-правила) и требуется обновить соответствующее программное обеспечение.

1) В файл-серверной системе мы "просто" вносим изменения в приложение и обновляем его версии на рабочих станциях. Но это "просто" влечет за собой максимальные трудозатраты.

2) В двухуровневой клиент-серверной системе, если алгоритм расчета зарплаты реализован на сервере в виде правила расчета зарплаты, его выполняет сервер бизнес-правил, выполненный, например, в виде OLE-сервера, и мы обновим один из его объектов, ничего не меняя ни в клиентском приложении, ни на сервере баз данных.

**3. Классификация стандартных серверов**

Как правило, каждый сервер обслуживает один (или несколько схожих) протоколов и серверы можно классифицировать по типу услуг, которые они предоставляют.

Универсальные серверы — особый вид серверной программы, не предоставляющий никаких услуг самостоятельно. Вместо этого универсальные серверы предоставляют серверам услуг упрощенный интерфейс к ресурсам межпроцессного взаимодействия и/или унифицированный доступ клиентов к различным услугам. Существуют несколько видов таких серверов:

* inetd от англ. internet super-server daemon демон сервисов IP — стандартное средство UNIX-систем — программа, позволяющая писать серверы TCP/IP (и сетевых протоколов других семейств), работающие с клиентом через перенаправленные inetd потоки стандартного ввода и вывода (stdin и stdout).
* RPC от англ. Remote Procedure Call удаленный вызов процедур — система интеграции серверов в виде процедур доступных для вызова удаленным пользователем через унифицированный интерфейс. Интерфейс изобретенный Sun Microsystems для своей операционной системы (SunOS, Solaris; Unix-система), в настоящее время используетстся как в большинстве Unix-систем, так и в Windows.
* Прикладные клиент-серверные технологии Windows:
	+ (D-)COM (англ. (Distributed) Component Object Model — модель составных объектов) и др. — Позволяет одним программам выполнять операции над объектами данных используя процедуры других программ. Изначально данная технология предназначена для их «внедрения и связывания объектов» (OLE англ. Object Linking and Embedding), но, в общем, позволяет писать широкий спектр различных прикладных серверов. COM работает только в пределах одного компьютера, DCOM доступна удаленно через RPC.
	+ Active-X — Расширение COM и DCOM для создания мультимедиа-приложений.

Универсальные серверы часто используются для написания всевозможных информационных серверов, серверов, которым не нужна какая-то специфическая работа с сетью, серверов, не имеющих никаких задач, кроме обслуживания клиентов. Например, в роли серверов для inetd могут выступать обычные консольные программы и скрипты.

Большинство внутренних и сетевых специфических серверов Windows работают через универсальные серверы (RPC, (D-)COM).

Сетевые службы обеспечивают функционирование сети, например серверы DHCP и BOOTP обеспечивают стартовую инициализацию серверов и рабочих станций, DNS — трансляцию имен в адреса и наоборот.

Серверы туннелирования (например, различные VPN-серверы) и прокси-серверы обеспечивают связь с сетью, недоступной роутингом.

Серверы AAA и Radius обеспечивают в сети единую аутентификацию, авторизацию и ведение логов доступа.

Информационные службы. К информационным службам можно отнести как простейшие серверы сообщающие информацию о хосте (time, daytime, motd), пользователях (finger, ident), так и серверы для мониторинга, например SNMP. Большинство информационных служб работают через универсальные серверы.

Особым видом информационных служб являются серверы синхронизации времени — NTP кроме информировании клиента о точном времени NTP-сервер периодически опрашивает несколько других серверов на предмет коррекции собственного времени. Кроме коррекции времени анализируется и корректируется скорость хода системных часов. Коррекция времени осуществляется ускорением или замедлением хода системных часов (в зависимости от направления коррекции), чтобы избежать проблем возможных при простой перестановке времени.

Файл-серверы представляют собой серверы для обеспечения доступа к файлам на диске сервера.

Прежде всего, это серверы передачи файлов по заказу, по протоколам FTP, TFTP, SFTP и HTTP. Протокол HTTP ориентирован на передачу текстовых файлов, но серверы могут отдавать в качестве запрошенных файлов и произвольные данные, например, динамически созданные веб-страницы, картинки, музыку и т. п.

Другие серверы позволяют монтировать дисковые разделы сервера в дисковое пространство клиента и полноценно работать с файлами на них. Это позволяют серверы протоколов NFS и SMB. Серверы NFS и SMB работают через интерфейс RPC.

Недостатки файл-серверной системы:

• Очень большая нагрузка на сеть, повышенные требования к пропускной способности. На практике это делает практически невозможной одновременную работу большого числа пользователей с большими объемами данных.

• Обработка данных осуществляется на компьютере пользователей. Это влечет повышенные требования к аппаратному обеспечению каждого пользователя. Чем больше пользователей, тем больше денег придется потратить на оснащение их компьютеров.

• Блокировка данных при редактировании одним пользователем делает невозможной работу с этими данными других пользователей.

• Безопасность. Для обеспечения возможности работы с такой системой Вам будет необходимо дать каждому пользователю полный доступ к целому файлу, в котором его может интересовать только одно поле

Серверы доступа к данным обслуживают базу данных и отдают данные по запросам. Один из самых простых серверов подобного типа — LDAP (англ. Lightweight Directory Access Protocol — облегчённый протокол доступа к спискам).

Для доступа к серверам баз данных единого протокола не существует, однако все серверы баз данных объединяет использование единых правил формирования запросов — язык SQL (англ. Structured Query Language — язык структурированных запросов).

Службы обмена сообщениями позволяют пользователю передавать и получать сообщения (обычно — текстовые).

В первую очередь это серверы электронной почты, работающие по протоколу SMTP. SMTP-сервер принимает сообщение и доставляет его в локальный почтовый ящик пользователя или на другой SMTP-сервер (сервер назначения или промежуточный). На многопользовательских компьютерах, пользователи работают с почтой прямо на терминале (или веб-интерфейсе). Для работы с почтой на персональном компьютере, почта забирается из почтового ящика через серверы, работающие по протоколам POP3 или IMAP.

Для организации конференций существует серверы новостей, работающие по протоколу NNTP.

Для обмена сообщениями в реальном времени существуют серверы чатов, стандартный чат-сервер работает по протоколу IRC — распределенный чат для интернета. Существует большое количество других чат-протоколов, например ICQ или Jabber.

Серверы удаленного доступа

Серверы удаленного доступа, через соответствующую клиентскую программу, обеспечивают пользователя консольным доступом к удаленной системе.

Для обеспечения доступа к командной строке служат серверы telnet, RSH, SSH.

Графический интерфейс для Unix-систем — X Window System, имеет встроенный сервер удаленного доступа, так как с такой возможностью разрабатывался изначально. Иногда возможность удаленного доступа к интерфейсу Х-Window неправильно называют «X-Server» (этим термином в X-Window называется видеодрайвер).

Стандартный сервер удаленного доступа к графическому интерфейсу Microsoft Windows называется терминальный сервер.

Некоторую разновидность управления (точнее мониторинга и конфигурирования), также, предоставляет протокол SNMP. Компьютер или аппаратное устройство для этого должно иметь SNMP-сервер.

Игровые серверы, служат для одновременной игры нескольких пользователей в единой игровой ситуации. Некоторые игры имеют сервер в основной поставке и позволяют запускать его в невыделенном режиме (то есть позволяют играть на машине, на которой запущен сервер).

Серверные решения — операционные системы и/или пакеты программ, оптимизированные под выполнение компьютером функций сервера и/или содержащие в своем составе комплект программ для реализации типичного сервисов.

Примером серверных решений можно привести Unix-системы, изначально предназначенные для реализации серверной инфраструктуры, или серверные модификации платформы Microsoft Windows.

Также необходимо выделить пакеты серверов и сопутствующих программ (например, комплект веб-сервер/PHP/MySQL для быстрой развертки хостинга) для установки под Windows (для Unix свойственна модульная или «пакетная» установка каждого компонента, поэтому такие решения редки).

В интегрированных серверных решениях установка всех компонентов выполняется единовременно, все компоненты в той или иной мере тесно интегрированы и предварительно настроены друг на друга. Однако в этом случае, замена одного из серверов или вторичных приложений (если их возможности не удовлетворяют потребностям) может представлять проблему.

Серверные решения служат для упрощения организации базовой ИТ-инфраструктуры компаний, то есть для оперативного построения полноценной сети в компании, в том числе и «с нуля». Компоновка отдельных серверных приложений в решение подразумевает, что решение предназначено для выполнения большинства типовых задач; при этом значительно снижается сложность развертывания и общая стоимость владения ИТ-инфраструктурой, построенной на таких решениях.

Прокси-сервер (от англ. proxy — «представитель, уполномоченный») служба в компьютерных сетях, позволяющая клиентам выполнять косвенные запросы к другим сетевым службам. Сначала клиент подключается к прокси-серверу и запрашивает какой-либо ресурс (например, e-mail), расположенный на другом сервере. Затем прокси-сервер либо подключается к указанному серверу и получает ресурс у него, либо возвращает ресурс из собственного кеша (в случаях, если прокси имеет свой кеш). В некоторых случаях запрос клиента или ответ сервера может быть изменён прокси-сервером в определённых целях. Также прокси-сервер позволяет защищать клиентский компьютер от некоторых сетевых атак.

#### Вывод

Таким образом, любая компьютерная сеть по сути является сетью клиент-сервер. Пользователь, подключивший свой компьютер к Интернет, будет иметь дело с сетью клиент-сервер, и даже если компьютер не имеет выхода в сеть, его программное обеспечение, да и сам он, скорее всего, организованы по схеме клиент-сервер.

**Список литературы:**

1. Валерий Коржов Многоуровневые системы клиент-сервер. Издательство Открытые системы (17 июня 1997).
2. msdn.com
3. ru.wikipedia.org