# Аннотация

Настоящая работа посвящена переносу базы данных на новую технологию с созданием клиентских частей под современные операционные системы Windows’95 и Windows NT для SQL базы данных. В пояснительной записке рассматриваются отличительные особенности технологии «клиент-сервер», по сравнению с технологией предыдущего поколения «файл-сервер». Производится описание процесса разработки клиентских частей под операционные системы Windows’95 и Windows NT для SQL базы данных. Описываются процесс разработки интерфейса пользователя под 32-разрядные операционные системы Windows’95 и Windows NT Workstation. Разрабатывается алгоритм переноса данных из старой базы данных в новую систему. Так же представлены результаты отладки и работы разработанной программы.

# СОДЕРЖАНИЕ

Введение 3

1. Теоретическая часть
   1. Обзор СУБД 5
      1. Sybase System 11 8
      2. IBM DB2 17
      3. RDMS Oracle 25
      4. Microsoft SQL Server 6.5 36
   2. Исследование предметной области 49
2. Практическая часть 62
   1. Анализ существующей программы 62
   2. Выбор платформы и программных средств 64
   3. Разработка структуры новой БД 78
   4. Перенос данных в новую базу данных 80
   5. Разработка программы 83
3. Отладка 88

Заключение 92

Литература 93

Приложение А. Описание Базы данных.

Приложение В. Листинг отлаженных программ.

# ВВЕДЕНИЕ

Для большинства средних и мелких российских предприятий информационные решения с использованием сетей персональных компьютеров являются фактическим стандартом. В тоже время, прикладное программное обеспечение, используемое этими предприятиями (такое как автоматизированные системы документооборота, системы управления промышленными и торговыми предприятиями, бухгалтерские системы и др.), создано при помощи инструментария предыдущего поколения, и не способно эффективно использовать ресурсы, предоставляемые новыми технологиями. К современным информационным системам уровня предприятия предъявляются очень высокие требования производительности, надежности, обеспечения целостности и безопасности данных (особенно при сегодняшнем развитии Internet), защиты от системных и аппаратных сбоев, масштабируемости, возможности взаимодействия с другими системами, работы в гетерогенных распределенных вычислительных сетях.

В течение последнего времени большое распространение получила новая технология построения баз данных – технология «клиент-сервер». Эта технология дает ряд неоспоримых преимуществ, по сравнению с технологией предыдущего поколения – технологией «файл-сервер». В частности, она предоставляет большие возможности по защите данных от несанкционированного доступа и разграничение прав доступа на уровне отдельных записей и полей, дает возможность работы с большими мультимедийными и нестандартными данными. Также новая технология позволяет работать как в локальных сетях, так и в глобальных и Internet, и многое другое. Системы, построенные на новой технологии «клиент-сервер», отличаются высокой степенью безопасности, территориально независимости и не требовательности к аппаратной мощности клиентских станций.

В настоящее время, в области баз данных и проектировании систем, с использованием новой технологии «клиент-сервер», ведутся работы в разных направлениях. Во-первых, продолжают вестись работы по усовершенствованию технологии и уменьшению требований к аппаратному и программному обеспечению, с одновременным увеличением производительности. Во-вторых, ведутся работы в направлении переноса всех программ, использующих технологию предыдущего поколения «файл-сервер», на новую и более современную технологию «клиент-сервер».

Весьма актуальным является проблема переноса бухгалтерских программ, рассчитанных на малые и средние предприятия и фирмы, на новую технологию. Это обусловлено тем, что область данных программ осталась почти не тронутая новой технологией. К тому же, все больше пользователей переводят свои персональные компьютеры под управление 32-разрядными операционными системами. 32-разрядные операционные системы клиентов, такие как Windows’95 и Windows NT Workstation, используют удобный в работе графический пользовательский интерфейс и предоставляют все необходимое для эффективной работы в распределенной среде.

# 1.1 Обзор СУБД

Несмотря на то, что технологии баз данных становятся все более распространенными в практике, многие разработчики прикладных систем, администраторы баз данных и конечные пользователи недостаточно хорошо понимают, что такое современные базы данных и системы управления базами данных (СУБД). Сказывается отсутствие или поверхностный уровень соответствующих курсов в высших учебных заведениях, практическое отсутствие литературы на русском языке и, наконец, привлечение к деятельности, связанной с использованием баз данных, специалистов, которые ранее работали в других областях. Обучение, проводимое коммерческими компаниями, обычно ориентируется на конкретные программные продукты и не может компенсировать отсутствие базовой подготовки.

Во всей истории вычислительной техники можно проследить две основных области ее использования. Первая область - применение вычислительной техники для выполнения численных расчетов, которые слишком долго или вообще невозможно производить вручную. Развитие этой области способствовало интенсификации методов численного решения сложных математических задач, развитию класса языков программирования, ориентированных на удобную запись численных алгоритмов, становлению обратной связи с разработчиками новых архитектур ЭВМ.

Вторая область - это использование средств вычислительной техники в автоматических или автоматизированных информационных системах. В самом широком смысле информационная система представляет собой программно-аппаратный комплекс, функции которого состоят в надежном хранении информации в памяти компьютера, выполнении специфических для данного приложения преобразований информации и/или вычислений, предоставлении пользователям удобного и легко осваиваемого интерфейса. Обычно такие системы имеют дело с большими объемами информации, и эта информация имеет достаточно сложную структуру.

Вторая область использования вычислительной техники возникла несколько позже первой. Это связано с тем, что на заре вычислительной техники возможности компьютеров по хранению информации были очень ограниченными.

Но поскольку в информационных системах требуется поддержка сложных структур данных, эти индивидуальные средства управления данными составляли существенную часть информационных систем, практически повторяясь (как программные компоненты) от одной системы к другой. Стремление выделить общую часть информационных систем, ответственную за управление сложно структурированными данными явилось первой побудительной причиной создания СУБД, которая, возможно, могла бы представлять некоторую общую библиотеку программ, доступную каждой информационной системе.

Однако очень скоро стало понятно, что невозможно обойтись такой общей библиотекой программ, реализующей над стандартной базовой файловой системой более сложные методы хранения данными.

Вообще, согласованность данных является ключевым понятием баз данных. На самом деле, если информационная система поддерживает согласованное хранение информации в нескольких файлах, можно говорить о том, что она поддерживает базу данных. Если же некоторая вспомогательная система управления данными позволяет работать с несколькими файлами, обеспечивая их согласованность, можно назвать ее системой управления базами данных. Уже только требование поддержания согласованности данных в нескольких файлах не позволяет обойтись библиотекой функций: такая система должна обладать некоторыми собственными данными (мета-данными) и даже знаниями, определяющими целостность данных.

Прежде чем приступить к конкретному обзору Систем Управления Базами Данных (СУБД), необходимо дать ряд определений терминам, использующимся для описания СУБД.

*Данные* – это информация, зафиксированная в определенной (структурированной) форме, пригодной для последующей обработки, хранения и передачи. К данным необходим «прозрачный» доступ сразу нескольких пользователей, т.е. любой пользователь должен иметь возможность получать необходимую ему информацию, модернизировать ее, заносить новую и удалять старую, причем конечный пользователь может обо всех этих операциях и не знать. *Банк Данных (БнД)* – это информационная система, включающая в себя информационные, математические (алгоритмические), программные, языковые, организационные и технические средства (аппаратная платформа и операционная система) обеспечивающие в совокупности централизованную поддержку и коллективное использование данных пользователем БнД. Языковые средства – языки, с помощью которых описывается структура данных (DDL) и языки манипулирование данными (DML ­– SQL). *База Данных (БД)* – это структурированная определенным образом совокупность данных, относящихся к конкретной задаче. БД может быть как локальная, так и распределенная. *Система Управления Базами Данных (СУБД)* представляет собой комплекс инструментальных средств (программных и языковых) реализующих централизованное управление БД и обеспечивающих доступ к данным (изменения, добавления, удаления, резервного копирования и т.д.). СУБД должна обеспечивать поиск, модификацию и сохранность данных, а также оперативный доступ (время отклика), защиту целостности данных от аппаратных сбоев и программных ошибок, разграничение прав и защита от несанкционированного доступа, поддержка совместной работы нескольких пользователей с данными.

При работе с СУБД можно выделить несколько уровней представления данных:

* *Внешний уровень (уровень конечного пользователя).* В некотором смысле это самый главный уровень. Именно с ним работает конечный пользователь или конечная программа. Пользователь воспринимает данные как совокупность некоторых взаимосвязанных полей в удобном виде, позволяющих ему решать свою задачу.
* *Концептуальный уровень (уровень программиста и администратора) –* это обобщенное представление обо всех данных, хранящихся в базе или совокупность внешних представлений. На этом уровне работают программист, создающий прикладные программы, и администратор, разрабатывающий структуру (схему) базы данных. Администратору доступна вся схема, вся информация, а конкретному программисту - только ее часть, ограниченная его привилегиями.
* *Физический уровень (уровень реализации).* На физическом уровне определяются способы хранения данных с учетом подробностей (вплоть до физического адреса) и доступа к ним. *Сервер СУБД* реализует именно этот уровень.

По логическому представлению структуры данных СУБД делятся на несколько типов: реляционные, сетевые и иерархические. Главная характеристика, определяющая тип – это используемое представление данных.

Основной структурой в *иерархических моделях данных* является «дерево». Особенности такого представления в наличии корня – единственной точки входа в дерево, и что каждый порожденный узел имеет только одного родителя. Недостатком этой системы является высокая избыточность. Одна запись БД – это совокупность деревьев. Через эту структуру нельзя построить отношение N:N (многие ко многим).

Основной структурой в *сетевых моделях данных* является «сеть». При таком представлении существует несколько входов в сеть – неоднозначность доступа к данным. Особенности такого представления: один или несколько узлов могут иметь больше одного родителя; время доступа изменяется в зависимости от исходного входа. Время доступа в сетевой структуре может быть больше, чем в иерархической структуре.

Недостатком обеих этих структур является то, что при добавлении новых вершин или установлении новых связей возникают проблемы выгрузки данных из базы, перегенерации полностью структуры, загрузка данных обратно в базу. При этом возникает вероятность потерять данные при обратной загрузке.

В основе структуры данных реляционной модели лежит мощный аппарат реляционной алгебры, реляционного исчисления и теории нормализации. При проектировании реляционной модели БД используется понятия ER-модели: сущность – объект, атрибут – свойства и связь. Связи бывают нескольких типов: 1:1, 1:N (N:1), N:N. У реляционной модели есть ряд ограничений: невозможность существования двух одинаковых записей; задание типа столбца, области значений атрибута. Достоинство реляционных СУБД, обеспечившее высокую популярность, заключается в не функциональности языка запросов. Это означает, что формулируете запрос не то, *как* надо найти данные, а *что* надо найти.

Сегодня основы современной информационной технологии составляют базы данных (БД) и системы управления базами данных (СУБД), роль которых как единого средства хранения, обработки и доступа к большим объемам информации постоянно возрастает. При этом существенным является постоянное повышение объемов информации, хранимой в БД, что влечет за собой требование увеличения производительности таких систем. Резко возрастает также в разнообразных применениях спрос на интеллектуальный доступ к информации. Это особенно проявляется при организации логической обработки информации в системах баз знаний, на основе которых создаются современные экспертные системы. Быстрое развитие потребностей применений БД выдвигает новые требования к СУБД: поддержка широкого спектра типов представляемых данных и операций над ними (включая фактографические, документальные, картинно-графические данные); естественные и эффективные представления в БД разнообразных отношений между объектами предметных областей (например, пространственно-временных с обеспечением визуализации данных); поддержка непротиворечивости данных и реализация дедуктивных БД; обеспечение целостности БД в широком диапазоне разнообразных предметных областей и операционных обстановок; управление распределенными БД, интеграция неоднородных баз данных; существенное повышение надежности функционирования БД.

На российском рынке наиболее популярны СУБД Progress (Progress software), Oracle RDBMS v7 (Oracle), Microsoft SQL Server v6.5 (Microsoft), DB2 (IBM), Sybase System 11 (Sybase).

## 1.1.1 Sybase System 11

Фирма Sybase - один из ведущих производителей промышленных СУБД, средств разработки приложений и других продуктов, реализующих технологию клиент-сервер. В конце 1995 года фирма выпустила Sybase System 11. Сейчас Sybase System 11 выпущена для Intel и основных UNIX-платформ. На Intel-платфорах работает СУБД для рабочих групп Sybase SQL Anywhere 5.0 - новая версия СУБД Watcom SQL, которая теперь имеет режим совместимости с Sybase SQL Server на уровне языка и интерфейсов.

Серверные продукты Sybase System 11 обладают архитектурой, построенной на основе продуктов и библиотек Sybase Open Client/Server. Среди основных серверных продуктов Sybase System 11:

* Sybase SQL Server - реляционная СУБД;
* Sybase MPP - расширение архитектуры Sybase SQL Server, разработанное и оптимизированное для массовой параллельной обработки. Оно обладает открытой параллельной архитектурой, предназначенной для поддержки очень больших баз данных (VLDB);
* Sybase IQ - серверный механизм построения битовых индексов для высокоскоростного выполнения сложных запросов к большим объемам данных;
* Sybase SQL Anywhere - "легкая" полнофункциональная СУБД на Intel-платформах для мобильных пользователей и небольших групп;
* Sybase Replication Server - репликационный сервер для построения распределенных систем на основе тиражирования данных;
* Sybase OmniConnect - сервер, обеспечивающий работу приложений-клиентов в "прозрачном" режиме с несколькими серверами так, как будто работа идет с одним сервером; при этом в распределенную систему могут включаться СУБД различных производителей - Sybase, Oracle, IBM и т.д.

Вспомогательные серверные продукты Sybase System 11 включают:

* Sybase Backup Server - специальный сервер для выгрузки и загрузки баз данных, не требующий остановки SQL Server и не снижающий его производительности;
* Sybase Monitor Server - в сочетании с графической клиентской частью выполняет мониторинг различных параметров состояния SQL Server;
* Sybase Replication Agent - специальные компоненты, отслеживающие изменения в данных через журналы транзакций различных СУБД для включения их в систему репликации. Replication Agent существуют, в частности, для Sybase SQL Server, Oracle, DB2, Sybase SQL Anywhere.
* Sybase Audit Server - записывает информацию о действиях пользователей в специальную базу данных, доступную для анализа.

К инструментальным средствам фирмы Sybase относятся средство быстрой разработки приложений PowerBuilder и CASE-система S-Designor, выпускаемые подразделением Powersoft. Эти средства работают со всеми основными СУБД.

Создание Sybase SQL Server 11 основывается на опыте работы предыдущих версий и содержит ряд новых возможностей:

* Масштабируемость, производительность и эффективность SQL Server 11 основывается на следующих факторах:
  + SQL Server 11 работает на множестве платформ, от персональных компьютеров до многопроцессорных суперсерверов;
  + обеспечена высокая производительность на каждой платформе благодаря тесному взаимодействию с производителями аппаратуры и оптимизации характеристик;
  + полностью симметричная многопотоковая СУБД достигает высокой пропускной способности и поддерживает большое количество пользователей.
* SQL Server обеспечивает надежность и целостность данных:
  + SQL Server содержит механизмы триггеров и процедур, декларативной ссылочной целостности, транзакций и т.д.;
  + СУБД соответствует уровню безопасности C2 NCSA (National Computer Security Council).
  + Доступность данных повышает производительность систем:
  + Sybase SQL Server программно поддерживает зеркальный журнал и зеркальную базу данных;
  + высокоскоростные средства загрузки/восстановления минимизируют влияние этих операций на работу системы.
* Открытость и соответствие стандартам:
  + SQL Server соответствует стандартам ANSI/ ISO SQL-89 и entry-level ANSI/ ISO SQL-92;
  + поддерживаются приложения в стандарте ODBC и X/Open XA;
  + SQL Server может использовать различные сетевые протоколы, что позволяет соединить клиент и сервер практически на каждой платформе.
* Управление и поддержка:
  + наличие многопотоковой архитектуры означает, что на компьютере запускается и требует управления только один процесс - СУБД;
  + для симметричной мультипроцессорной обработки можно конфигурировать количество процессоров, распределенных для СУБД;
  + имеется набор продуктов для конфигурирования областей памяти, пользователей, контроля доступа и производительности - от одной базы данных до множества сетей в масштабах предприятия.

Рассмотрим основные характеристики Sybase SQL Server.

Работа SQL Server с кэшами в памяти. Запись в журнал из кэша теперь происходит пакетами. Это снижает уровень конкуренции за доступ к ресурсу журнала и, соответственно, повышает производительность. Системный администратор может разделить кэш SQL Server на несколько именованных областей и приписать эти области различным базам данных и объектам баз данных. Имеется возможность группировать именованные области кэша так, чтобы более эффективно проходил обмен с диском большими блоками. Связывание именованных кэшей и объектов баз данных осуществляется при помощи вызова системных процедур.

При создании таблицы или индекса можно указать максимальное число строк, хранимых на странице данных или странице индекса. Эта возможность позволяет оптимизировать блокировки для часто обновляемых таблиц. SQL Server использует установленное значение при добавлении и удалении строк.

Проверка взаимных блокировок. SQL Server использует алгоритм выявления взаимных блокировок транзакций. Имеется возможность сконфигурировать то время, через которое выполняется такая проверка. Цель конфигурирования этого параметра - повышение производительности. Проверка взаимоблокировок - это достаточно длительная операция для сервера. Так как проверка запускается не сразу, выше вероятность освобождения ресурса к моменту запуска проверки. С другой стороны, излишнее увеличение времени задержки может привести к увеличенному времени реакции для приложения, выдавшего взаимоблокирующий запрос.

Несколько процессов, работающих с сетевыми соединениями. В предыдущей версии Sybase System 10 сетевым обменом занимался только один процессор в SMP-архитектуре. Это ограничивало масштабируемость сервера на симметричной мультипроцессорной архитектуре, так как было "узким местом" при увеличении числа процессоров. В версии Sybase System 11 сетевым обменом могут заниматься все процессоры.

Протокольные службы. Библиотеки Sybase обеспечивают работу серверных и клиентских компонент со множеством сетевых протоколов от разных производителей, в том числе TCP/IP, SPX/IPX, Named Pipes, DECNet и другие.

Архитектура Sybase позволяет как приложению-клиенту, так и серверу одновременно работать с несколькими интерфейсами. Например, SQL Server для Novell NetWare или Windows NT может одновременно допускать соединения через SPX и TCP/IP.

Управление транзакциями. Архитектура Sybase System 11 поддерживает как синхронную, так и асинхронную модель управления транзакциями. Модель выбирается в соответствии с требованиями предметной области.

Централизованное хранение данных и доступ к центральной БД в условиях географически распределенной системы приводят к необходимости установления соединений между центральным сервером, хранящим данные, и компьютерами-клиентами. Большинство компьютеров-клиентов отделены от центрального сервера медленными и недостаточно надежными линиями связи, поэтому работа в режиме удаленного клиента становится почти невозможной. Обмен данными на практике часто реализуется регламентной передачей файлов, либо через модемное соединение, либо "с курьером". То, что данные доставляются к месту назначения не системными средствами, а путем экспорта/импорта файлов, приводит к необходимости участия человека в процессе обмена, что влечет за собой неоперативность поступления данных и необходимость реализации внешних механизмов контроля целостности и непротиворечивости. Результатом является высокая вероятность ошибок. Кроме того, реализация всех алгоритмов обмена данными и контроля в этом случае возлагается на прикладных программистов, проектирующих АРМ. Объем работ по программированию и отладке подпрограмм обмена соответствует числу различных АРМ. Это также приводит к повышению вероятности ошибок в системе.

В современной технологии АРМ объединены в локальную сеть. АРМ-клиент выдает запросы на выборку и обновление данных, а СУБД исполняет их. Запросы клиента в соответствии с требованиями задачи сгруппированы в транзакции. Если все операции с базой данных, содержащиеся внутри транзакции, выполнены успешно, транзакция в целом выполняется успешно (фиксируется). Если хотя бы одна из операций с БД внутри транзакции не выполнилась успешно, то все изменения в БД, произведенные к этому моменту из транзакции, отменяются (происходит откат транзакции). Такое функционирование обеспечивает логическую целостность данных в базе данных.

При распределенной обработке изменения, проводимые приложением-клиентом, могут затрагивать более чем один сервер СУБД. В этом случае для поддержания целостности необходимо применение транзакционного механизма, реализуемого системными средствами, а не прикладной программой.

Производители современных промышленных СУБД обеспечивают поддержку распределенной обработки транзакций. Распределенная обработка данных основывается на синхронных или асинхронных механизмах обработки распределенных транзакций. Эти механизмы могут использоваться совместно. Выбор того или иного механизма зависит от требований конкретной подзадачи, так как каждый механизм обладает сильными и слабыми сторонами.

Фиксация. Исторически первым появился метод синхронного внесения изменений в несколько БД, называемый двухфазной фиксацией (2PC - two-phase commit). Этот механизм реализован сейчас практически у всех производителей СУБД.

Метод двухфазной фиксации состоит в том, что при завершении транзакции серверы БД, участвующие в ней, получают команду "приготовиться к фиксации транзакции". После получении подтверждений от всех серверов транзакция фиксируется на каждом из них. Все ресурсы, используемые в транзакции, остаются блокированными до тех пор, пока все компоненты транзакции могут быть атомарно завершены успешно, либо все отменены. Таким образом, в любой момент времени обеспечивается целостность данных в распределенной системе. Платой за это является требование доступности всех участвующих серверов и линий связи во время проведения транзакции и невозможность работы приложений-клиентов при недоступности, например, удаленного сервера. Кроме того, требуется высокое быстродействие линий связи для обеспечения приемлемого времени реакции у приложения-клиента.

В распределенной системе идеальным являлось бы состояние, когда каждая программа-клиент обращается только к тем серверам, которые находятся в пределах ее локальной сети, а передача данных и обеспечение целостности осуществляется системными средствами и не требуют специальных действий со стороны прикладной программы. Такое распределение функций возможно и реализуется на практике с помощью механизма асинхронного тиражирования транзакций.

Асинхронная репликация не делает линии связи более надежными или скоростными. Она перекладывает передачу данных, обеспечение их целостности и ожидание при передаче данных с прикладной программы и пользователя на системный уровень.

Асинхронная репликация, в отличие от 2РС, не обеспечивает полной синхронности информации на всех серверах в любой момент времени. Синхронизация происходит через некоторый, обычно небольшой, интервал времени, величина которого определяется быстродействием соответствующего канала связи. Для большинства задач кратковременное наличие устаревших данных в удаленных узлах вполне допустимо. Вместе с тем асинхронная репликация транзакций принципиально обеспечивает целостность данных, так как объектом обмена данными здесь является логическая единица работы - транзакция, а не просто данные из измененных таблиц.

Репликационный сервер Sybase Replication Server, входящий в состав Sybase System 11, использует асинхронную модель репликации транзакций. При разработке модели данных проектируются и правила репликации. Затем проводится конфигурирование системы. При работе прикладной программы изменения данных отслеживаются системными средствами и в соответствии с конфигурацией требуемые данные передаются в удаленную СУБД.

Репликационный сервер представляет собой отдельную задачу, запускаемую одновременно с СУБД. Он имеет свой входной язык и стандартный для продуктов Sybase сетевой интерфейс Open Server. Такое разделение снижает нагрузку на СУБД и делает систему в целом более открытой.

Транзакция может вносить изменения в одну или несколько таблиц базы данных. Выбранные для репликации таблицы специальным образом помечаются. Для каждой такой таблицы или группы ее строк, выбранной по заданному условию, определяется один узел, в котором данные таблицы являются первичными. Это тот узел, в котором происходит наиболее активное обновление данных. Репликационному серверу, обслуживающему БД с первичными данными, задается описание тиражирования (replication definition). В этом описании, в частности, могут быть заданы интервалы значений первичного ключа таблицы (или другое условие на первичный ключ), при выполнении которого измененные данные будут тиражироваться из этого узла к подписчикам. Если условие не задано, то описание тиражирования действует для всех записей таблицы.

Возможность тиражирования группы записей таблицы означает, в частности, что часть записей таблицы может быть первичными данными в одном узле, а часть - в других. В одном или нескольких узлах (СУБД), которым нужны измененные данные, в обслуживающем его репликационном сервере создается подписка (subscription) на соответствующее описание тиражирования.

В узле, содержащем первичные данные, для каждой тиражируемой базы данных запускается специальная компонента - репликационный агент (Replication Agent - RA). Он подключается к серверу БД и получает от него уведомления о завершении транзакций. Измененные данные передаются репликационному серверу, обслуживающему этот узел. Репликационный сервер в соответствии с описанием тиражирования и подписками отправляет данные в специальном эффективном протоколе по месту назначения - соответствующим репликационным серверам в удаленных узлах.

Именно в этом месте - между репликационными серверами - связь может быть медленной или недостаточно надежной. Передаваемые данные в составе транзакции при недоступности узла-получателя записываются в стабильные очереди на диске и затем передаются по мере возможности. Данные могут передаваться в удаленный узел по маршруту, содержащему несколько репликационных серверов. Данная возможность лежит в основе построения иерархических систем репликации.

СУБД, хранящая вторичные данные, может быть любой СУБД, доступной через шлюз, в том числе Oracle, Informix, Ingres, DB2, RMS, ISAM, или даже приложение Open Server. СУБД, хранящая первичные данные, требует наличия для нее RA. Сейчас RA имеется для Sybase SQL Server, Oracle, DB2, Sybase SQL Anywhere. Готовятся RA и для других СУБД. Интерфейс RA открыт и возможно создание RA для нестандартных источников данных.

Важно, что репликационный сервер тиражирует транзакции, а не отдельные изменения в базе данных. Метод тиражирования транзакций гарантирует целостность внутри транзакции, и, как следствие, невозможность нарушения ссылочной целостности. Схема обновления первичных данных и копий данных исключает возможность возникновения конфликтов (конфликты могут быть вызваны только неправильным проектированием системы или сбоем).

В различных узлах предприятия используются базы данных от разных производителей. Например, в системе принятия решений это может быть Sybase, а в геоинформационной системе - Oracle.

Sybase OmniConnect осуществляет унифицированный доступ приложений к разнородным источникам данных. Специальные шлюзовые компоненты организуют работу в системе с любой промышленной СУБД, включая Oracle, Informix, Ingres, DB/2, RMS, ISAM. Приложения-клиенты при этом работают только с сервером OmniConnect на диалекте SQL фирмы Sybase (Transact-SQL), а необходимая трансляция языка SQL и преобразование типов данных автоматически осуществляется шлюзовыми модулями. Для работы с хранилищами данных на "больших" ЭВМ (mainframe) Sybase поставляет также продукцию фирмы Micro Decisionware (Sybase купила фирму MDI в начале 1994 года). MDI предоставляет шлюзы в DB/2, SQL/DS, SQL/400, в том числе через IBM DRDA-интерфейс.

OmniConnect хранит информацию о размещении таблиц на том или ином сервере БД. Централизованно хранятся и исполняются глобальные хранимые процедуры. Приложение-клиент может осуществлять транзакции, в которых участвуют таблицы из различных БД, а также выполнять процедуры, которые OmniConnect при работе с СУБД, отличными от Sybase, прозрачно преобразует к соответствующему диалекту SQL.

Sybase MPP - это расширение архитектуры Sybase SQL Server, разработанное и оптимизированное для массовой параллельной обработки. Он обладает открытой параллельной архитектурой, предназначенной для поддержки очень больших баз данных (VLDB). Sybase MPP использует стандартный SQL и открытые интерфейсы. С ним работают те же приложения, что и с SQL Server, без необходимости перепрограммирования.

Sybase MPP выполняет параллельно операции считывания (выборки), добавления, обновления и удаления записей. Параллельно выполняются и загрузка/восстановление, и создание индексов. Архитектура Sybase MPP не содержит узких мест, связанных с разделяемой памятью или разделяемым дисковым пространством. При выполнении параллельной выборки Sybase MPP использует индексы.

Дополнительные процессоры и диски могут добавляться в систему постепенно, достигая масштабируемости в сотни раз. Имеется возможность тиражировать и перестартовать ключевые компоненты системы так, чтобы обеспечить быстрое восстановление при сбоях.

С точки зрения приложений, пользователей и разработчиков Sybase MPP выглядит как сервер с одной логической базой данных. Для этой базы данных работает оптимизатор запросов. Поддерживаются хранимые процедуры и глобальный репозитарий, где хранится информация о размещении данных. Для управления системой имеются графические утилиты.

Sybase IQ. В системах поддержки принятия решений используется два типа продуктов. Одни оптимизированы для предварительно известных запросов, а другие - для запросов "на лету" (т.е. заранее неизвестных). Sybase IQ не требует заранее определять "пути", то есть не требует использовать предварительные знания о структуре запросов.

С использованием побитовой схемой индексации Sybase IQ практически все данные в БД могут быть проиндексированы. Поэтому никакой запрос не приведет к просмотру записей таблиц.

Sybase IQ не требует изменений в приложениях - любая программа, работающая с SQL Server, будет работать с IQ. Собственно Sybase IQ не выполняет отдельных обновлений данных. Он в прозрачном для клиента режиме передает их для выполнения SQL Server. Sybase IQ очень эффективно выполняет пакетные дополнения к базе данных. В отличие от технологий, основанных на B-деревьях, при добавлении 10 миллионов строк в таблицу, где уже есть десятки миллионов строк, Sybase IQ просто построит дополнительные страницы индекса и не потребует перестраивать весь индекс целиком.

Sybase Backup Server - это специальный сервер для выгрузки и загрузки баз данных, не требующий остановки SQL Server и не снижающий его производительности.

Дамп базы данных и журнала производится без прекращения использования БД. Поэтому имеется возможность выполнять дамп часто, что повышает надежность системы. Backup Server выполняет весь необходимый ввод-вывод. Команды на дамп или загрузку выдаются непосредственно для SQL Server, который обращается к Backup Server.

Выгрузка производится в два этапа - сначала выгружается состояние данных на момент начала дампа, а затем оно дополняется изменениями, произошедшими за время дампа. Имеется возможность получения как полного дампа базы данных, так и дампа изменений.

Для анализа функционирования сервера Sybase предоставляет компоненту SQL Monitor, представляющую на любом компьютере-клиенте в графическом виде данные по загрузке сервера, вводу/выводу, интенсивности транзакций, использованию памяти сервером. SQL Monitor как клиент взаимодействует с SQL Monitor-сервером, выполняющемся на том же компьютере, что и SQL Server. SQL Monitor-сервер использует разделяемую память для доступа к информации о работе SQL Server, и поэтому не загружает SQL Server.

Для управления конфигурацией сервера имеется как набор хранимых процедур и set-команд, так и графическое средство. При работе с параметрами конфигурации в командном режиме введено три уровня представления - базовый уровень, служащий для управления основными параметрами, промежуточный уровень и детальный уровень, на котором доступны все параметры тонкой настройки (рис.10). Параметры организованы иерархически в соответствии с группами функций SQL Server, которыми они управляют. Имеется возможность создать несколько поименованных конфигураций и переключаться между ними.

Программные интерфейсы ко всем службам, предоставляемым архитектурой Sybase, реализуются через API Open Client и Open Server. Программные интерфейсы Open Client/Server независимы от платформы. Среди поддерживаемых платформ DOS, Windows, MVS/CICS, Macintosh, NetWare, Windows NT, OS/2, UNIX (все главные варианты), VMS и OpenVMS.

При разработке приложений-клиентов на языке 3-го поколения используются библиотеки с Си-интерфейсом: DB-Library, CT-Library или ODBC (только под Windows). При разработке приложений серверного типа используется библиотека Open Server. Этот набор блоков для построения сервера может использоваться, например, для доступа к нестандартному оборудованию или построения шлюзов.

ODBC API представляет собой набор вызовов функций. Доступ к базе данных в нем задается операторами SQL, которые передаются соответствующим функциям в виде строковых параметров. Спецификация ODBC, как и Embedded SQL, поддерживает курсоры. Имеется возможность вызывать хранимые процедуры. Из программы могут выдаваться любые операторы SQL, в том числе DDL-операторы. ODBC-драйверы для Sybase выпускают несколько фирм. Такой драйвер входит в состав Sybase Open Client.

Большинство приложений, связанных с обработкой данных в среде MS-Windows, поддерживают ODBC-интерфейс или DB-Library, и, соответственно, имеют доступ к СУБД Sybase. Среди таких приложений Microsoft Excel, Word, Access, Visual Basic.

Технология и библиотеки OpenServer, входящие в состав Sybase System 11, позволяют разрабатывать собственные приложения, использующие данные от технологического оборудования. Другое применение Open Server - разработка серверных компонент для математической или криптографической обработки данных. Обработчик Open Server может быть использован как приложением-клиентом, так и вызван из хранимой процедуры SQL Server.

***SQL Anywhere 5.0*** - составная часть Sybase System 11 (Sybase SQL Anywhere - это новое название для пятой версии Watcom SQL). С объединением компаний Sybase и Powersoft в феврале 1995 года фирма Watcom вошла в состав Sybase, Inc. SQL Anywhere работает на платформах Windows 3.x, Windows NT и Windows 95, OS/2, DOS, Novell NetWare и QNX. Среди поддерживаемых сетевых протоколов - TCP/IP, Named Pipes, IPX/SPX.

Приложения-клиенты могут разрабатываться с использованием ODBC, Embedded SQL и собственного интерфейса Watcom HLI. Имеется собственный DDE-сервер для интеграции, например, с Excel, Word или Visual Basic.

Включение SQL Anywhere в состав линии продуктов Sybase придало новые качества продукту: собственная репликация данных (SQL Remote); поддержка системы репликации (Sybase Replication Server); графический инструмент администрирования (SQL Central); поддержка (Transact-SQL); поддержка ODBC2.1; повышение производительности и мониторинг производительности; расширения языка Watcom SQL; универсальный серверный интерфейс SQL Anywhere Open Server.

База данных SQL Anywhere может участвовать в схеме репликации Sybase Replication Server. Для интеграции с Replication Server используется специальный шлюзовой компонент - Open Server Gateway для SQL Anywhere, который "транслирует" стандартный для продуктов Sybase интерфейс Open Client/Server в интерфейс SQL Anywhere. Для отслеживания изменений в базе данных SQL Anywhere предусмотрен компонент Replication Agent.

Другой механизм репликации (SQL Remote) - это система репликации только между базами данных SQL Anywhere. Это менее гибкая система, чем Replication Server; например, в ней жестко требуется, чтобы имена объектов тиражирования были одинаковыми во всех базах данных. Зато SQL Remote легко администрируется, пригоден для широкого использования в том числе и в случаях, когда базы данных не имеют прямого соединения друг с другом.

SQL Remote - это система репликации (тиражирования данных) между базами данных SQL Anywhere, основанная на передаче сообщений. Эта система администрируется из единого центра и наиболее подходит для обмена данными с laptop-компьютерами и другими пользователями, связь с которыми есть от случая к случаю. Система SQL Remote может использовать для репликации средства электронной почты (MAPI, VIM, SMTP), файловый обмен и готовящийся к выходу механизм Sybase Messaging Services.

SQL Remote реплицирует данные между "консолидированной" БД и одной или несколькими "удаленными" БД. В качестве удаленных могут выступать как сетевые серверы SQL Anywhere, так и локальные однопользовательские БД. SQL Anywhere поддерживает иерархическую модель репликации.

SQL Central - графическое средство администрирования БД, работающее под Windows 95 и Windows NT. Из SQL Central осуществляются все административные действия от создания баз данных, загрузки/выгрузки до создания таблиц, процедур и назначения полномочий. SQL Central также управляет системой репликации SQL Remote, описанной выше. Для удобства предусмотрен drag-and-drop интерфейс, контекстная подсказка и контекстное меню по правой кнопке мыши.

SQL Anywhere 5.0 включает набор расширений языка Watcom SQL в сторону диалекта Transact-SQL Sybase, используемого в Sybase SQL Server на различных платформах. Диалект Transact-SQL сам по себе не является лучше или хуже диалекта Watcom SQL. Достигнутая совместимость означает, что на нижних уровнях распределенной информационной системы можно использовать "легкую" по ресурсам, мощную по функциональности и дешевую СУБД SQL Anywhere, а на уровне выше - промышленную высокопроизводительную СУБД Sybase. При этом приложения, работающие с этими СУБД, окажутся инвариантны к тому, с какой СУБД они работают.

Использование журнала транзакций в SQL Anywhere уменьшает время фиксации каждой транзакции. Сервер SQL Anywhere является 32-разрядным (кроме платформы Windows 3.X).

СУБД использует оптимизатор, который выбирает лучшую стратегию для выполнения каждого запроса. Для этого оптимизатор определяет стоимость каждой стратегии, оценивая требуемое количество считываний и записей на диск. Выбранная стратегия задает, в каком порядке происходит доступ к таблицам в запросе и следует ли использовать индекс для каждой таблицы. Окно статистики в интерактивном средстве для формирования запросов (ISQL) позволяет просмотреть выбранный план выполнения любого оператора SQL.

В SQL Anywhere предусмотрен набор статистической информации, предназначенной для оценки производительности. Эта информация доступна из инструмента администратора SQL Central и приложений-клиентов. В дополнение, подобная информация предоставляется из SQL Anywhere в монитор производительности Windows NT.

WSQL Dynamic Data Exchange (DDE) - это метод взаимодействия приложений. DDE использует архитектуру клиент-сервер. Так, приложение-клиент может запрашивать данные у приложения-сервера.

WSQL DDE-сервер - это Windows-приложение, которое позволяет выбирать и изменять данные в БД SQL Anywhere. Получаемые данные могут передаваться приложению-клиенту через буфер в памяти или через Clipboard.

Sybase System 11 - семейство интегрированных продуктов, нацеленных на каждую область применения реляционной СУБД, от обработки данных в реальном времени до задач принятия решений и хранилищ данных, от тысяч пользователей и очень больших баз данных до "легкого" сервера для рабочих групп, который работает на персональном компьютере.

## 1.1.2 IBM DB2

Фирма IBM является крупнейшей фирмой в мире, занимающейся компьютерной техникой и информационными технологиями. Среди многочисленных аппаратных и программных решений, предлагаемых IBM, весьма почетное место всегда занимали компоненты для сохранения и использования данных. Семейство DB2 систем управления реляционными базами данных является ключевым программным компонентом, предлагаемым фирмой IBM для хранения и интенсивного использования больших объемов данных.

Прикладные системы, для которых используется DB2, - это стандартные OLTP (on-line transaction processing) системы. Кроме того, DB2 широко применяется для информационных систем различного назначения, для построения информационных хранилищ данных и для систем поддержки принятия решений.

Еще некоторое время назад IBM рассматривала DB2, в первую очередь, как необходимое средство поддержки для аппаратных платформ, производимых и продаваемых IBM (DB2 для MVS, VM, VSE, AIX, OS/400, OS/2), и не выпускала DB2 для серверных операционных систем других фирм. Бурное развитие программного рынка в последние годы дало жизнь версиям DB2 для разнообразных вариантов ОС Unix и Windows NT.

Появление работы E. Кодда в 1970 г. с предложением реляционного подхода для создания баз данных положило начало исследовательской активности в области реляционных баз данных в ряде лабораторий IBM. Первые проекты и прототипы были консолидированы в лаборатории Санта-Тереза, Калифорния в середине семидесятых годов и завершились созданием System R. Эта система во многом определила архитектуру современных реляционных баз данных, в частности использование непроцедурного языка запросов SQL. После исследовательской стадии, в 1981 г. появился коммерческий продукт SQL/DS для VM, а в 1983 г. - собственно DB2 для мэйнфреймов под MVS и MVS/ESA. Другим интересным проектом, влияние которого ощутимо в DB2, был Starburst на рубеже 80-90-x.

Многочисленные современные проекты, ведущиеся в настоящее время вокруг семейства DB2 в Альмадене, Санта-Тереза и Торонто позволяют судить о том, в каких направлениях видят развитие баз данных разработчики DB2.

Значительная часть ведущихся проектов касается расширения сферы применения реляционных баз данных для новых задач, связанных с хранением и использованием новых сложных типов данных, поддержки объектно-ориентированных расширений и реализаций новых методов поиска и анализа данных. Другая часть исследований касается вопросов повышения производительности и масштабируемости, использования методов распараллеливания в работе серверов баз данных. Часть результатов этих проектов уже воплотилась в текущей версии DB2, другие проекты должны получить развитие в следующей DB2 Universal Server (DB2 версия 5, согласно новой унифицированной нумерации).

Программные решения IBM, наряду с DB2, включают и другие программные средства, например: транзакционные продукты CICS, Encina, MQSeries, которые вместе с DB2 могут образовывать более гибкую и подходящую для конкретного случая архитектуру, чем основанное только на использовании СУБД решение.

***Продукты семейства DB2.*** DB2 Parallel Edition ориентирована на технологию распараллеливания запросов к очень большим базам данных (VLDB), размещенным на системах с массовым параллелизмом (MPP). Серверы из семейства DB2 Common Server представляют общее решение для рабочих групп, малых и средних организаций, переносимое и масштабируемое на серверах, работающих под управлением OS/2, Windows NT, AIX, HP-UX, Sun Solaris, SCO, SINIX. Существуют также более специализированные серверные продукты типа DataJoiner, который предоставляет клиентам-приложениям оптимизированный доступ к распределенным данным, управляемым Oracle, Sybase, MS SQL Server, DB2, и предлагает единую базу-образ гетерогенной среды.

Поддержка существующих международных отраслевых стандартов, таких как SQL 92 Entry Level, ODBC, XA, JDBC, новых версий SQL и корпоративных стандартов фирмы IBM, является необходимым качеством семейства DB2, обеспечивающим взаимодействие разнообразных программных и аппаратных средств.

Вокруг базовой группы продуктов существует разнообразная инфраструктура комплектующих и дополнительных продуктов, предназначенных для администрирования баз данных – DataHub и компоненты Tivoli, для репликации данных - DataPropagator, семейство средств разработки VisualAge для различных языков программирования (С++, Smalltalk, 4GL, Java, Basic) с CASE-средствами IBM DataAtlas и VisualAge PackBase, пакеты средств формирования запросов и создания отчетов Intelligent Decision Server и QMF, комплексное решение для построения локальных информационных хранилищ данных - Visual Warehouse, средства анализа и поиска информации в базах данных Intelligent Miner, средство для работы с метаданными и построения каталога информационных ресурсов DataGuide, решения для поддержки OLAP-приложений - DB2 OLAP Server and IDS, и так далее.

***DB2 как сервер поддержки OLTP.*** На сегодняшний момент большинство реляционных баз данных занято хранением операционных данных, полученных от OLTP (OnLine Transaction Processing) приложений. Такие приложения предназначены для поддержки текущей деловой активности финансовых, административных и промышленных организаций, где данные, в основном, числовые и символьные.

OLTP-приложения оперируют с критически-важными данными, которые требуют защиты от несанкционированного доступа, от нарушений целостности, от аппаратных и программных сбоев. Типичной является ситуация, когда система должна поддерживать быстрый доступ к данным большого числа пользователей, выполняющих относительно несложные запросы на поиск и обновление данных. Критическим показателем для OLTP является время ожидания выполнения типичных запросов, которое не должно превышать нескольких секунд.

В условиях постоянного роста объема хранимых данных, необходимым требованием к DB2 становится возможность масштабировать систему при увеличении объемов данных, при увеличении потока транзакций и количества пользователей, в первую очередь, путем наращивания аппаратных ресурсов, при сохранении времени отклика системы в допустимых пределах.

До появления реляционных СУБД задачи поддержки OLTP решались с использованием дополнительного программного обеспечения - мониторов транзакций, например CICS. Однако практика внедрения архитектуры клиент-сервер без применения промежуточного монитора транзакций потребовала универсализации сервера баз данных и оснащения его некоторыми свойствами мониторов транзакций для внешнего взаимодействия и повышения производительности.

Наглядными примерами тенденции преобразования многих СУБД и, в частности, DB2 в легкие мониторы транзакций могут служить появившиеся у СУБД функции поддержки хранимых процедур и компонентов для координации изменений в распределенной базе данных.

***DB2 для OS/390 как корпоративный сервер.*** DB2 для мэйнфреймов появилась в эпоху вычислительных центров и с самого своего появления была ориентирована на поддержку очень крупных централизованных систем, для которых главными требованиями являлись производительность и очень высокая надежность.

DB2 предназначалась для взаимодействия с приложениями, работающими под MVS-TSO, транзакционными продуктами CICS, IMS, и поддержки пакетной обработки больших объемов данных. Последующие версии DB2 стали поддерживать архитектуру клиент-сервер.

Более высокая надежность платформы OS/390 по сравнению с вычислительными системами других типов вместе с развитием и удешевлением технологий хранения данных и высокопроизводительных коммуникаций определяет использование DB2 для OS/390 как корпоративного сервера баз данных и центра массивной обработки данных.

Для DB2/390 основными тенденциями развития являются, с одной стороны, стремление к наращиванию производительности, что связано с тенденцией физического роста корпоративных баз данных, а с другой стороны, укрепление интеграции с прочими программными средствами на различных платформах.

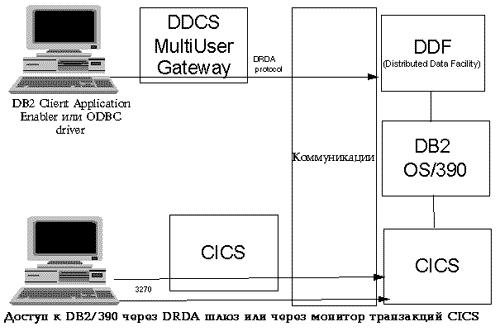
DB2 для MVS версии 3, появившаяся в 1992 году, обладала серьезными улучшениями в области производительности, на основе распараллеливания операций ввода/вывода и работы с независимыми разделами дисковых устройств, использования возможностей аппаратного обеспечения и операционной системы, например методов компрессии данных и аппаратной поддержки сортировки.

В DB2 для MVS версии 4 важные добавления были сделаны в поддержку архитектуры клиент-сервер, использования DB2 в масштабируемой архитектуре Parallel Sysplex, возможности распараллеленного исполнения запросов. Появились также хранимые процедуры как важное средство снижения сетевого трафика и перенесения бизнес-логики приложений на сервер баз данных. Также увеличились возможности DB2 по поддержке клиентов до 25 тысяч на один сервер, а с учетом возможности параллельной работы в группе до 32 узлов DB2 в Parallel Sysplex - до 800 тысяч клиентов.

Для архитектуры клиент-сервер наиболее важные улучшения связаны с поддержкой клиентов в сетях TCP/IP, хранения данных в ASCII-форматах, более развитых механизмов хранимых процедур, стандартизацией DB2 SQL и появлением продукта DB2 WWW Connection для доступа к данным из Internet.

***Архитектура DRDA.*** При изначальной ориентации DB2 на платформах MVS, VM, VSE на централизованные приложения впоследствии, в условиях распространения архитектуры клиент-сервер, потребовалась дополнительная поддержка удаленного доступа для PC- и Unix-рабочих станций к базам данных на мэйнфреймах и на системах типа AS/400.

Решая проблему поддержки архитектуры клиент-сервер, IBM предложила архитектуру DRDA (Distributed Relational Database Architecture) в виде открытой спецификации, описывающей протоколы взаимодействия удаленного клиента с базами данных. Реализация DRDA должна была позволить продуктам от разных производителей баз данных прозрачно взаимодействовать и, в частности, образовывать единую распределенную базу данных.



Подход DRDA позволяет решить многочисленные проблемы кросс-платформенного взаимодействия, в частности конвертацию ASCII- и EBCDIC-данных, различия в диалектах SQL, командах, типах данных, строении каталогов. В настоящее время архитектуру DRDA поддерживают прежде всего все серверы семейства DB2; остальные производители реализовали продукты типа DRDA-реквесторов, которые позволяют прикладным программам-клиентам иметь доступ к базам DB2, например к DB2/390.

Типичный пример такого DRDA-реквестора производства IBM - продукт DDСS, работающий на OS/2, AIX, Windows, Windows NT, служащий многопользовательским шлюзом к DB2 для пользователей локальной сети или, в случае однопользовательского варианта, работающий на компьютере клиента. Реализация поддержки DRDA по протоколам TCP/IP, в добавление к традиционной поддержке протоколов SNA, в новых версиях серверов DB2 и шлюзов DDCS значительно упрощает многочисленным пользователям доступ к базам данных DB2 на мэйнфреймах и AS/400.

***DB2 Common Server.*** Эксплуатация реляционных баз данных поставила перед разработчиками практические задачи по дополнению СУБД новыми возможностями и привела к появлению нового поколения систем управления базами данных, так называемых расширенных реляционных (extended relational) СУБД. К этому классу относят системы управления базами данных, поддерживающие ряд дополнительных возможностей, которые выходят за рамки реляционной алгебры, - триггеры, хранимые процедуры, контроль целостности и т. д.

На сегодняшний день практически все системы управления реляционными базами данных ведущих производителей, в том числе DB2, можно отнести к категории расширенных.

Целью проекта IBM Starburst в конце восьмидесятых годов было создание такой расширяемой системы управления реляционными базами данных. Практическим результатом проекта Starburst и его продолжения проекта Starwings было появление семейства DB2 Common Server в 1993 г.

Для поддержки OLTP-приложений в DB2 Common Server реализовано большое число механизмов, улучшающих производительность, включая разнообразные алгоритмы буферизации, алгоритмы контроля ресурсов и методы мониторинга, конфигурации и настройки параметров системы, использующие статистику системы.

Система управления буферизацией использует алгоритмы распараллеливания операций ввода/вывода, предварительного чтения данных и индексов, асинхронной записи на диск и многие другие. DB2 Performance Monitor, поставляемый вместе с DB2, предоставляет широкие возможности для сбора и анализа данных о производительности системы, включая информацию о событиях и периодические срезы параметров производительности.

Оптимизатор DB2 является одним из наиболее важных компонентов, обеспечивающих DB2 высокую производительность и адаптацию к различным задачам. DB2 строит так называемую QGM (Query Graph Model) для внутреннего представления запросов и использует ее на этапах проверки семантики запросов, преобразования запросов к оптимальному виду и построения плана исполнения запроса. Расширяемость QGM позволяет легко добавлять к SQL DB2 новые конструкции.

При анализе плана исполнения запроса оптимизатор, используя статистику каталогов и параметры аппаратной части, оценивает эффективность того или иного плана исполнения запроса и выбирает наилучший. Один из административных компонентов DB2, Visual Explain, позволяет наглядно представить выбранный план исполнения запроса и даже оценить его эффективность при изменении параметров системы.

***Объектно-реляционные свойства DB2.*** В настоящее время существует множество приложений, оперирующих с данными, которые имеют гораздо более сложную и чаще изменяемую структуру, чем традиционно используемая в реляционных базах данных. Стремительно растет число мультимедийных приложений. Кроме того, актуальна более гибкая поддержка серверами баз данных бизнес-логики приложений.

DB2 Common Server, появившаяся в 1995 году, уже содержит инфраструктуру для реализации объектно-ориентированных функций, на основании которой построены реляционные расширения DB2 (relational extenders). Расширения позволяют определять структуру, атрибуты и поведение новых типов данных, сохранять эти данные в таблицах DB2 и затем использовать их в SQL-выражениях. В общем случае при создании новых типов данных используется UDT (User Defined Type - определяемые пользователем типы данных) DB2, часто основанные на применении больших объектов DB2, поведение новых типов данных определяется с помощью нескольких UDF (User Defined Function - определяемая пользователем функция). При этом механизмы триггеров (triggers) и ограничений (constrains), предлагаемые DB2, оснащающие базу данных возможностями хранить правила поведения данных, могут использоваться для управления внутренней структурой новых сложных типов данных.

Подобно некоторым другим базам данных, DB2 Common Server позволяет хранить данные в больших бинарных (BLOB) и символьных (CLOB) объектах. Размер объекта может достигать 2 Гбайт.

Поскольку размер таких объектов сильно отличается от традиционных данных, на обработку которых настроены серверы реляционных баз данных, то DB2 содержат ряд функций помогающих обеспечить нормальную производительность: переменные типа локаторов, ссылки, специальные режимы при журналировании. Кроме того, IBM предлагает специализированные программные и аппаратные решения, такие как Digital Library, ориентированные на хранение и высокопроизводительную обработку мультимедийных данных и на взаимодействие с DB2.

Постоянно растущие объемы текущих операционных данных представляют собой значительную ценность для решения разнообразных задач управления, поскольку являются объективным отражением происходящих деловых процессов.

На сегодняшний день задача построения информационных хранилищ представляет весьма сложный комплекс проблем и решений, касающихся пополнения хранилищ информацией, трансформации, хранения, представления и использования информации. Причем важнейшую роль в этом комплексе играют весьма сложные инструментальные средства. Качественное изменение характера данных в информационных хранилищах и изменение характера работ, производимых над базой данных, требуют определенных технологических изменений в ядре самой СУБД, в частности поддержания новых методов хранения и размещения данных и новых методов поиска.

DB2 кроме естественной роли быть источником операционных данных для пополнения хранилищ обеспечивает хранение самих информационных данных и эффективное выполнение сложных запросов, включающих многочисленные соединения таблиц, вычисления и методы группировки данных. В частности, уже сейчас оптимизатор DB2 Common Server поддерживает оптимизацию запросов к базам данных, смоделированным по принципу звезды (Star Schema), широко используемым для OLAP (Online Analytical Processing) приложений и состоящим из большой таблицы фактов и нескольких таблиц размерностей.

Для поддержки очень больших баз данных объемом в сотни гигабайт и даже терабайт семейство DB2 предлагает два решения, основанные на технологиях распараллеливания - DB2/390 в Parallel Sysplex (архитектура Data Sharing) и DB2 Parallel Edition.

Архитектура DataSharing позволяет масштабировать решения путем подключения дополнительных серверов и при увеличении объемов данных, и при увеличении количества и сложности запросов. При выполнении сложных запросов поддерживается техника разделения запроса на отдельные задачи и выполнение этих задач параллельно несколькими серверами DB2, входящими в Sysplex.

DB2 Parallel Edition создана на основе DB2 для RS/6000 и предназначена для поддержки приложений, требующих выполнения сложных запросов к большим массивам данных. DB2 Parallel Edition использует технологию Sharing Nothing, позволяющею почти линейно масштабировать систему до сотен и даже тысяч параллельно работающих узлов.

DB2 Parallel Edition разработана для работы на различной аппаратной архитектуре, на системах POWERparallel SP2, на комплексах HACMP/6000 и группе рабочих станций RISC/6000, связанных локальной сетью.

Данные любой базы данных распределяются между несколькими узлами DB2 Parallel Edition с использованием схемы хеширования. При этом алгоритмы распределения данных обеспечивают сбалансированность работы между узлами, позволяющую избежать перегрузки одних узлов и простоя других, и минимизирование передачи данных между узлам во время исполнения запросов, например.

IBM предлагает набор продуктов для репликации данных между серверами семейства DB2, а также между DB2 и базами данных других производителей. Решение от IBM DataReplication состоит из двух типов компонентов Capture и Apply для всех платформ, где функционирует DB2. Компоненты Capture предназначены для выборки из базы данных источника измененных данных и организации таблиц для промежуточного хранения и обработки реплицируемых данных. Компоненты Apply ответственны за передачу реплицируемых данных между серверами баз данных и добавление их в целевые таблицы.

Сложность построения хранилища данных, охватывающего все источники данных большой корпорации или предприятия, заставляет иногда предпочесть локальные и более дешевые варианты внедрения небольших информационных хранилищ для отдельного подразделения или конкретной предметной области. Продукт IBM Visual Warehouse использует в качестве основы административной базы данных для хранилища DB2 для OS/2 или Windows NT и серверы из семейства DB2 для самого хранилища.

Компоненты собственно Visual Warehouse обеспечивают процесс преобразования данных из баз данных DB2, Oracle, Informix, Sybase, ODBC - источников в информационные данные, и организуют семантически значимые представления (business view) для разнообразных аналитических, статистических и отчетных приложений клиентов. Другой важнейшей функцией, которую выполняют административные компоненты Visual Warehouse, является автоматизация непрерывных процессов создания и управления хранилища.

Продукт ***IBM Intelligent Miner*** представляет собой интегрированное средство для сложного анализа данных, хранящихся в реляционных базах данных и файлах. Он позволяет добывать из баз данных ранее неизвестную и содержательную информацию, предоставлять ее для анализа и принятия решений.

Набор API для приложений-клиентов позволяет разработчикам создавать свои собственные приложения, использующие алгоритмы Intel-ligent Miner. Для конечных пользователей Intelligent Miner имеет функцию подготовки данных к поиску и представления найденной информации в графическом виде. Серверные компоненты Intelligent Miner функционируют в настоящее время под AIX, OS/390, OS/400.

По сравнению с многочисленными средствами создания отчетов и запросов для персональных компьютеров и рабочих станций, аналогичных средств для хост-систем не так много. В своем составе QMF (Query Management Facility для DB2/390) имеет средство формирования запросов, редактор таблиц, средство составления отчетов и обеспечивает интерфейсы для поддержки приложений. QMF поддерживает несколько методов формирования интерактивных запросов. Результаты запроса могут быть выведены на экран в самых разных форматах, включая табличный, матричный, свободный и графический. QMF является достаточно мощным продуктом, даже с точки зрения специалистов в области обработки данных. Последние версии QMF поддерживают работу в среде рабочих станций, а также содержат ряд усовершенствований для среды мэйнфреймов.

Клиентский компонент QMF для работы в среде Windows, который известен под названием Shuttle, дает пользователям возможность выполнять запросы QMF к центральному хост-компьютеру и выводить результат на экран рабочей станции для встраивания в другие программные продукты для рабочих станций, например в электронные таблицы Lotus 1-2-3 или Microsoft Excel.

Стремительное развитие Internet и рост популярности WWW, наблюдаемые в настоящее время, открывают новые возможности использования баз данных.

С одной стороны, многое обещает организация доступа огромного числа пользователей Internet к коммерческим OLTP-системам. Распространение intranet как технологии для корпораций делает эту задачу еще более актуальной.

С другой стороны, перспективным является построение новых Web-серверов с использованием мультимедиа. Применение баз данных позволяет создавать информационные узлы, сочетающие возможности эффективного поиска, обеспечиваемого реляционными базами данных с наглядным представлением информации и удобным к ней доступом, предоставляемыми Internet. При этом требуется не только статическое хранение Web-страниц, но и динамическая их генерация с использованием реляционных данных.

Использование в Internet потребовало создания определенных дополнений для DB2, таких как поддержка JDBС, приложений, хранимых процедур и UDF, написанных на Java, и дополнительных программных средств для взаимодействия с серверами Internet, такими как DB2 WWW Connection и являющимся его развитием Net Data.

## 1.1.3 RDMS Oracle

Компания Oracle проникла на российский рынок более десяти лет назад, и продукция этой фирмы хорошо известна. В 1979 г. небольшая компания Silicon Valley разработала Oracle - первую коммерческую реляционную базу данных с языком доступа к данным SQL. Первой СУБД клиент/сервер стал выпущенный в 1985 г. Oracle5. До недавнего времени, Oracle7 была последней версией сервера базы данных Oracle, появившейся в 1992 г. В прошлом году фирма выпустила новую версию Oracle8. К сожалению, пока еще очень мало литературы по новой версии, так что придется рассматривать технологию уже не самую "горячую". С другой стороны практически все направления развития серверной технологии, получившие отражение в Oracle8, в той или иной степени уже заложены в Oracle7.3.

Oracle7 это реляционная СУБД и семейство продуктов, обеспечивающих создание автоматизированных и информационных систем различного назначения. В состав семейства входят: СУБД Oracle7 RDBMS, средства проектирования приложений CDE CASE (Designer/2000), средства разработки приложений CDE Tools (Developer/2000), средства конечного пользователя, средства интерфейса с программными продуктами третьих фирм, коммуникационные средства и т.д.

***Общие функциональные возможности.*** Версия 7.3 сервера Oracle содержит ряд функциональных новшеств, направленных как на расширение возможностей разработчиков приложений, так и на развитие возможностей самой системы по обслуживанию большого числа одновременных пользователей. Обусловлено это целым рядом архитектурных решений, и не в последнюю очередь хорошо выверенным механизмом блокировок. Oracle устроен так, что разработчик приложений может не заботиться об эффектах многопользовательского режима работы. Сервер сам обеспечивает все необходимые блокировки (хотя позволяет выпонять их и "вручную"), причем осуществляет их всегда на минимально возможном уровне: скажем при изменении записи только эта запись и будет заблокирована от изменений другими пользователями (до завершения транзакции). В Oracle необходимость обеспечения блокировок учитывается уже в организации хранения данных, а сам этот механизм является неотъемлемой частью ядра сервера, "переплетаясь" со всеми его внутренними алгоритмами.

И все-таки, проблема блокировки (моды изоляции чтения) продолжает существовать (пока один пользователь читает данные, другой пользователь может эти данные изменять). Стандарт ANSI SQL-92 описывает требования к реализации нескольких мод изоляции операций чтения от выполняющихся одновременно с ним транзакций. Они варьируются от самой "слабой" моды - "незафиксированного" (часто называемого "грязного") чтения, при котором допускается считывание данных незафиксированных транзакций, до самой "сильной" - "повторяемого" чтения, при котором гарантируется повторяемость результата при повторении операции в рамках транзакции. Беда в том, что само наличие всех этих различных мод изоляции в стандарте SQL отражает отнюдь не потребности пользователей, а различные степени компромисса с возможностями разработчиков СУБД. Пользователей же волнует совсем другое: как избежать тех неприятных эффектов, которые могут быть связаны с использованием всех стандарных мод изоляции, кроме самой "сильной" из них.

Сущность моды изоляции "согласованное чтение", реализуемой сервером Oracle состоит в том, что любая операция чтения в Oracle выдает пользователю данные только тех транзакций, которые были завершены к моменту начала операции. Oracle реализует "согласованное чтение" без использования блокировок вообще. Операция чтения в Oracle никогда не блокируется и никогда не блокирует других. Данный режим работы является среди коммерческих СУБД уникальным. Мода "согласованного чтения" не совпадает ни с одной из мод изоляции, принятых в стандарте SQL-92. Она "сильнее" (и следовательно покрывает) все моды, кроме "повторяемого чтения", но она "слабее" последней. Действительно, при повторе операции в моде "согласованного чтения" можно получить совсем другой результат, ибо изменится момент времени, по которому синхронизуется "срез" данных. Oracle, правда, предоставляет возможность объединять несколько операций чтения в read-only транзакцию, синхронизуя их при этом к одному моменту времени. В версии 7.3 Oracle позволяет в явном виде установить моду изоляции "repeatable read", причем опять без использования блокировок.

***Функциональные новшества.*** В Oracle 7.3 появилась возможность читать и писать поля таблиц типа Long по частям (на уровне Oracle Call Interface), что безусловно полезно, ибо размер таких полей может доходить до 2 Гбайт. Расширился набор типов представлений (views), для которых допускается их непосредственная модификация. Появился ряд новшеств в языке PL/SQL (процедурном расширении SQL), самое заметное из которых - поддержка таблиц, хранимых в памяти сервера. Новые алгоритмы обработки запросов. Выполнение SQL-запроса - особенно имеющего сложную структуру - обычно распадается на несколько взаимосвязанных операций. Само это разбиение, а тем более выбор методов выполнения операций, как правило, допускают множество альтернативных решений. Выбор оптимальной их комбинации - задача оптимизатора, который на основании как характера запроса, так и имеющейся информации о задействованных таблицах и индексах, наличии тех или иных системных ресурсов (в Oracle 7.3 расширен набор видов предоставляемой оптимизатору информации: теперь он может учитывать частотные гистограммы индексируемых полей) строит оценку стоимости разных вариантов решения.

Помимо "джентльменского" набора более или менее универсальных методов существует также целый ряд более узкоспециализированных, т. е. таких, которые очень хорошо работают в некоторых ситуациях, но могут быть совсем неэффективными (или даже неприменимыми) в других. Несмотря на такой недостаток, применение этих методов может дать очень заметный эффект, особенно при выполнении сложных запросов над большими объемами данных, что характерно для систем поддержки принятия решений (DSS) (хранилищ данных). В Oracle 7.3 введен целый ряд таких специализированных алгоритмов.

• "Звездообразные запросы" (star queries). В DSS-системах довольно часто применяются запросы, выполняющие соединение одной большой таблицы (таблицы фактов) с несколькими маленькими (справочными таблицами). При выполнении подобных запросов оказывается эффективным достаточно экзотический алгоритм, выполняющий сначала вычисление декартова произведения справочных таблиц, а затем слияние сортировкой полученного результата с таблицей фактов.

• "Слияние хэшированием". Альтернативный слиянию сортировкой метод выполнения соединений таблиц (еще одна альтернатива - вложенные циклы).

• Применение битовых строк для индексирования. До версии 7.3 Oracle применял для индексирования только B-деревья и хэш-функции (если таблица помещалась в хэш-кластер). В версии 7.3 появилась возможность использования индексов с битовыми строками (bit-map indexes). Их идея очень проста. Если некоторое поле таблицы может принимать ограниченное число значений, то каждому такому значению можно сопоставить битовую строку (количество бит равно количеству записей в таблице), в которой единицы находятся в позициях, соответствующих тем записям, которые имеют данное значение в индексируемом поле. Ясно, что такой индекс позволяет очень быстро находить нужные записи по значениям проиндексированного поля (и любым их логическим комбинациям), а также выполнять операции агрегирования опять-таки по этому полю. Метод эффективен лишь для полей с небольшим количеством допустимых значений, неэффективны операции сравнения с предшествованием (больше/меньше), неэффективны операции вставки, удаления и модификации записей. В действительности "в чистом виде" битовые строки не применяются: они размещаются в листьях B-дерева, что позволяет смягчить указанные недостатки, но очевидно, что в целом они носят принципиальный характер, а потому не устранимы полностью. Oracle позволяет применять bit-map индексирование в сочетании с другими методами индексирования на одной и той же таблице.

До версии Oracle 7.3 основным средством администратора являлся Server Manager - программный продукт с графическим интерфейсом, но ориентированный на управление одной БД (в случае нескольких БД приходилось использовать несколько сессий), не имевший удобных средств графического мониторинга системы и не позволявший непосредственно управлять удаленными заданиями, требовавшими привлечения системных команд и ресурсов, не находящихся под контролем СУБД Oracle. Пробел заполнялся достаточно многочисленными программными продуктами третьих фирм, специализирующихся именно на средствах администрирования. Однако обеспечение единообразного администрирования распределенных систем стало настолько актуальной задачей, что стимулировала развитие новой стратегии корпорации в области средств администрирования сервера БД.

В комплекте с сервером версии 7.3 (в вариантах Workgroup и Enterprise) поставляется Oracle Enterprise Manager. В состав этого программного продукта входит набор утилит управления, интегрированных в единую консоль администратора. Через специальный связной процесс - Communication Deamon - эта консоль может взаимодействовать с интеллектуальными агентами - специальными процессами, функционирующими на компьютерах-серверах, обеспечивающими возможность удаленного управления (впрочем, агент требуется только для выполнения удаленных заданий и контроля за событиями - все основные административные функции реализуются через непосредственную связь консоли с сервером БД). Все управляемые компоненты - БД, серверы (узлы), процессы - отображаются на консоли в навигаторе объектов, позволяющем быстро находить требуемый объект и детализировать представление его структуры до нужного уровня. Непосредственно административные функции выполняются с помощью явного или неявного вызова соответствующих утилит. Для выполнения некоторых действий (перенос пользователя из одной БД в другую, присвоение новой роли пользователю и др.) достаточно "буксировки" мышкой.

Принципиально новой особенностью Enterprise Manager по сравнению с более ранними аналогичными продуктами Oracle является возможность определения и управления выполнением удаленных заданий, реализация которых выходит за рамки возможностей самой СУБД (сбросы, команды ОС и т. п.), а также возможность заставить систему саму извещать администратора о возникших (или даже предполагаемых) проблемах с помощью механизма событий.

Задания могут выполняться по заданному расписанию, причем непосредственный контроль за этим осуществляется локально интеллектуальным агентом, так что в принципе постоянная поддержка связи консоли с сервером не требуется (хотя для того, чтобы изменить задание или время его выполнения, необходимо, чтобы "агент вышел на связь"). Помимо использования набора стандартных типов заданий и их комбинаций администратор может определять принципиально новые, исользуя системно-независимый язык TCL (Task Control Language). Фактически и "стандартные" типы заданий строятся с применением "шаблонов" на этом языке, тексты которых можно использовать в качестве образцов. Интерпретация TCL в конкретной ОС того или иного сервера осуществляется соответствующим интеллектуальным агентом, что делает управление СУБД почти не зависящим от платформы сервера (а таких платформ Oracle поддерживает более 80).

Набор возможных регистрируемых событий варьирует от самых простых типа запуска и останова сервера БД до достаточно "тонких" типа превышения частоты обращений к диску заданного администратором порога. События регистрируются интеллектуальными агентами и передаются на консоль администратора (точнее, на те из консолей, которые "интересуются" данным событием), а если потребуется, сообщение о событии может быть послано администратору по электронной почте или на пейджер.

Еще одной важной особенностью Oracle Enterprise Manager является то, что он имеет открытые интерфейсы на всех своих уровнях, что открывает возможность наращивания его функциональности за счет добавления новых административных утилит, управляющих процессов и пр. Эта возможность прежде всего ориентирована на фирмы, являющиеся поставщиками средств администрирования, но ею могут воспользоваться и сами пользователи СУБД.

Отдельного упоминания заслуживают поставляемые Oracle утилиты, входящие в Performance Package. В него входят: утилита мониторинга системы (несколько десятков стандартных динамических диаграмм плюс возможность определять свои собственные); утилита, показывающая в наглядной форме физическое расположение объектов БД в файлах данных и позволяющая выполнять оптимизирующие операции (дефрагментацию); утилита, показывающая информацию о сессиях, потребляющих наибольшее количество ресурсов (есть возможность сортировки сессий по различным параметрам, для любой из выбранных сессий можно легко "спуститься" по лестнице детализации информации о ней вплоть до используемых курсоров и планов выполнения соответствующих им запросов). Наконец, есть еще две утилиты, стоящие несколько особняком. Это Oracle Trace - управляемая событиями трассировка - и Oracle Expert - экспертная система, проводящя анализ структуры, параметров и функционирования СУБД и генерирующая рекомендации (а также готовые административные скрипты) для ее оптимизирующей настройки.

***Поддержка параллельных систем.*** Одно из общепризнанных достоинств сервера Oracle - его высокая степень масштабируемости: как горизонтальной, так и вертикальной. Oracle владеет в настоящий момент абсолютными рекордами производительности как в OLTP-тестах TPC-C (причем этот рекорд держится с апреля 1996 года), так и в DSS-тестах TPC-D (в варианте с объемом данных 300 Гбайт). Наиболее широко распространены симметрично-параллельные (SMP) системы, т. е. такие, где процессоры равноправно используют все остальные системные ресурсы (прежде всего оперативную память и диски), являющиеся общими для них. Количество процессоров в таких системах, предлагаемых на рынке, может доходить до 64. Для SMP-систем часто еще употребляют определение "система с полным разделением ресурсов" (shared-everything system). Следующий тип параллельной архитектуры - кластер: в нем узлы, имеющие свою собственную оперативную память (а возможно и собственные диски), через специальный контроллер имеют доступ к общим дискам ("система с разделяемыми дисками" - shared-disks system). Как правило, каждый из узлов кластера представляет собой SMP-систему, а количество узлов в кластерах, предлагаемых на рынке, доходит до 8. Наконец, третий тип архитектуры - массивно-параллельный (MPP). В ней узлы живут практически независимой жизнью, но между ними каким-то образом реализуется очень быстрая связь. Количество узлов в такой системе вполне может достигать ста и больше. Безусловно система должна в той или иной степени обеспечивать взаимодействие и совместное пользование ресурсами для своих узлов, тем не менее к системам с данной архитектурой часто применяют определение "система без разделения ресурсов" (shared-nothing system).

Сервер Oracle в любой конфигурации поддерживает параллелизм при выполнении потока операций в SMP-архитектуре, для параллельного выполнения отдельных запросов требуется установка Parallel Query Option. Для кластеров и MPP-систем Oracle предлагает архитектуру, позволяющую всем узлам этих систем параллельно осуществлять доступ к одной БД: чтобы добиться этого, достаточно установить Parallel Server Option. Для обеспечения параллелизма в SMP-системах Oracle предлагает возможность использования многопотоковых разделяемых серверных процессов.

Опция Oracle Parallel Server позволяет нескольким узлам системы (фактически всем, функционирующим в данный момент времени) параллельно работать с одной БД, находящейся на общих дисках (в MPP-системе это будут "виртуальные" общие диски, поддерживаемые ОС). Пользовательские сессии взаимодействуют каждая со своим узлом, но при этом фактически работают с одними и теми же данными (помимо возможности использования полной мощности параллельной системы для работы с БД). Можно заметить, что в Oracle8 даже эта операция не обязательна: новая версия сервера позволяет выполнять автоматическое переключение сессий со сбойного узла, так что, например, прерванные запросы попросту продолжают выполняться после небольшой задержки.

Однако нельзя утверждать, что при применении OPS не возникает никаких проблем. По сравнению с SMP-системами возникает одна проблема: синхронизация кэшей в оперативной памяти узлов – каждый узел системы кэширует данные БД в своей оперативной памяти и может держать их там достаточно долгое время без переписывания на диск. Если один из узлов модифицировал некую запись БД, но не переписал ее на диск, то при обращении к той же записи другой узел не имеет права ни пользоваться ее копией в своей памяти (она уже не актуальна), ни даже считать ее с диска. Для разрешения этой проблемы вводятся блокировки параллельного кэша: при модификации данных узел параллельной системы как бы вешает на них свой "замок", так что любой другой узел при обращении к этим данным должен сначала "снять замок", что включает в себя передачу ему актуальных данных. Ясно, что если различные узлы будут часто модифицировать одни и те же данные, то блокировки параллельного кэша могут заметно снизить производительность сервера в целом.

К сожалению, от данной проблемы нельзя полностью избавиться ни с помощью технических ухищрений, ни с помощью альтернативных решений. К счастью, если пользователи, работающие с разными узлами, редко модифицируют одни и те же записи, то и блокировки параллельного кэша возникают редко. Такой режим легко обеспечивается, если на разные узлы сервера "назначаются" пользователи, работающие с разными приложениями, или работающие с данными различных отделов (филиалов) и пр. Приложения, осуществляющие "хаотичные" обращения к большой БД, также имеют слабую тенденцию к порождению блокировок параллельного кэша. Тем не менее, распределение пользователей между узлами сервера должно осуществляться не наобум, а с учетом того, с какими данными и в каком режиме они работают. Как бы то ни было, OPS уже достаточно давно и успешно используется - особенно в инсталляциях, требующих повышенной надежности системы. Нелишне заметить, что и рекорд в тестах TPC-C поставлен с использованием OPS на кластере (Digital Alpha 8400). Надо сказать, что до последнего времени понятия "кластер" и "параллельный сервер" ассоциировались только с весьма мощными и дорогостоящими конфигурациями аппаратуры. Отчасти это было связано с реальными потребностями рынка, а отчасти с тем фактом, что поддержка кластерного режима работы требует весьма значительных системных ресурсов. Одним из первых пожирателей ресурсов является менеджер распределенных блокировок (Distributed Locks Manager - DLM). Это программный компонент (реализованный обычно в виде набора процессов), обычно поставляемый фирмой-разработчиком ОС, задача которого - управление доступом к разделяемым ресурсам на уровне системы в целом. Именно с помощью DLM Oracle реализует блокировки параллельного кэша и вообще синхронизацию работы узлов. Универсальность DLM в сочетании с тем, что он является "внешней составляющей" OPS, приводит к тому, что общее количество блокировок параллельного кэша становится критичным ресурсом. Чтобы снизить потребность в нем, в Oracle 7.3 введен ряд усовершенствований в управлении выделением этих блокировок, но для радикального решения проблемы безусловно требовался другой подход к реализации DLM. В частности, по этой причине уже в версии 7.3 Oracle постепенно переходит к реализации DLM собственными средствами в составе ядра сервера - окончательно этот процесс завершен в Oracle 8. Как бы то ни было, уже в релизе Oracle 7.3.3 существует параллельный сервер для кластеров, функционирующих под управлением таких "легковесных" ОС, как SCO UnixWare и Windows NT (последняя - как для платформы Intel, так и для DEC Alpha).

При параллелизме, в задачах типа DSS, при выполнении отдельных операций оптимизатор выбирает один из возможных алгоритмов выполнения запросов (при этом важно, что в Oracle он с самого начала при оценке стоимости того или иного решения учитывает заданную для данного запроса степень параллелизма), затем каждый шаг алгоритма разбивается на несколько параллельных потоков. Координатор выполнения запроса запускает нужное число процессов (при этом используются все наличные процессоры - включая различные узлы кластера или MPP-системы) и обеспечивает как внутриоперационный (параллельные потоки внутри шага алгоритма), так и межоперационный параллелизм. В список операций, подлежащих распараллеливанию, помимо просмотра таблиц включены также все алгоритмы соединения (и "антисоединения") таблиц, сортировки, операции агрегирования, вложенные подзапросы, объединения и некоторые другие. Кроме того, возможно параллельное выполнение таких операций, как создание таблицы по результатам запроса, загрузка данных, сброс и восстановление БД, выполнение операций тиражирования данных. В Oracle8 к этому списку добавятся операции INSERT, UPDATE и DELETE.

Одним из наиболее фундаментальных вопросов, которые приходится решать при реализации параллельного выполнения запросов, является выбор метода распределения данных между параллельными потоками при выполнении таких операций, как полный просмотр таблиц. Самым простым (и исторически реализованным первым - фирмой Tandem) методом является "привязка" параллелизма к статическому разбиению нужных таблиц на разделы, проводимому по правилу, заданному администратором системы. Этот метод и до сих пор является краеугольным камнем параллелизма в ряде СУБД.

В идее разбиения таблиц на разделы безусловно есть серьезные положительные стороны, особенно когда это разбиение осуществляется на основе диапазонов значений содержательных параметров либо функций от них. Тогда, во-первых, может быть облегчена работа администратора БД в случае, когда таблица содержит большой объем данных (например, если разбить таблицу фактических продаж по месяцам, то можно выполнять сбросы только последнего раздела таблицы), во-вторых, становится возможным исключение разделов при выполнении запросов, содержащих условие на параметр разбиения. Однако когда параллелизм в выполнении запросов ставится в зависимость от статичного разбиения таблиц, это приводит к ряду проблем. Дело в том, что для достижения оптимального параллелизма в этом случае требуется (по очевидным причинам), чтобы данные были распределены по разделам равномерно. В принципе этого нетрудно добиться, если помещать каждую новую запись в новый раздел по циклическому алгоритму (round-robin). Но в этом случае, как нетрудно заметить, полностью теряются указанные выше два преимущества. И наоборот, если выполнять разбиение по содержательному критерию, то весьма часто получается, что данные распределяются по разделам неравномерно, что неизбежно приводит к тому, что, закончив свою работу, параллельные процессы ждут "отстающего товарища", которому не повезло с разделом. Если речь идет об устоявшихся (т. е. фактически не обновляемых) данных и о конкретном запросе с небольшими вариациями, то практически всегда можно найти некий компромиссный вариант разбиения, но в реальных системах типа DSS запросы как правило носят нерегламентированный характер (ad-hoc), а данные - опять-таки как правило - периодически обновляются. Все это как минимум приводит к серьезной административной работе, связанной с перестройкой разделов (что становится попросту обязательным, если требуется сменить степень параллелизма), но даже это не гарантирует оптимального параллельного выполнения запросов.

Такие соображения побудили разработчиков Oracle7 отказаться от принципа "статичного" параллелизма и реализовать алгоритм динамического разбиения таблиц при параллельном выполнении запросов. Упрощенно его суть в том, что таблица логически "разбивается" непосредственно при выполнении запроса в соответствии с заданной степенью параллелизма. Это не означает, впрочем, что она попросту делится на равные части произвольным образом - все гораздо изощреннее. Дело в том, что скорость обработки одного и того же объема данных в разных разделах может быть различна в зависимости как от характера запроса, так и от того, на каких физических устройствах располагается динамический раздел, да и от других порой трудно предсказуемых причин. Поэтому таблица делится реально на число разделов гораздо большее, чем степень параллелизма (и разделы эти бывают различного размера), а их назначение параллельным процессам регулируется динамически в зависимости от того, с какой скоростью они справляются с уже порученной работой.

Надо сказать, что алгоритм динамического разбиения таблиц весьма непрост, и было бы нечестно утверждать, что в нем с самого начала все было сделано самым оптимальным образом. Однако одно из самых важных преимуществ этого алгоритма в его гибкости, поэтому в него постоянно вносились усовершенствования на основании накопленного опыта эксплуатации в реальных инсталляциях, в результате чего от релиза к релизу Oracle7 добивался все более оптимальных характеристик параллелизма в выполнении запросов. К примеру, в релизе 7.3 основные усовершенствования были связаны с поддержкой MPP-архитектур. Дело в том, что в них диски не равноценны по скорости доступа для каждого из узлов системы (к "своим" дискам доступ осуществляется быстрее, чем к "чужим"), поэтому и динамические разделы стали выделяться параллельным процессам преимущественно на локальных для соответствующих узлов дисках (преимущественно - опять-таки потому, что, завершив свою "локальную" работу, процесс не прекращает деятельность, а начинает помогать "отстающим").

Как бы то ни было, сейчас можно с уверенностью констатировать, что метод динамического разбиения таблиц оправдал себя, позволив при минимальной дополнительной нагрузке на администратора БД добиться, тем не менее, практически оптимального распраллеливания выполнения запросов. Высокая масштабируемость Oracle в параллельном выполнении запросов на системах с различной архитектурой иллюстрируется также и тем фактом, что Oracle 7.3 сумел показать рекордные параметры в TPC-D тесте (в варианте с объемом данных 300 Гбайт) как среди MPP-систем (на IBM SP/2), так и среди SMP-систем (на Sun Enterprise Server 10000).

Oracle 7, к сожалению, не включает в себя явной операции построения статических разделов таблицы (эта возможность вводится в Oracle 8), но в неявном виде это тем не менее можно сделать с помощью имитации разделов отдельными таблицами, объединенными в единое представление с помощью операции UNION ALL. При выполнении запросов к такому представлению оптимизатор Oracle7 трактует его именно как таблицу, разбитую на разделы, в частности выполняет исключение разделов, если это возможно.

Oracle традиционно славится как поставщик СУБД для крупных инсталляций, однако в связи с этим бытует (и активно поддерживается конкурентами) также и мнение о том, что для небольших систем Oracle слишком тяжеловесен, сложен, дорог и пр. Oracle прикладывает немало усилий, чтобы по всем параметрам (включая цены) утвердиться в качестве основного поставщика во всех сегментах рынка СУБД, начиная с небольших рабочих групп. С 1994 года помимо уже привычного "сервера масштаба предприятия" поставляются другие его варианты: "сервер для рабочих групп" (Workgroup server) и "персональный Oracle" (Personal Oracle) в двух редакциях - полной и "облегченной" (Personal Oracle Lite). В этих продуктах особый упор сделан на их относительную дешевизну, простоту установки и сопровождения. При этом все варианты сервера Oracle функционально идентичны, за исключением некоторых опций (только в нынешней версии Personal Oracle Lite отсутствует часть базовой функциональности: он не поддерживает многопользовательские схемы данных и процедурные расширения SQL).

Oracle 7 в одинаковой степени может быть оптимизирован и для OLTP-приложений, и для приложений DSS, причем их вполне можно исполнять одновременно, не беспокоясь о дополнительных блокировках, модах изоляции и прочих темах, способных вызвать головную боль у знакомых с ними на практике специалистов при одном только их упоминании. Это не означает впрочем, что оба режима будут одинаково оптимальны при одном и том же выборе параметров системы. Безусловно, по крайней мере часть параметров требуют разного подхода при их оптимизации для OLTP- и DSS-систем. По этой причине поддержка "смешанного" режима обязательно сопряжена с некоторым компромиссом, и не следует трактовать приведенное утверждение как рекомендацию совмещать оперативную систему с хранилищем данных. Более разумно на практике говорить о том, что, скажем, если основной режим системы - OLTP, то совсем не обязательно дожидаться ночной паузы в работе, чтобы выполнить на ней сложный - но срочный - отчет, и Oracle гарантирует, во-первых, корректность результатов отчета, во-вторых, что его выполнение не повлияет сколько-нибудь заметно на работу пользователей.

Oracle поддерживает ***любой тип данных.*** В сущности, речь идет о расширении стандартного набора типов данных, характерного для РСУБД, а в перспективе о переходе к объектно-реляционной модели СУБД. В свою очередь, эта задача может быть разделена на две:

* поддержка поставщиками СУБД дополнительных "базовых" типов данных;
* возможность расширять набор типов данных за счет модулей третьих производителей или самими пользователями.

Oracle развивает свой сервер в обоих направлениях. В версии 7.3 уже поддерживается несколько новых типов данных: неструктурированные тексты, пространственные данные, видеоданные. Собственно говоря, хранить такие данные в БД и осуществлять к ним доступ можно было и раньше: новизна в том, что если раньше этот доступ осуществлялся через самостоятельно работающие серверные процессы, и для работы с ними требовалось использование специального интерфейса на уровне приложений, то теперь данная функциональность интегрирована в "базовый" сервер.

Опция для работы с пространственными данными (Spatial Data Option) фактически вводит тип данных "пространственная точка" и операции над ним в СУБД, позволяя хранить соответствующие данные в таблицах оптиальным образом и на порядок (а порой и на два) ускорять выполнение запросов.

Что касается видеоданных, то соответствующая им опция - Video Option - единственная, "живущая самостоятельной жизнью" по отношению к серверу РСУБД (но не к БД). Более того, рекомендуется конфигурация, в которой Video Server запускается на отдельном компьютере от сервера БД. Связано это с тем, что воспроизведение видеофрагментов в реальном времени (особенно по нескольким каналам) - что как раз и обеспечивает Video Server - трудно совместимо на современных массово производимых компьютерах с функционированием сервера СУБД из-за чисто аппаратных ограничений. Тем не менее приложение, работающее с Video Server, может осуществлять поиск видеофрагментов по описательным атрибутам и воспроизведение этих фрагментов - как единую интегрированную операцию.

Относительно Web Option, пожалуй, не совсем правильно говорить о функциональных расширениях сервера, поскольку, в сущности, главная задача опции - обеспечение интерфейса с Web Application Server и соответственно через него с пользователями Intranet/Internet.

Oracle OLAP Option едва ли можно было рассматривать как интегрированную компоненту сервера Oracle (продукты OLAP работают с собственным - многомерным - представлением данных, хранимым отдельно) до недавнего времени, когда с помощью Access Manager появилась возможность устанавливать динамическую связь многомерного куба OLAP с реляционными данными, стирая тем самым грань между MOLAP и ROLAP (для аналитика, работающего с приложениями OLAP, стало совершенно незаметно, работает ли он с предварительно сформированным многомерным кубом или с динамическим многомерным представлением реляционных данных).

***Развитие подхода в Oracle 8.*** В отличие от Oracle 7 восьмая версия сервера Oracle не просто предоставляет расширенный набор встроенных типов данных, но и позволяет конструировать новые типы данных со спецификацией методов доступа к ним. Это означает фактически, что разработчики получают в руки не просто систему для хранения и обработки, скажем, видеоданных (что, понятно, нужно далеко не в каждом приложении), а и инструмент, позволяющий строить структурированные типы данных, непосредственно отображающие сущности предметной области. Влияние этого фактора на возможности разработчиков можно сравнить с эффектом от перехода на реляционные СУБД в начале 80-х годов.

Oracle8 фактически опирается на новый стандарт SQL, позволяющий описывать определения новых типов объектов, состоящих из атрибутов (скалярных - т. е. других типов, множеств объектов, ссылок на объекты), и обладающих ассоциированными с ним методами. Любая колонка таблицы может быть любого типа, поддерживаются также вложенные таблицы и массивы объектов переменной длины.

Что касается методов доступа, то они могут быть определены несколькими способами. Более простой из них предполагает контроль ядра сервера за выполнением метода: это достигается, если методы реализуются на PL/SQL (который также расширен для поддержки объектно-реляционных структур) или Java. Поскольку PL/SQL практически не уступает по своей функциональности универсальным языкам программирования, для подавляющего большинства составных типов данных таких возможностей будет достаточно. Если же новый тип данных требует специальной обработки, не реализуемой стандартными средствами ядра СУБД (к примеру, работа с мультимедийными данными, хранимыми в BLOB-полях в БД), можно использовать вызовы внешних процедур (call-outs), которые могут быть написаны, допустим, на языке C.

При использовании внешних процедур возникает серьезная проблема организации их взаимодействия с ядром сервера. Наиболее соблазнительная - на первый взгляд - идея включения их непосредственно в ядро таит в себе угрозу нарушения стабильности этого ядра, поскольку оно оказывается незащищенным перед "чужим" кодом, и, следовательно, при любом сбое (или неправильном функционировании) становится практически невозможно определить, явилось ли это следствием ошибки в самом ядре, или же это "наведенный" эффект от внешней процедуры. Поэтому Oracle изначально отказался от такой идеи и реализует взаимодействие ядра сервера с внешними процедурами в защищенном режиме (т. е. в различных адресных пространствах). Для реализации такого взаимодействия и для доступа из внешних процедур к данным БД требуется наличие специального программного интерфейса. Oracle в данном вопросе пошел по пути поддержки имеющегося стандарта интерфейса Corba, позволяя оформлять расширения ядра сервера как "картриджи данных" (Data Cartridges), входящие в более общую архитектуру сетевых вычислений (Network Computing Architecture).

Для обеспечения постепенной миграции приложений и данных в новую объектно-реляционную среду введены объектные представления (object views), которые позволяют использовать в новых приложениях объектный интерфейс, работая при этом с обычными реляционными таблицами (т. о. сохраняя работоспособность старых приложений).

Новые возможности в администрировании Oracle 8 - управляемые сервером сбросы и восстановления (собственно говоря, это расширенная интеграция применявшейся в Oracle 7 утилиты Enterprise Backup), централизованное хранение паролей (в Oracle 7 достигалось при использовании Advanced Networking Option или при идентификации пользователей через ОС, имеющую соответствующую функциональность), контроль за назначением и устареванием паролей (в Oracle7 - при идентификации пользователей через ОС). Новые режимы взаимодействия с сервером - поддержка очередей приоритетных сообщений, задающих описание транзакции или ее части (эта функциональность, кстати, может быть использована мониторами транзакций), возможность мультиплексирования сессий как на физических, так и на логических каналах связи. Фактическое снятие ограничений на количество BLOB-колонок в таблицах, возможность их тиражирования. Возможность разбиения BLOB-полей и их отдельного хранения (даже вне БД). Расширение функциональных возможностей тиражирования данных, введение программного интерфейса тиражирования, позволяющего реализовать поддержку репликации с самыми разнообразными системами хранения данных. Поддержка таблиц, целиком хранимых в индексах. Это далеко не полный список. Большая часть новшеств возникла не на пустом месте, а скорее представляет собой развитие тех черт, которые уже содержались в том или ином виде в Oracle7. Это не случайность: Oracle гарантирует совместимость версий сервера снизу вверх, при переходе к Oracle8 пользователям даже не потребуется перестраивать свои БД.

## 1.1.4 MS SQL Server

Компания Microsoft широко известна рынке ПО. В 1988 г. фирма Microsoft совместно со своими партнерами Acton-Tate и Sybase представили свою первую версию SQL Server, построенную под операционную систему OS/2. В дальнейшем, фирма Microsoft перенесла SQL Server под Windows NT. Эти изменения потребовали коренных перестроек в ядре SQL Server, но, тем самым, обеспечили продукту SQL Server мощность мультипроцессорной RDBMS в среде Windows NT. В 1992 г. фирма Microsoft начала процесс отделения от Sybase и стала сосредотачивать больше внимания на собственной версии SQL Server. В конце концов, Microsoft и Sybase закончили совместную работу, и к Microsoft перешел полный контроль над разработкой SQL Server. Далее в SQL Server были добавлены следующие возможности: поддержка RISC-платформы, MAPI-интерфейс для разработки приложений, выполняющих запросы в базу данных, инструменты переноса данных, интеграция с объектами OLE и системой программирования Visual Basic и многое другое.

Microsoft SQL Server 6.5 является одним из наиболее стремительно развивающихся серверов баз данных на рынке корпоративных СУБД. Разумеется, невозможно подробно остановиться на характеристиках этого продукта в той мере, в какой это хотелось бы сделать и какой он, безусловно, заслуживает. рассмотрим хотя бы некоторые из базовых возможностей Microsoft SQL Server 6.5 применительно к перечисленным выше функциям сервера баз данных.

Симметричная мультипроцессорная ***архитектура Microsoft SQL Server 6.5*** предусматривает использование "родных" сервисов операционной системы Windows NT для управления потоками, памятью, операциями дискового чтения/записи, сетевыми службами, функциями безопасности, а также для поддержки параллельного выполнения потоков на нескольких CPU. Использование потоков Windows NT позволяет MS SQL Server автоматически масштабироваться при работе на многопроцессорных платформах, что исключает необходимость дополнительной конфигурации или программной настройки. Например, на Comdex была продемонстрирована работа MS SQL Server на платформе AlphaServer 8400 производства Digital, оснащенным 12 процессорами, 28 Гбайт памяти и 39-ти терабайтным хранилищем. В отличие от большинства распространенных СУБД, вынужденных иметь в своем составе механизмы дублирования ядра операционной системы для обеспечения кросс-платформенной переносимости, MS SQL Server обладает достаточно легковесной прозрачной архитектурой, не перетяжеленной несвойственными ей функциями. В результате, например, при смене типа процессора не требуется заново приобретать MS SQL Server для новой аппаратной платформы. Он ставится, по определению, на все, на чем работает Windows NT (на сегодня это Intel, Alpha, MIPS и PowerPC). По мере того как Windows NT завоевывает все большее признание и все ведущие производители СУБД уже выпустили версии своих продуктов под этой операционной системой или уже заявили о своей готовности это сделать в ближайшее время, изначальная ориентированность MS SQL Server 6.5 на тесную интеграцию с Windows NT выступает в качестве одного из серьезных преимуществ.

На каждое пользовательское соединение в MS SQL Server назначается отдельный рабочий поток (порядка 55К) в рамках единого серверного процесса. Так как каждый из этих потоков в действительности является потоком Win32, на них распространяются соответствующие функции контроля операционной системы, включая защиту памяти, правила доступа к оборудованию и планирование выполнения потоков во времени (thread scheduling). Это предоставляет улучшенные способности к масштабированию при росте числа одновременно работающих пользователей, динамическую балансировку при загрузке процессоров и повышенную надежность, так как пользовательские запросы, исполняющиеся на разных потоках, защищены друг от друга. Несмотря на то что пул соединений ограничен 1024 потоками, динамическое управление пользовательскими соединениями и свободными потоками позволяет увеличить эту величину до 32 767. Кроме этого, другие пулы потоков могут использоваться для параллельного выполнения операций сканирования данных, удаления и обновления, резервного копирования, проверки целостности базы, индексирования, асинхронного опережающего чтения данных в кэш на основе алгоритмов предсказания, создания и управления курсорами и т. д.

Сетевые службы Windows NT обеспечивают MS SQL Server поддержку протоколов TCP/IP, NWLink IPX/SPX, Named Pipes (NetBEUI), Banyan Vines, AppleTalk (ADSP) и DECNet. В версии 6.5 к ним добавилась дополнительная сетевая библиотека – multi-protocol network library, которая "умеет слушать" порты TCP/IP, сокеты SPX или поименованные каналы (named pipes), которые обычно выбираются динамически. Несомненным достоинством multi-protocol является наличие сетевого сервиса, обеспечивающего взаимодействие между процессами при помощи вызовов удаленных процедур, что позволяет, например, использовать шифрование при передаче данных.

Многопоточное ядро и интеграция со службами планирования потоков Windows NT обеспечивает высокую ***производительность*** MS SQL Server при обработке OLTP- и DSS-запросов, что особенно заметно при одновременной работе нескольких сотен пользователей. В опубликованных результатах по тестированию MS SQL Server 6.5 на максимальное число одновременно работающих пользователей приводится цифра 3500, хотя известны реально работающие приложения, где нагрузка доходила до 5000 одновременных пользовательских соединений. За период с октября 1996 г. по декабрь 1997 г. производительность MS SQL Server , измеренная по тестам TPC-C, выросла более чем в 3 раза. Для сравнения заметим, что ежедневный объем транзакций в расчетной системе VISA составляет от 10 до 40 млн. Темп 7,5 тыс. транзакций в минуту означает, что один MS SQL Server способен при режиме работы 24х7 обслужить немногим менее 11 млн. транзакций в сутки. Существует еще один параметр, тесно связанный с производительностью, который, не являясь в строгом смысле слова техническим, очень популярен на Западе при оценке возможностей того или иного сервера баз данных, так как от него существенно зависит стоимость владения продуктом. Речь идет об удельной цене за транзакцию в минуту, иными словами, сколько придется заплатить за достижение такой скорости обработки запроса. За тот же самый период, в течение которого рассматривался рост производительности, показатель "цена/производительность" стал менее 65 долл. за транзакцию в минуту, что говорит о разумной стоимости систем на базе MS SQL Server при высоких требованиях к скорости обработки.

***Распределенная среда управления.*** В состав MS SQL Server 6.5 входит свыше 20 графических средств управления и утилит командной строки. Кроме этого, MS SQL Server 6.5 включает Web-assistant - программу мастер для подготовки публикации на Web-страницах данных из базы, SQL Mail - утилиту, обеспечивающую интеграцию с электронной почтой MS Mail или MS Exchange, MS Distributed Transaction Coordinator (MS DTC) для проведения распределенных транзакций и некоторые другие средства. SQL Server, MS DTC и SQL Executive функционируют как сервисы операционной системы. Согласованная работа этих компонентов достигается благодаря трехуровневой архитектуре SQL -DMF (Distributed Management Frame-work).



Легко масштабируемая распределенная среда управления позволяет значительно упростить процессы централизованного контроля над многими серверами, которые могут объединяться в группы по соображениям безопасности или с административными целями, и их объектами.

SQL Enterprise Manager интегрирует в себе все функции управления, включая создание баз данных и объектов внутри них, назначение прав доступа, резервное копирование, тиражирование и т.д. При желании имеется возможность автоматизировать процесс составления плана поддержки базы при помощи специальной программы-помощника (Database Main-tenance Wizard). Различные подходы к системному администрированию зачастую могут содержать ряд малоприятных моментов, например необходимость выполнять резервное копирование базы, командировать сотрудников в какой-нибудь удаленный филиал, где отсутствует должным образом подготовленный IT-персонал. MS SQL Server 6.5 позволяет решить эти проблемы, во-первых, за счет централизованного управления удаленными серверами, во-вторых, за счет наличия мощного средства диспетчеризации задач во времени, предоставляемого SQL Executive. Для каждой административной функции может быть назначен временной график ее выполнения. Практически все СУБД содержат развитые средства по ликвидации тех или иных неблагоприятных последствий. Microsoft SQL Server, помимо этого, предоставляет обширный инструментарий диагностики, позволяющий своевременно предотвратить причины сбоев. Утилиты SQL Performance Monitor и Alert Manager могут использоваться для программирования реакции сервера на различные классы событий, возникающих в системе, в том числе и на бизнес-события, MS SQL Server может послать вам (или указанным вами лицам) по электронной почте или на пейджер соответствующее предупреждение и/или выполнить предусмотренный вами скрипт, cmd- или exe-файл для устранения ошибки, а также зафиксировать появление этого события в системном журнале. В целом можно сказать, что распределенная среда управления позволяет существенно упростить жизнь администратора базы данных.

***SQL-DMO (Distributed Management Objects).*** В качестве промежуточного слоя в архитектуре распределенной среды управления выступают распределенные объекты управления (DMO), которые играют исключительно важную роль в концепции построения MS SQL Server и потому заслуживают более тщательного рассмотрения. По мере того как приложения приобретали все менее централизованный характер, поддержка распределенных баз данных становилась одним из самых актуальных вопросов построения современных СУБД. SQL Enterprise Manager позволяет осуществлять удобное администрирование распределенных серверов из единого центра, однако наряду с этим хотелось бы иметь возможность программного обращения к административным функциям из высокоуровневых языков. Обычно использовавшимся для этих целей в других СУБД сценарным языкам типа REXX или PERL недоставало функциональных возможностей, библиотек классов, отладчика и т. д.

Поэтому в случае с Microsoft SQL Server был избран более открытый подход: сервер был разработан как совместно с набором объектов управления, которые могли быть вызваны из любого языка программирования, поддерживающего технологию СОМ (Component Object Model). MS SQL Server 6.5 предоставляет интерфейс OLE Automation с более, чем 70 объектами, обладающими 1500 свойствами.

Это означает, что фактически любая из административных задач, включая операции над базами данных, ограничениями (constraints), триггерами, таблицами, представлениями, полями, индексами, пользователями, группами, публикациями и пр., может быть оформлена как вызов соответствующего метода соответствующего объекта и выполнена (при наличии прав доступа) из Visual Basic, Visual C++, Visual J++, Visual FoxPro и т. д.

В основе практически всех вышеперечисленных утилит лежит код языка ***Transact-SQL***. Microsoft SQL Server 6.5 был первой СУБД, прошедшей сертификационные испытания Правительства США на соответствие входному уровню (entry level) федеральных стандартов обработки информации (FIPS) 127.2. Эти тесты основываются на известных стандартах ANSI SQL92 и включают дополнительные требования, в частности по поддержке трехуровневых архитектур. MS SQL Server 6.5 содержит большое количество черт и функций, относящихся к более высоким уровням стандарта ANSI SQL92 (intermediate и full), например скроллируемые в обоих направлениях курсоры с абсолютным и относительным позиционированием. Насколько мне известно, ни одна из СУБД на сегодня не достигла полного соответствия уровню ANSI SQL92, более высокому, чем входной.

Transact-SQL включает операторы для изменения настроек сервера, пользовательской сессии, просмотра и редактирования данных, создания и модификации баз и их объектов. В настоящее время в MS SQL Server поддерживается только строгий (restrict) тип ссылочной целостности.

Помимо обычных хранимых процедур MS SQL Server предоставляет возможность динамической загрузки и выполнения функций, которые называются расширенными хранимыми процедурами и выполнены в виде dll-библиотек. Расширенные процедуры объединены в dll-библиотеки в целях повышения производительности по сравнению с оформлением в виде отдельных процессов. Кроме расширенных процедур, входящих в Transact-SQL, MS SQL Server позволяет создавать пользовательские расширенные процедуры c использованием кода на C при помощи MS Open Data Service (ODS) API. MS ODS является мощным средством разработки и применяется также для создания шлюзов к неподдерживаемым штатно пользовательским ресурсам, программирования задач аудита, извещения о событиях и пр. Дополнительный уровень защиты обеспечивается обработчиком исключений MS SQL Server, который предотвращает сервер от сбоя в случае нарушений защиты памяти в расширенной процедуре.

В версии 6.5 в Transact-SQL вошли хранимые процедуры для работы с объектами OLE Automation. Таким образом, фактически появилась возможность писать расширенные хранимые процедуры на любом языке программирования, поддерживающем создание серверов OLE Automation: Visual Basic версии 4 и выше, Visual FoxPro 5.х и т. д. Экземпляр соответствующего объекта создается непосредственно в коде Transact-SQL.

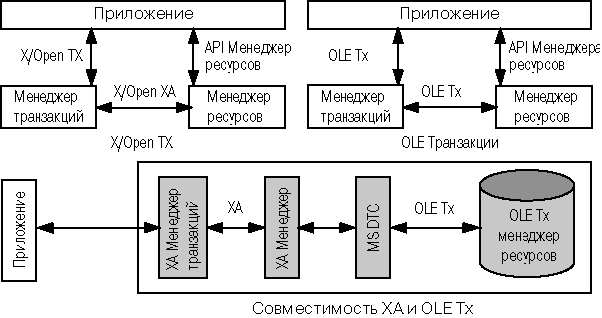
Механизм вызовов удаленных хранимых процедур (RPC) позволяет организовать межсерверное взаимодействие и является мощным средством построения распределенных баз. RPC означает вызов с одного сервера процедуры, принадлежащей другому серверу баз данных. Клиентское приложение может вызывать процедуру на своем основном сервере, которая неявно для клиента может порождать каскад вызовов удаленных хранимых процедур на других серверах. RPC представляет собой достаточно удобный способ работы с распределенными данными без необходимости внесения изменений в клиентскую часть приложения.

***MS Distributed Transaction Coordinator (DTC).*** Создание распределенных приложений приводит к тому, что транзакции также приобретают распределенный характер. Структуризация приложения в виде многих самостоятельных компонентов способна существенно повысить масштабируемость и повторную используемость, а также упростить его разработку. Однако при этом необходимо иметь в виду, что сбой в работе одного из компонентов (например, в результате выхода из строя компьютера, на котором она была запущена) не должен сказываться на целостности функционирования всего приложения в целом, т. е. компонент может временно выключиться из согласованной работы приложения, но связанные с ней сообщения должны быть обработаны корректно. Участниками распределенной транзакции являются приложение, менеджеры транзакций, менеджеры ресурсов и сами ресурсы, затрагиваемые транзакцией. В этой цепочке MS DTC выполняет роль менеджера транзакций.

MS DTC содержит компоненты клиентской и серверной настройки. Установка клиентского компонента требуется только в том случае, если данный клиент будет сам инициировать распределенные транзакции, а не использовать транзакции, начатые на серверной стороне. MS DTC достаточно легок и удобен в настройке и управлении.

OLE Transaction выгодно отличается от некоторых других распространенных стандартов тем, что построен на основе объектной модели и поддерживает приложения, работающие одновременно со многими потоками. OLE Transaction обладает улучшенными характеристиками по сравнению с ранее разработанными стандартами, лишенными, например, возможности восстановления (recovery), инициированного менеджером ресурсов. Тем не менее при помощи процесса XA Mapper MS DTC, выполняющего роль переводчика между XA и OLE Transaction, обеспечивается определенное взаимодействие с продуктами, совместимыми со стандартом X/Open DTP XA.

MS DTC может участвовать в транзакциях, координируемых мониторами транзакций Encina, TopEnd и Tuxedo, для которых он выглядит как некоторый менеджер ресурсов. Стандарт OLE Transaction содержит возможности расширения для работы с широким спектром транзакционно-защищенных ресурсов, к которым могут быть отнесены документы, образы, очереди сообщений и другие виды плохо структурированной информации.



MS SQL Server использует следующие ***типы блокировок***:

* shared - для операций, не изменяющих содержимое данных, например select;
* update - когда сервер намеревается изменить данные, во время непосредственной записи обновлений этот тип блокировки изменяется на exclusive (для таблиц см. intent);
* exclusive - при модификации данных (insert, update, delete).

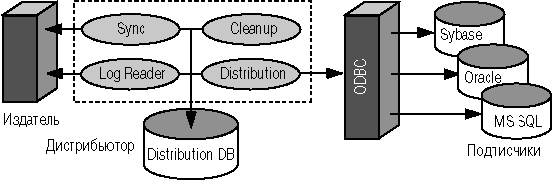
***Надежность хранения информации.*** В критических для бизнеса приложениях, когда сервер СУБД должен быть постоянно доступен для клиентов, большинство профилактических работ по поддержке базы данных приходится выполнять фактически в режиме on-line. MS SQL Server обладает возможностями динамического резервного копирования данных. В случае сбоя оборудования, отключения питания и т. д. механизм автоматического восстановления MS SQL Server восстанавливает все базы данных до их последнего целостного состояния без вмешательства администратора. Все завершенные, но не отраженные в базе транзакции из журнала транзакций применяются к базе данных (это фактически то, что происходит при событии chekpoint), а незавершенные транзакции, т. е. те, которые были активными на момент сбоя , вычищаются из журнала.

MS SQL Server 6.5 предусматривает возможность зеркалирования устройств, переключения на зеркальные устройства в качестве основных, выключения зеркалирования и уничтожения зеркального устройства также "на лету", т. е. без остановки штатной работы сервера по обслуживанию пользовательских запросов. Зеркалирование и дуплексирование устройств для работы с MS SQL Server может быть также выполнено средствами Windows NT, а также на аппаратном уровне (поддержка различных RAID-систем и т. д.). Появление следующей версии MS SQL Server должно обеспечить работу серверов в кластере как единого виртуального сервера.

Наличие развитого ***механизма тиражирования*** в любой серьезной системе управления базами данных обуславливается необходимостью приближения данных к местам их непосредственного потребления, что является особенно важным фактором при построении витрин данных в системах принятия решений, разгрузки приложений от избыточных функций чтения/поиска при создании отчетов и т. д. Создание распределенных приложений с использованием средств тиражирования положительно сказывается на относительной автономии сайтов, повышении масштабируемости и производительности. Традиционно в построении распределенных систем данных существуют два основных подхода. Один из них основан на плотной целостности данных (loose consistency). Протокол двухфазной фиксации гарантирует идентичность данных в любой момент времени на всех узлах сети, однако необходимо иметь в виду, что этот подход требует наличия высокоскоростных каналов передачи данных и постоянной доступности каждого узла. Другой подход, основанный на слабой целостности (loose consistency), допускает, вообще говоря, некоторый временной интервал между внесением изменений в оригинал и их отражением в образе. Приложения, основанные на принципе слабой целостности, являются значительно менее чувствительными к доступности узлов, а также пропускной способности и надежности каналов передачи данных. Тиражирование в MS SQL Server построено на использовании именно второго подхода.

На дистрибьюторе существуют еще два вида процесса: распространение и очистка. Задача распространения создается для каждой пары "тиражируемая база/подписавшаяся база", а задача очистки - для пары "издатель/подписчик". Распространение (distribution task) применяет прочитанные из базы данных распространения sql-команды к базе данных подписчика. Процесс очистки (cleanup task) уничтожает все выполненные работы (т. е. транзакции) из базы данных распространения через некоторый настраиваемый интервал после того, как они были доведены до подписчика. Несмотря на то что организация всего процесса тиражирования может быть записана в кодах при помощи вызовов специальных хранимых процедур, эта черта используется на практике крайне редко и главным образом в целях отладки. В обычных ситуациях настройка и управление тиражированием осуществляются из графической среды SQL Enterprise Manager и планировщика задач SQL Executive.

Соединение дистрибьютора с издателем происходит на основе DB-Library, а с подписчиком - через ODBC. Таким образом, в качестве подписчиков MS SQL Server может выступать широкий спектр ODBC-достижимых ресурсов, к которым, например, относятся другой Access, Sybase, Oracle, DB2 и т. д. Тиражирование в MS SQL Server основано на интегрированном режиме безопасности (см. Безопасность), следовательно, между дистрибьютором и подписчиком должны быть установлены доверительные соединения (trusted connections) с использованием поименованных каналов (named pipes) или мультипротокола. Если серверы находятся в разных доменах, между доменами должны быть установлены двусторонние доверительные отношения. В случае небольших объемов тиражируемых данных издатель часто совмещает с дистрибьютором на одном MS SQL Server.



MS SQL Server обладает обширными возможностями настройки процесса тиражирования. Отметим, что для каждой статьи имеется возможность назначить к тиражированию только необходимые типы транзакций. В зависимости от административного акцента MS SQL Server позволяет организовать подписку на стороне издателя либо на стороне подписчика. Первый вид подписки (push subscription) используется при централизованном распространении, когда подписки создаются "выталкиванием" статей на те или иные серверы-подписчики, которые могут не иметь своих администраторов. Второй вид (pull subscription) предполагает известную автономию сервера-подписчика, администратор которого определяет, какие публикации ему принимать. По умолчанию все публикации создаются со статусом безопасности "неограничено", они видны и на них могут подписаться любые зарегистрированные серверы подписки. Ограниченная публикация может быть выписана только теми серверами, которые имеют на это соответствующие права.

***Безопасность доступа.*** MS SQL Server использует в своей работе сервисы безопасности Windows NT. Напомним, что Windows NT на сегодня сертифицирована по классам безопасности С2/Е3. MS SQL Server может быть настроен на работу в одном из трех режимах безопасности. Интегрированный режим предусматривает использование механизмов аутентификации Windows NT для обеспечения безопасности всех пользовательских соединений. В этом случае к серверу разрешаются только трастовые, или аутентифицирующие, соединения (named pipes и multiprotocol). Стандартный режим безопасности предполагает, что на MS SQL Server будут заводиться самостоятельные login id и соответствующие им пароли. Смешанный режим использует интегрированную модель при установлении соединений по поименованным каналам или мультипротоколу и стандартную модель во всех остальных случаях.

MS SQL Server обеспечивает многоуровневую проверку привилегий при загрузке на сервер. Сначала идентифицируются права пользователя на установление соединения с выбранным сервером и выполнение административных функций: создание устройств и баз данных, назначение прав другим пользователям, изменение параметров настройки сервера и т.д. На уровне базы данных каждый пользователь, загрузившийся на сервер, может иметь имя пользователя (username) базы и права на доступ к объектам внутри нее. Имеется возможность отобразить нескольких login id на одного пользователя базы данных, а также объединять пользователей в группы для удобства администрирования и назначения сходных привилегий. По отношению к объектам базы данных пользователю могут быть назначены права на выполнение различных операций над ними: чтение, добавление, удаление, изменение, декларативная ссылочная целостность (DRI), выполнение хранимых процедур, а также права на доступ к отдельным полям. Наконец, можно вообще запретить пользователю непосредственный доступ к данным, оставив за ним лишь права на выполнение хранимых процедур, в которых будет прописан весь сценарий его доступа к базе.

***MS SQL Server в Internet/Intranet-приложениях.*** Времена статических страниц объявлений и рекламы миновали - бурное развитие бизнеса в Internet предполагает непосредственное участие клиента в совершении сделок. Говоря об использовании MS SQL Server при построении активных Internet/intranet-приложений, мы снова должны обратиться к преимуществам его тесной интеграции со всеми продуктами семейства Microsoft BackOffice. На этот раз речь пойдет об Internet Information Server (IIS).

Помимо исполнения CGI-скриптов MS IIS предоставляет разработчикам возможность создания с помощью соответствующего прикладного программного интерфейса (ISAPI) приложений в виде динамических библиотек, запуск которых происходит в ответ на команду или выбор линка на Web-странице. В отличие от CGI, где каждый скрипт исполняется как иной, нежели Web-сервер, процесс, что быстро "съедает" ресурсы даже достаточно мощной машины при большом количестве заходов на сервер, ISAPI-приложение выполняется в адресном пространстве Web-сервера, что, естественно, повышает скорость работы и существенно экономит машинные ресурсы. В зависимости от сложности сайта и приложений, dll могут быть предзагружены одновременно с запуском сервера, либо подгружаться/выгружаться из памяти по мере необходимости. К наиболее известным средствам разработки приложений на основе ISAPI относятся входящий в состав MS IIS Internet Database Connector (IDC), а также свободно распространяемый dbWeb.

Microsoft dbWeb представляет собой шлюз между 32-битными ODBC-ресурсами и MS IIS. dbWeb предусматривает создавание схемы, содержащей описание данных и связанных с ними Web-страниц. Он поддерживает исполнение запросов в реальном режиме времени на основе "pull"-модели публикации, позволяя тем самым создавать активные Web-страницы. Microsoft dbWeb структурно состоит из двух основных компонентов: dbWeb Service и dbWeb Administrator. dbWeb Service является типичным ISAPI-приложением, которое обрабатывает пользовательские запросы, направляемые посетителем страницы через броузер, и управляет соединениями между броузером, ODBC-ресурсом и IIS. К функциям dbWeb Administrator относится создание HTML-страниц, содержащих результаты выполнения запросов на основе уже упоминавшихся схем, с помощью которых осуществляется управление публикуемыми данными.

IDC входит в состав MS IIS. С помощью вызовов функций ODBC API он обеспечивает прямую связь между полями HTML-формы и соответствующим ODBC-достижимым источником данных. Для доступа к данным и публикации на Web IDC использует файлы двух типов - .idc и .htx. Файл с расширением idc (см. пример) содержит всю необходимую информацию о соединении с источником данных, текст запроса, а также ссылку на соответствующий htx-файл. Файл с расширением htx (см. пример) служит шаблоном страницы, на которой будут опубликованы данные из базы, а также элементы оформления в виде статического текста, графики, видео и т. п.

SQL Web Assistant, входящий в состав MS SQL Server 6.5, в отличие от двух только что рассмотренных инструментов, не является ISAPI-приложением и работает только с MS SQL Server . Web Assistant имеет интерфейс мастера (wizard), благодаря которому, администратор может сэкономить время по выполнению рутинного HTML-кодирования и получить готовую (в HTML-кодах) страницу, содержащую результаты опубликования произвольного запроса к базе. Полученная страница не является активной в строгом смысле этого слова, так как публикуется при помощи push-метода, т. е. обновление происходит по инициативе сервера и не допускает обновления со стороны клиента. Однако сервер может производить обновление (перегенерацию) страницы на триггерной основе или на основе расписаний задач под управлением SQL Executive.

Active Data Objects (ADO) в достаточно грубом приближении служат VB-интерфейсом к OLE DB. Их роль видится особенно важной в развитии компонентного подхода и технологий универсального доступа к данным. Активные серверные страницы представляют собой инструмент для эффективной разработки серверных Web-приложений, интегрирующих в своем составе HTML-код, VBScript и компоненты ActiveX. С их помощью в уже существующие наработки легко могут быть встроены фрагменты кода на VBScript или JavaScript, а также вызовы соответствующих объектов ActiveX.

MS SQL Server 6.5 представляет собой мощный полнофункциональный сервер баз данных, отличающийся высокой производительностью, быстротой освоения и удобным интерфейсом администрирования. Под его управлением могут работать базы данных в широком диапазоне от уровня среднего звена предприятия до распределенных баз масштаба корпорации. Доступ к MS SQL Server возможен из большого числа средств разработки клиентских front-end, настольных баз данных и офисных продуктов. MS SQL Server изначально ориентирован на интеграцию с другими серверами MS BackOffice, что позволяет непосредственно охватить решение комплексных задач автоматизации хранения и обработки информации, электронной почты и документооборота, построения Internet/intranet приложений и т. д. MS SQL Server работает в как в традиционных клиент-серверных платформах, так и в многоуровневых средах.

***Заключение.*** В течение 80-х годов поставщики мэйнфреймов и мини-компьютеров пытались решить задачу создания систем обработки транзакций на базе языка SQL, способных обрабатывать более 1000 транзакций в секунду.

В период с 1989 по 1992 годы по таким параметрам, как производительность и отношение стоимость/производительность, первенство принадлежало патентованным системам. В частности, лучшие показатели имели компьютеры VAX компании Digital и Cyclone/CLX компании Tandem. Наилучшая производительность компьютеров компании HP была зарегистрирована для ее продукта Allbase. В то же время линия изделий AS/400 компании IBM также имела впечатляющие параметры (существенно превосходящие соответствующие характеристики изделий серии RS/6000-AIX). Характерно, что компания IBM никогда не публиковала результаты тестирования системы DB2 на своих мэйнфреймах. Конечно, СУБД DB2 имела превосходную производительность (оценивавшуюся в сотнях транзакций в секунду), но она работала на дорогих мэйнфреймах. Вероятнее всего, IBM не хотела показывать неэкономичность своих мэйнфреймов путем публикации результатов контрольных испытаний на тесте TPC-A. Единственным поставщиком мэйнфреймов, рискнувшим опубликовать результаты тестирования, оказалась компания Unisys, система которой показала отношение стоимость/производительность примерно на уровне 45 K$/tps. В то время этот показатель вдвое превышал соответствующие показатели ее конкурентов.

В период между 1989 и 1993 годами операционные системы (SCO Unix, NetWare, NT), системы управления базами данных (Oracle, Informix, Sybase, Ingres) и мониторы транзакций (Tuxedo, VIS/TP, Encina), которые сегодня смело можно отнести к разряду продуктов массового потребления, резко улучшили свою производительность при решении задач обработки простых транзакций.

В 1993 году комбинация продуктов Unix/ Oracle/ Tuxedo стала лидером по отношению стоимость/производительность. Oracle, Tuxedo и операционная система Dynix, работающие на многопроцессорной системе компании Sequent, построенной на базе процессоров Intel 486, были первыми, преодолевшими барьер 1 Ktps, который продержался более десятилетия. Чуть позже СУБД Rdb и Oracle преодолели этот барьер с несколько лучшим отношением стоимость/производительность при выполнении тестов на шестипроцессорной системе Alpha AXP компании Digital, работающей под управлением ОС VMS. Аналогичных результатов добились и компании HP и Sun. В 1994 году лидером по отношению стоимость/производительность оказалась комбинация продуктов Compaq/SCO Unix/Oracle. Системы компаний Digital, HP и Sun имели более высокую производительность, но и более дорогие решения.

Еще несколько лет назад о компьютерах, построенных на базе платформы Intel (например ПК-совместимых системах компании Compaq), сравнивая их с системами компаний Digital, HP, IBM и Sun, можно было сказать, что в них отсутствует контроль четности в памяти или процессоре, что они используют сравнительно малонадежные диски, или что для них, например, отсутствует программное обеспечение для оперативной обработки транзакций (OLTP). Естественно, такие машины, попросту говоря, не были конкурентоспособными. Сегодня ситуация полностью изменилась. Компания Compaq является не только самым крупным в мире поставщиком серверов уровня рабочих групп, но и самым крупным поставщиком дисковых массивов уровня RAID-5. "Корпоративные" версии ее продуктов имеют мощные средства встроенной диагностики, удаленного обслуживания, интегрированные источники бесперебойного питания и даже некоторые средства резервирования узлов.

Еще два года назад наиболее отработанная технология кластеризации была ограничена операционной системой Guardian компании Tandem, системой DBC/1024 компании Teradata и операционной системой VMS компании Digital. Однако уже в то время практически каждый крупный поставщик вычислительных систем предлагал свои средства кластеризации на базе операционных систем Unix и NT.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Company** | **System** | **Thrughput (tpmC)** | **Price/Perf ($/tpmC)** | **Database Software** |
| Compaq | ProLiant 5000 6/166 4/Pentium Pro/200MHz | 6184.90 | $111 | Microsoft SQL Server 6.5 |
| Digital | AlphaServer 8400 5/350 8/DECchip21164/350MHz | 14227.25 | $269 | Oracle Rdb7 V 7.0 |
| Digital | AlphaServer 4100 5/400 4/DECchip21164/400MHz | 7985.15 | $174 | Oracle Rdb7 V 7.0 |
| Digital | AlphaServer 4100 5/400 4/DECchip21164/400MHz | 7998.63 | $152 | Sybase SQL Server 1.0 |
| HP | HP 9000 Model D370 2/PA-RISC 8000/160MHz | 5822.23 | $148 | Sybase SQL Server 1.0 |
| HP | HP 9000 Model K460 4/PA-RISC 8000/180MHz | 12321.87 | $187 | Sybase SQL Server 1.0 |
| IBM | RS6000 Power PC Server J40 8/Power PC 604/112MHz | 577.07 | $243 | Sybase SQL Server 1.0 |
| SGI | Challenge XL Server 16/R4400/250MHz | 6313.78 | $479 | Informix OnLine V.7.11.UDI |
| Sun | Ultra Enterprise 4000 12/UltraSPARC/167 MHz | 11465.93 | $189 | Sybase SQL Server v.11.0.2 |
| Sun | Ultra Enterprise 3000 6/UltraSPARC/167 MHz | 6662.47 | $152 | Sybase SQL |

Кроме того, оказалось, что новое программное обеспечение существенно проще использовать. На пример, комбинация продуктов NT/Sybase обеспечивает единообразный механизм доменов наименования и секретности, графический интерфейс для администрирования и работы, а также современные инструментальные средства разработки. Хранимые процедуры SQL, генераторы приложений типа PowerBuilder, SQLWindows, Windows 4GL и другие средства значительно упрощают построение TP-приложений клиент-сервер, поддерживающих более сотни пользователей на каждом сервере. Правда, масштабирование системы для обслуживания значительно большего числа пользователей требует разделения задачи на несколько меньших серверов или использования традиционных мониторов обработки транзакций типа Tuxedo, Encina, ACMSxp или CICS. Программное обеспечение для автоматизации построения подобного рода систем клиент-сервер может быть реализовано с помощью инструментальных средств типа Ellipse и Forte.

Таким образом, времена изменились. Полным ходом идет процесс перевода на массовую технологию систем, ранее базировавшихся на мэйнфреймах. Программное обеспечение массового потребления установило и новые точки отсчета в области ценообразования.

Системы управления базами данных (СУБД) являются одним из наиболее распространенных классов прикладных систем для серверов, выпускаемых большинством компаний-производителей компьютерной техники. Следует отметить, что приложения, ориентированные на использование баз данных, и сами СУБД сильно различаются по своей организации. Если системы на базе файловых серверов сравнительно просто разделить по типу рабочей нагрузки на два принципиально различных класса (с интенсивной обработкой атрибутов файлов и с интенсивной обработкой самих данных), то провести подобную классификацию среди приложений баз данных и СУБД просто невозможно. Стандартный язык реляционных СУБД (SQL) намного богаче, чем набор операций сетевой файловой системы.

Результаты испытаний многих систем на тестах TPC-A, TPC-B, TPC-C и TPC-D продемонстрировали, что на сегодняшний день имеются все предпосылки (необходимая производительность и соответствующее ПО) для полного переноса приложений оперативной обработки транзакций и систем поддержки принятия решений с мэйнфреймов на системы, построенные на базе микропроцессоров. При этом одной из актуальных задач является выбор аппаратно-программной платформы и конфигурации системы.

Оценка конфигурации все еще остается некоторым видом искусства, но к ней можно подойти с научных позиций. Для выполнения анализа конфигурации система должна рассматриваться как ряд соединенных друг с другом компонентов. Ограничения производительности некоторой конфигурации по любому направлению (например, в части организации дискового ввода-вывода) обычно могут быть предсказаны исходя из анализа наиболее слабых компонентов.

Поскольку современные комплексы почти всегда включают несколько работающих совместно систем, точная оценка полной конфигурации требует ее рассмотрения как на макроскопическом уровне (уровне сети), так и на микроскопическом (уровне компонентов или подсистем).

# 1.2 Исследование предметной области

В последние годы экономическая система нашей страны переживает бурное развитие. Несмотря на существующие недостатки российского законодательства, ситуация неуклонно меняется к лучшему. Прошли времена, когда можно было легко зарабатывать на спекулятивных операциях с валютой и мошенничестве. Сегодня все больше банков делает ставку на профессионализм своих сотрудников и новые технологии.

Трудно представить себе более благодатную почву для внедрения новых компьютерных технологий, чем банковская деятельность. В принципе почти все задачи, которые возникают в ходе работы банка достаточно легко поддаются автоматизации. Быстрая и бесперебойная обработка значительных потоков информации является одной из главных за­дач любой крупной финансовой организации. В соответствии с этим очевидна необходимость обладания вычислительной се­тью, позволяющей обрабатывать все возрастающие информационные потоки. Кроме того, именно банки обладают достаточными финансовыми возможностями для использования самой современной техники. Однако не следует считать, что средний банк готов тратить огромные суммы на компьютеризацию. Банк является прежде всего финансовой организацией, предназначенной для получения прибыли, поэтому затраты на модернизацию должны быть сопоставимы с предполагаемой пользой от ее проведения. В соответствии с общемировой практикой в среднем банке расходы на компьютеризацию составляют не менее 17% от общей сметы годовых расходов.

Интерес к развитию компьютеризированных банковских систем определяется не желанием извлечь сиюминутную выгоду, а, главным образом, стратегическими интересами. Как показывает практика, инвестиции в такие проекты начинают приносить прибыль лишь через определенный период времени, необходимый для обучения персонала и адаптации системы к конкретным условиям. Вкладывая средства в программное обеспечение, компьютерное и телекоммуникационное оборудование и создание базы для перехода к новым вычислительным платформам, банки, в первую очередь, стремятся к удешевлению и ускорению своей рутинной работы и победе в конкурентной борьбе.

Информатизация банка не может быть успешной и без предварительного анализа и моделирования. Даже покупка готового продукта предполагает понимание особенностей своего банка и информационных возможностей приобретаемой системы.

Создавая информационную модель банка, следует прежде всего обратить внимание на объекты (сущности) системы и их отношения, направление и характер потоков информации, которыми обмениваются эти объекты (а также на вид и характер носителей этой информации – бумажные документы, телефонные и электронные сообщения и пр.), и на операции, которые производятся над информационными потоками, порождая, поглощая и видоизменяя их.

Информационная система банка – настолько сложная и переменчивая структура, что изначально ставить задачу точного и однозначного ее описании и моделирования не только нереалистично, но и методически неверно. Необходимо с самого начала ориентироваться на создание гибкой модели в условиях частичной неопределенности. Гибкость и легкость перестраивания модели согласно ежедневно выявляющимся требованиям банковской системы – свидетельство высокого качества разработки, залог ее успешного внедрения и долгой жизни.

Современные технические средства сделали вполне реализуемой давнюю мечту специалистов о создании систем, действующих по «безбумажной» технологии. Но переход на такую технологию обработки информации не означает полного отказа от бумажных документов. Для нужд обмена с партнерами, для работы с аудиторами и контролирующими организациями, для документарной фиксации внутрибанковского оборота подготовка твердых копий, заверенных подписями ответственных лиц, остается необходимой и сейчас. Тем не менее происходит смена акцентов, и становым хребтом информационной системы становятся данные в электронной форме, а необходимая документация продуцируется как отражение электронных данных на бумажных носителях. Только в отдельных случаях (например, при введении "электронной подписи"), когда юридический статус электронного сообщения равен статусу бумажного документа, удается полностью отказаться от бумажного документооборота. Преимущества безбумажной технологии всегда были достаточно очевидны, а сегодня стали практически реализуемыми и экономически эффективными.

Применение безбумажных технологий ставит проблему оптимизации распределения информации между бумажными и электронными носителями. С этих позиций банковскую информацию можно разделить на три класса:

* данные, участвующие в расчетах (они обязательно должны быть представлены в электронной форме);
* часто используемая и поддающаяся формализации дополнительная информация (ее также целесообразно хранить в электронной форме);
* детальная и трудно формализуемая, в частности текстовая, информация (ее лучше хранить на бумажных носителях).

При одновременном хранении документов как в бумажной, так и в электронной форме существенным становится вопрос аутентичности формализованного и общеупотребительного способов представления информации в соответствующих документах. Одним из способов решения этой проблемы является последовательное выполнение правил, согласно которым всякий бумажный документ должен пройти автоматизированную регистрацию, а подготовка всех исходящих документов дня, особенно связанных с переводом средств, должна происходить только через и после внесения необходимых изменений в базы данных.

Проблема надежности информационной системы в банковском деле имеет особое значение. Неполнота или недостоверность информации, несвоевременность и ошибки при ее обработке оборачиваются не только немедленными финансовыми потерями, но и утратой доверия к банку.

***Выбор АБС.*** Самой главной задачей компьютерного департамента банка зачастую является выбор наилучшего решения из предлагаемых на рынке вариантов АБС или выбор стратегии разработки или модернизации существующей АБС. Рассмотрим критерии такого выбора.

Требования к сложной банковской системе существенно зависят от объема опера­ций, проводимых банком. Целью является создание АБС, которая обеспечи­вала бы персонал и клиентов банка необходимыми видами услуг, при условии, что расходы на создание и эксплуатацию не превышают доходов от внедре­ния АБС. Для выбора наиболее удачного решения необхо­димо учитывать:

*Стоимость АБС.* Здесь следует обратить внимание на выбор вычислительной платформы, сетевого оборудования и ПО. Немаловажна и стоимость обслуживания и сопровождения системы. Важно учитывать стандартность платформы и число независимых поставщиков оборудования и ПО. Очевидно, что конкуренция поставщиков увеличивает шансы найти более дешевое решение.

*Возможность Масштабирования.* В случае роста банка стоимость модернизации при неудачном выборе резко возрастает. Необходимо, чтобы выбранная вычислительная платформа допускала бы постепенное наращивание ресурсов в тех частях системы, где это требуется.

*Использование существующих ресурсов.* От эффективности использования уже имеющихся компьютеров, сетей и каналов связи существенно зависят и затраты на построение АБС.

*Наличие системы защиты информации.* Безопасность данных является одним из главных требований к АБС. Должна быть предусмотрена как устойчивость работы при неправильных действиях персонала, так и специализированные системы защиты от преднамеренного взлома АБС с корыстными или иными целями. На сегодняшний день безопасность АБС так важна, что мы рассмотрим этот вопрос подробнее. Система защиты и безопасности информации в АБС предполагает наличие:

1. Средства физического ограничения доступа к компьютерам АБС (идентификационные карточки, съемные блокирующие устройства и т.п.).
2. Предоставление полномочий, привилегий и прав доступа к АБС на уровне отдельного пользователя (сотрудника или клиента банка).
3. Средства централизованного обнаружения несанкциониро­ванных попыток проникнуть к ресурсам АБС, дающие возможность своевременно принять соответствующие меры.
4. Защита данных при их передаче по каналам связи (особенно актуально при использовании открытых каналов связи, например сети Internet). Здесь возможно использование "цифровой электронной подписи" и других криптографических методов.
5. Надежность системы. Отказы отдельных элементов АБС не должны приводить к ее полному выходу из строя. Кроме того, необходимо обеспечить высокую устойчивость работы АБС в условиях дестабилизи­рующих факторов (например помех в линиях связи или ошибочных действий персонала банка).
6. Наличие средств восстановления при сбоях. В АБС должны быть предусмотрены средства для прогноза, фиксации и локализации различных нештатных ситуаций и отказов оборудования (таких как: повреждений и перегрузок каналов связи; перегрузок устройств внешней памяти; нарушения целостности БД; попыток несанкциони­рованного доступа в систему и т.д.)
7. Возможность адаптации к изменениям финансового законодательства или структуры банка и другим событиям.
8. Возможность работы в режиме реального времени. В настоящее время системы типа OLTP (On-line Transaction Processing) становятся все более распространенными при создании АБС. Внедрение систем OLTP требует от банка весьма больших инвестиций, но преимущества таких систем оправдывают все затраты.

***Программные Платформы и Базы Данных.*** Среди операционных систем для АБС лидирует связка DOS + Novell (в качестве программной платформы DOS на рабочих станциях и операционная система Novell NetWare на файл-серверах) – 47,5% банков предпочитают именно такую, наиболее доступную в финансовом отношении конфигурацию. На втором месте среди операционных систем, в «угрожающей близости» к Novell NetWare находится Windows NT, ее предпочитают 43,7% банков. Поскольку третье место занимает связки операционных систем Windows/Windows'95 - 32,2%, то здесь явно заметен качественный рост влияния продуктов фирмы Microsoft на развитие банковских технологий. Всего лишь четвертое место с показателем 29,0% занимает ОС UNIX. Хотя по динамике изменений в 1994 – 1996 гг. казалось, что разрыв между вторым местом UNIX и бессменно лидирующей связкой DOS + Novell NetWare неизбежно сократится.

Собственно эта тройка (четверка) операционных систем – Novell, Windows (NT), UNIX – и составляет большинство программных платформ в российских банках, поскольку идущие на пятом-шестом местах OS/400 (на аппаратных средствах AS/400) и ОS/2 занимают незначительные доли рынка: 4,9 и 3,3% соответственно. На последнем, седьмом месте находится VAX/VMS с минимальной долей – 0,5%.

Соответственно распределению предпочтений по ОС среди СУБД лидирует «родной» для Novell NetWare менеджер записей Btrieve – 42,6% банков предпочитают именно эти технологии. Второе место с небольшим отрывом занимает профессиональная СУБД Oracle – 35,5%.

Два явных лидера среди СУБД Btrieve и Oracle – опережают в три-четыре раза идущую на третьем месте Sybase (11,5%) и в пять-шесть раз занимающую четвертое место Informix (7,7%).

Эти соотношения позволяют сделать грубую оценку возможного перехода банков пользователей технологий на основе Btrieve на банковские системы нового поколения на основе СУБД Sybase без замены фирмы-разработчика. Так, «новые Sybase-АБС» подготавливаются к промышленному внедрению в основном тремя фирмами – лидерами рынка: «Диасофт», «R-Style Software Lab.» и «Кворум», чьи базовые системы сегодня реализованы на основе Btrieve. Соотношение 42,6% против 11,5% говорит о том, что примерно одна пятая часть банков – пользователей Btrieve-AБC в перспективе может перейти на Sybase-АБС.

Особого комментария заслуживает пятое место Microsoft SQL Server с небольшим показателем 4,9%. Огромная популярность операционной среды Windows в сфере банковских технологий, казалось бы, могла обеспечить этой «дочерней СУБД» более высокие цифры. Но здесь важно следующее: сам инструментарий MS SQL так интенсивно модернизируется и меняется, что серьезные финансовые системы типа АБС, требующие высокой надежности и сделанные на его основе, просто не успевают пройти необходимый цикл тестирования, отладки и обкатки. Конечно же, это существенно сдерживает возможности распространения финансовых приложений на основе Microsoft SQL Server.

Шестое место занимает СУБД Progress – ее предпочитают 3,8% банков, далее с малыми значениями идут Interbase – 2,2%, Gupta/Centura – 1,6%, DB/2 – 1,1%, Ingress – 0,5%.

Примерно один из девяти банков предпочитает работать сразу на нескольких СУБД. Если Windows NT и Windows'95 считать как одну и ту же ОС, то выходит, что примерно каждый третий российский банк работает в гетерогенных программных средах.

***Известность фирм-разработчиков.*** Расчет рейтинговой известности фирм давно известен и неоднократно публиковался. И хотя его методика не совсем совершенна, только ее постоянство и неизменность позволяют корректно сравнивать показатели, полученные в разное время.

Основой расчета рейтинга известности и других сравнительных показателей являются ответы банков на вопросы анкеты об используемых программных продуктах, планах по модернизации АБС и мнения отдельных банкиров о состоянии и развитии рынка банковских программных технологий.

*(по материалам третьего форума разработчиков)*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Разработ­чики АБС** | **Исполь­зуют** | **Извест­ность** | **Лучшая** | **Пер­спек­тивная** | **Хотели бы узнать** | **Хотят купить** | **Рейтинг Извест­ности** |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| R-Style Soft. Lab. | 37 | 141 | 37 | 22 | 19 | 14 | 380 |
| Диасофт | 22 | 153 | 31 | 13 | 11 | 12 | 320 |
| ФОРС | 6 | 58 | 21 | 19 | 10 | 5 | 170 |
| ПрограмБанк | 13 | 69 | 4 | 4 | 2 | 2 | 117 |
| ЦФТ | 10 | 37 | 9 | 9 | 5 | 7 | 112 |
| Кворум | 13 | 37 | 6 | 3 | 1 | 5 | 92 |
| Инверсия | 9 | 38 | 3 | 2 | 0 | 1 | 68 |
| Асофт | 4 | 19 | 2 | 0 | 0 | 0 | 31 |
| МИМ-Технология | 6 | 3 | 3 | 1 | 1 | 1 | 26 |
| Midas-Kapiti | 3 | 7 | 3 | 1 | 4 | 0 | 25 |
| CSBI EE | 2 | 9 | 1 | 1 | 2 | 1 | 21 |
| Temenos Systems | 0 | 1 | 1 | 3 | 3 | 0 | 12 |
| Мебиус | 4 | 1 | 0 | 0 | 0 | 0 | 9 |
| Канопус | 2 | 2 | 1 | 0 | 0 | 0 | 8 |
| Киевский ОДБ | 4 | 0 | 0 | 0 | 0 | 0 | 8 |
| БИС | 1 | 4 | 0 | 0 | 0 | 0 | 6 |
| Сибирский Банк | 0 | 1 | 1 | 0 | 0 | 1 | 5 |
| UniSAB | 1 | 1 | 1 | 0 | 0 | 0 | 5 |

*Примечания:*

*Рейтинг известности = “2” + “5” + 2\*(“1” + “3” + “4” + “6”);*

Заметно первое изменение на Олимпе банковской автоматизации; поколеблено единоличное лидерство фирмы «Диасофт», длившееся с начала 1994 г. до первой половины 1997 г. включительно (завидное постоянство!). Теперь она занимает второе место. Первую строчку с отрывом в 15,8% от второго места (380 баллов против 320) заняла «R-Style Software Lab.». Каждая из этих лидирующих фирм почти вдвое опережает идущую на третьем месте фирму ФОРС (170 баллов) и почти втрое следующих за ними – фирму «ПрограмБанк» (117 баллов – 4 место) и ЦФТ (112 баллов – 5 место).

Еще три фирмы – «МИМ-Технология», «Канопус», БИС – занимают неплохие места; девятое, четырнадцатое и шестнадцатое соответственно. Всего банкиры при ответах на вопросы называли 41 фирму.

Понятно, что абсолютная известность определяется как маркетинговой активностью фирмы, так и распространенностью и перспективностью ее программных технологий. Для характеристики потенциала фирм к расширению бизнеса, был введен дополнительный показатель – «Относительный рейтинг известности», показывающий усредненные (в пересчете на один банк) показатели известности фирмы.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Разработ­чики АБС** | **Исполь­зуют** | **Извест­ность** | **Лучшая** | **Пер­спек­тивная** | **Хотели бы узнать** | **Хотят купить** | **Рейтинг Извест­ности** |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| ФОРС | 100% | 9,67 | 3,50 | 3,17 | 1,67 | 0,83 | 18,83 |
| Диасофт | 100% | 6,95 | 1,41 | 0,59 | 0,50 | 0,55 | 10,00 |
| ЦФТ | 100% | 3,70 | 0,90 | 0,90 | 0,50 | 0,70 | 6,70 |
| R-Style | 100% | 3,81 | 1,00 | 0,59 | 0,51 | 0,38 | 6,30 |
| ПрограмБанк | 100% | 5,31 | 0,31 | 0,31 | 0,15 | 0,15 | 6,23 |
| Инверсия | 100% | 4,22 | 0,33 | 0,22 | 0,00 | 0,11 | 4,89 |
| Кворум | 100% | 2,85 | 0,46 | 0,23 | 0,08 | 0,38 | 4,00 |
| МИМ | 100% | 0,5 | 0,5 | 0,17 | 0,17 | 0,17 | 1,50 |

В относительном рейтинге картина значительно изменилась – на первое место с большим отрывом от других вышла фирма ФОРС (18,83 балла). Вторую строчку очень уверенно занимает «Диасофт» (10,00), далее идет весьма плотная по показателям группа – ЦФТ (6,70), «R-Style Software Lab.» (6,30) и «ПрограмБанк» (6,23).

***Привлекательность АБС.*** Такой показатель, как «коэффициент привлекательности» рассчитывается по результатам опроса банков – пользователей каждой фирмы. При расчете учитывались ответы банков на вопросы: «Удовлетворяет ли Ваш банк используемая автоматизированная банковская система?» и «Собирается ли Ваш банк и ближайшее время менять или модернизировать используемую АБС (отметьте и при необходимости расшифруйте нужные строки)?» – с вариантами ответов. В расчетах также учитывались значения графы “6” табл. 1 (число банков, собирающихся перейти на АВС каждой из фирм).

Поскольку разные представители одного и того же банка могли отвечать на один и тот же вопрос по разному, а в одной анкете респондент отметил даже несколько позиций, то для устранения некорректности вычислений по каждой фирме были рассчитаны максимальное и минимальное значения коэффициента.

При расчете максимального коэффициента все «многозначности» в ответах представителей одного банка трактовались «в пользу» фирмы-разработчика, а при расчете минимального – «против».

Вычисление такого диапазона значений, пожалуй, более корректно, чем вычисление методической погрешности по одному Банку. Поскольку величина расчетной погрешности по большинству показателей меньше, чем разница минимального и максимального значений, введение диапазона значений позволяет более корректно учитывать многозначные ответы.

Смысл этого коэффициента относительно несложен. Если все пользователи какой-либо АБС вполне удовлетворены этой системой, собираются обновлять версии или ни чего не собираются в ней менять, то эта АБС еще будет жить на рынке и ее «коэффициент привлекательности» равен единице. Если же какие-либо банки собираются переходить на эту ЛБС, то у нее есть перспективы дополнительного развития, и ее «коэффициент привлекательности» выше единицы. А если все пользователи отрицательно оценивают удовлетворенность системой и собираются закупать другую российскую или зарубежную разработку, то данная АБС морально устаревает и ее «коэффициент привлекательности» близок к минус единице.

Значения коэффициента, близкие к плюс единице, можно считать признаком поступательного развития как фирмы в целом, так и ее банковских разработок в частности. Любые положительные значения «коэффициента привлекательности» считаются вполне приемлемыми, а отрицательные – служат сигналом неблагополучия. Близкие к нулю значения позволяют говорить о некотором снижении числа банков-пользователей.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Разработчики | Опрошено | Удовлетворяет ли АБС? | | | | | Закупка/модернизация АБС | | | | | | | Привлека­тельность | |
| Да | Скорее Да | Не пол­ностью | Скорее нет | Нет | Нет | Новую версию | Другую рос. АБС | Зарубеж. АБС | Доработка модулей | Доработка АБС | Купить эту АБС | мах | min |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Форс | 6 | 1 | 3 | 3 | 0 | 3 | 1 | 4 | 0 | 0 | 3 | 0 | 5 | 1,40 | 1,20 |
| ЦФТ | 10 | 1 | 5 | 3 | 1 | 1 | 2 | 6 | 1 | 1 | 2 | 0 | 7 | 1,18 | 0,97 |
| R-Style | 37 | 2 | 17 | 16 | 1 | 1 | 3 | 34 | 2 | 2 | 11 | 1 | 14 | 1,00 | 0,86 |
| Midas-Kapiti | 3 | 2 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 0 | 0 | 1,00 | 0,88 |
| Кворум | 13 | 0 | 8 | 4 | 2 | 0 | 5 | 6 | 0 | 0 | 3 | 0 | 5 | 0,99 | 0,90 |
| Диасофт | 22 | 2 | 8 | 9 | 2 | 1 | 6 | 13 | 3 | 1 | 7 | 2 | 12 | 0,98 | 0,82 |
| МИМ | 6 | 1 | 3 | 2 | 0 | 0 | 0 | 4 | 0 | 0 | 1 | 0 | 1 | 0,90 | 0,90 |
| Мебиус | 4 | 0 | 1 | 3 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 0,63 | 0,63 |
| ПрограмБанк | 13 | 0 | 5 | 5 | 2 | 2 | 4 | 7 | 3 | 0 | 1 | 0 | 2 | 0,50 | 0,32 |
| Инверсия | 9 | 0 | 3 | 4 | 1 | 1 | 0 | 6 | 5 | 0 | 2 | 0 | 1 | 0,25 | -0,01 |
| АСофт | 4 | 0 | 1 | 2 | 0 | 1 | 0 | 3 | 2 | 1 | 0 | 0 | 0 | 0,13 | -0,13 |
| Собственная | 59 | 5 | 10 | 30 | 8 | 6 | 7 | 8 | 15 | 1 | 19 | 21 | 0 | - | - |
| Киевский ОДБ | 4 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 3 | 0 | 0 | 1 | 0 | -0,91 | -0,91 |

В первую очередь обращает внимание высокая плотность семи верхних результатов. Формально третье место «R-Style Software Lab.» и формально шестое место фирмы «Диасофт» разделяют всего 0,02 по максимальному значению и 0,04 – по минимальному; при этом пересечение диапазонов для этих фирм – 0,12. Кроме того, заметим, что среднее по всем фирмам значение «коэффициента привлекательности» весьма велико. Следовательно, можно утверждать, что банки предпочитают в тяжелые времена технологического перехода к новым правилам бухгалтерского учета решать возникающие сложные проблемы со своими разработчиками, а не заменять АБС на «самую мощную и полно функциональную».

***Стратегии развития АБС.*** Семилетняя история автоматизации российских банков позволяет узнать: есть ли общие черты в путях автоматизации российских банков. При описании базовых стратегий технологического развития банков (в части программных решений) удобно использовать классификацию, предложенную В. Колсаноным (в прошлом – начальник управления банковских технологий Тверьуниверсалбанка). Любая стратегия технологического развития банка подразумевает формулирование функциональных и экономических требований (с различной степенью детальности) к системе автоматизации и соответствующую реализацию этих требовании.

Собственная разработка. Этот этап в развитии автоматизированных технологий в разное время был пройден подавляющим большинством российских банков, поэтому ему стоит уделить внимание. В этот же сегмент включены так называемые «эксклюзивные» разработки, выполненные под заказ одного банка и не поддающиеся тиражированию. Основные признаки подобной стратегии:

* первыми автоматизируются исторически «базовые» функции банковской системы – бухгалтерия и расчетно-кассовое обслуживание, существующие по крайней мере в двух-трех десятках тиражируемых российских систем;
* не разделены служба разработки и служба эксплуатации банковской системы, не выделены моменты передачи версий банковской системы в эксплуатацию, и, как правило, понятие «версия АБС» вообще не приемлемо;
* неполноценное документирование системы (частичное наличие проектной и эксплутационной документации) либо полное отсутствие документации.
* С точки зрения специалистов, имеющих опыт создания и внедрения АБС в крупных и средних банках, разумными основаниями для собственной разработки могут являться:
* интеграция нескольких систем при числе автоматизированных рабочих мест в банке более 200;
* разработка целых модулей системы при числе автоматизированных рабочих мест в банке более 500;
* наличие у банка уникальных бизнес требований и уникальной стратегии информатизации, удовлетворить которые не в состоянии ни специализированные системные интеграторы, ни разработчики промышленных тиражируемых решений. В этом случае требования и стратегия должны быть явно сформулированы во внутренних документах банка с соответствующим контролем (аудитом) их реализации;
* столь динамичное развитие банка, что бизнес требования не могут быть выдвинуты даже на год вперед. В этом случае разрабатываемая система является только макетом будущей информационной системы. Сегодня это уже в прошлом: наступает этап стабильного развития банковской системы.

С точки зрения специалистов, имеющих опыт профессионального банковского консалтинга, помимо приведенных причин, собственная банковская разработка может быть оправдана:

* для небольшого по количеству персонала одно-филиального банка с небольшим количеством ежедневных операций – «домашнего» банка крупной промышленной либо торговой компании, консорциума, холдинга (примеров тому довольно много);
* как связующее звено для автоматизации головного офиса многофилиального банка при установке однотипных тиражируемых решений в филиалах и отделениях;
* для узкоспециализированного банка со значительным преобладанием операций, не характерных для банковского сообщества в целом;
* для крупного универсального банка в части автоматизации передовых финансовых технологий, пока еще недостаточно освоенных российским банковским рынком. При отсутствии адекватного предложения на рынке тиражируемых решений такая разработка бывает оправданной в части указанных финансовых технологий вплоть до их сопряжения с основной AБC. Эта причина становится все менее определяющей, поскольку промышленные решения ведущих российских разработчиков быстро развиваются в функциональном отношении;
* для среднего банка, имеющего целью стать «законодателем технологической моды» в области автоматизации финансовых операций при сильной динамике развития банковского рынка в целом (например, Инком банк, Кредо банк, Сибирский Торговый банк, Оптимум-банк). Однако и эта причина осталась в прошлом как из-за нынешней стабилизации банковского рынка, так и из-за наличия большого числа конкурентоспособных предложении. Заметим также, что правильнее изначально ставить задачу создания «отчуждаемых технологий», а не «технологий для себя».

Основным недостатком собственной разработки является сложность интеграции консолидированных технологий в получаемые проектные или программные решения. Кроме того, обычно собственные разработки характеризуются низким (в большинстве случаев) качеством закладываемых проектных проработок, которые позволяют получить решение проблемы лишь «на сегодня», но не дают возможности оперативно отслеживать изменение внешних условий в будущем (отчетность, развитие финансового рынка, введение новых банковских операций, развитие информационных технологий вообще).

Еще одна опасность заключена в неоправданном завышении приоритетов системных требований (аппаратное обеспечение, операционная среда, инструментарий разработки, коммуникации) перед бизнес-требованиями (функциональность, надежность, безопасность, поддержание и развитие технологий).

Одним из негативных последствий этого преобладания является то, что в банке могут более или менее успешно решаться только текущие проблемы автоматизации типа «не проводится документ», «потерян клиентский платеж», «требуется более мощный компьютер», «нет связи с филиалом/отделением». Кроме того, хотя все пользователи знают, что в далеком будущем банк будет работать на «самой лучшей» программно-аппаратной платформе типа Sun + Oracle или Hewlett-Packard + Informix, никто не знает, как будет развиваться автоматизированная система в среднесрочной перспективе и какие финансовые технологии «завтрашнего дня» можно использовать «уже сегодня», хотя бы в ограниченном объеме.

Замеченная еще в 1995 – 1996 гг. тенденция к снижению относительного качества и количества собственных разработок сегодня получает дальнейшее развитие. В 1997 г. существенно сузится тот сегмент технологий банковской автоматизации, где представлены собственные разработки. В дальнейшем, на рынке останутся лишь те, которые выполнены по законам классической программной инженерии: *Техническое задание Технический проект  Информационная модель  Прототип АБС  Система комплексной автоматизации.*

Корпоративно-субъективная. Цель такой стратегии состоит в реализации любых требований всех (или почти всех) банковских пользователей. Финансирование стратегии ведется в зависимости от возможностей банка и искусства руководителя управления (департамента, отдела) автоматизации, роль которого весьма велика. Стратегия характерна главным образом для малых банков, а для средних и крупных она оказывается крайне опасной. Банкам не всегда удается удержаться в рамках подобной стратегии, особенно при неравномерном развитии и при смене приоритетов в банковском бизнесе. Поскольку требования по переходу на новую систему учета в конце 1997 г. были «самыми главными», то для небольших банков с подобной автоматизацией приемлемыми были решения по обновлению версии или по покупке наиболее дешевых из «двадцатиразрядных АБС».

Партнерская. В рамках этой стратегии обычно выбирается стратегический поставщик информационных технологий, и вся работа ведется исключительно (или главным образом) с ним. Возможен вариант, при котором банк и фирма являются взаимными акционерами или учредителями друг друга. Как правило, поставщик является производителем интегрированных программных решений класса АБС. Разновидностью этой стратегии стало использование конкретной промышленной АБС во всех филиалах крупной «банковской империи». В целом это достаточно дорогой способ, но эффективный при следующих условиях:

фирма-партнер профессионально специализируется в области банковских информационных технологий и имеет опыт разработки и внедрения высокотехнологичных отчуждаемых программно-аппаратных решении;

при действительно партнерских отношениях фирма-партнер должна быть тесно интегрирована с банком (желательное условие) и иметь успешный опыт работы с банками такой же «весовой категории» (обязательное условие). Разумеется, что банк и фирма-партнер должны территориально находиться в одном городе.

Фирма может быть как частично интегрирована в структуру банка, так и полностью автономна. Тесная интеграция партнеров подразумевает следование разработчиков принятым в банке технологическим решениям. Автономный вариант подразумевает интеграцию технологий банков, поэтому у каждой фирмы (Диасофт, R-Style Software Lab., ПрограмБанк, ФОРС) может быть несколько банков-партнеров.

Правильный выбор фирмы-партнера дает банку весомую гарантию поступательного развития в течение нескольких лет. Сегодня в банковскую практику вошло партнерство не только на этапе эксплуатации АБС, но и на этапах проектирования и разработки АБС. Примером тому служит новый проект новосибирского Центра Финансовых Технологий.

Если банк не ошибся ранее в выборе партнера, то он находится н наиболее благоприятном положении: именно под его запросы и пожелания и будет разрабатываться (точнее, уже разработана) АБС под новые правила учета.

Престижная. Стратегия подразумевает совершенное решение любых возникающих проблем. Самый дорогой и не всегда самый эффективный с точки зрения использования вложений способ. Для его реализации в качестве партнеров выбираются крупнейшие в мире фирмы-консультанты («Маккинзи», «Эрнст энд Янг», «Прайс Уотерхаузэ», «Артур Андерсен» и т.п.) и/или крупнейшие поставщики высококачественного оборудования и системные интеграторы (IBM, Unisys, DЕС, Sun и т.п.), которые занимаются всеми проблемами в сфере информационных технологий, в том числе и банковскими решениями.

Этот способ предпочтителен для банков, чья активность сосредоточена исключительно на внешних рынках, поскольку скорость адаптации получаемых решений к изменениям российского банковского рынка и законодательства очень низка.

Банки с престижной автоматизацией временно оказываются в наиболее тяжелом положении, поскольку для них вопрос возможной замены системы учета может привести либо к омертвлению ранее сделанных инвестиций, либо к неоправданно высоким затратам для решения срочных проблем.

Экономичная. Подобная стратегия подразумевает самое дешевое решение периодически возникающих проблем. Это наилучший способ на начальном периоде развития банка, поскольку совмещает и бизнес требования, и экономические требования. Если требования банка хотя бы как-то формализованы, эта стратегия может оказаться действенной в течение нескольких лет.

Однако примерно каждые 2 – 3 года накапливаются несовместимые между собой решения, требующие почти полного технологического перевооружения. Подавляющее большинство малых российских банков и некоторое число средних и крупных идут именно по этому пути, эффективному на короткое время, но проигрышному в перспективном плане.

Для банков с экономичной автоматизацией переход на новый план счетов может стать как раз той точкой, когда можно относительно легко перейти к партнерскому способу автоматизации либо со штатным поставщиком программных технологий, либо с другой фирмой, имеющей более высокую надежность в качестве разработчика комплексных АБС.

Научная. Данный вариант стратегии включает глубокую формализацию бизнес требований банка и проведение тендера между поставщиками (производителями) технологических решений. Как уже отмечалось, к организации тендера целесообразно привлечение профессиональных консалтинговых фирм. Вариант тендера наиболее подходит в период организационной стабилизации банка, позволявший учесть его специфику, а также оценить имеющиеся на рынке предложения не только по технологическим решениям, но и по вариантам взаимодействия с поставщиком (разработчиком) этих решений в перспективе на 5 – 7 лет.

Этот способ, как и предыдущий, совмещает требования банка, но на более глубоком уровне. Здесь самым важным является этап формализованного описания требований банка.

Проведение тендера на разработку/внедрение АБС приводит к задержке начала работ по автоматизации примерно на 3 – 6 месяцев, подает лучшие результаты в перспективе, поскольку позволяет заранее оценить и сравнить возможные решения по их стоимости, эффективности, срокам реализации и технологии взаимодействия с разработчиком.

Реализация научного способа с очень высокой вероятностью приводит банк к партнерской автоматизации.

Возможен и так называемый «парадоксальный» вариант, при котором банком планомерно ведутся работы по тендеру (выбору стратегического партнера) и одновременно закупается и внедряется упрощенное временное решение для перехода на новую систему учета.

Если попытаться охарактеризовать состояние рынка банковской автоматизации во второй половине 1997 года одной-двумя фразами, то получится примерно так: горизонтальный рынок автоматизированных банковских систем уже пришел в состояние жесткой нестабильности: выигрывает не тот, кто предлагает наилучшее решение, а тот, чью неидеальную систему можно быстро внедрить. Важно, что бы фирмы-разработчики должным образом инвестировали получаемые средства для нормальной работы в нынешнем году, когда финансовая емкость рынка АБС значительно падает. Одновременно с этим вертикальная составляющая этого рынка продолжает динамичное развитие все большему числу банков – необходимы все более сложные программно-технологические решения. И именно вертикальный рынок сложных решений для «технологически продвинутых» банков станет определяющим в 1998 г. и в дальнейшем при приближении российской банковской системы к мировым стандартам.

На текущий момент региональный рынок представлен довольно большим количеством систем ведения бухгалтерского учета: Фолио (АО «Центр экономических компьютерных программ ФОЛИО»), ИНФО-Бухгалтер (ТОО «Информатик»), ИНФИН-бухгалтерия (Аудиторская компания «ИНФИН»), «СуперМенджер» (Фирма «Ланкс»), AUBI (Фирма «О'стрим»), ABACUS (АО «ОМЕГА»), 1С Бухгалтерия и т.д.

Система СуперМенджер, предназначенная для автоматизации бухгалтерского учета на предприятиях сложной структуры различных форм собственности. Система позволяет оперировать операциями: аналитического и синтетического учета; автоматического учета курсовой разницы; приведения учетных данных к любой национальной валюте; ведения журналов-ордеров, главной книги и баланса в любой валюте и сводно по эквиваленту; формирования сложных проводок; консолидации данных различных организаций и филиалов.

СуперМенджер работает в различных компьютерных сетях и на компьютерах IBM и Macintosh. Существуют версии программ для DOS, Windows и Macintosh.

Программа ИНФО-Бухгалтер в любой момент выдает баланс со всеми приложениями, оборотная ведомость, главная книга, ведомости аналитического учета по счетам, журналы ордера и ведомости к ним, шахматка, разнообразные ведомости и справки, анализ финансовой деятельности с построением графиков и диаграмм.

Программа ФОЛИО позволяет вести: бухгалтерский учет любого числа предприятий на одном компьютере с возможностью получения сводного бланка нескольких предприятий; подробный финансовый анализ деятельности организаций, по которым ведется бухгалтерия; учет движения денежных средств в динамике; финансовый баланс для руководителя и отчет о прибыли и убытках по месяцам и годам; различные аналитические показатели; возможность генерации новых форм отчетности.

Продуманная структура программы ИНФИН-Бухгалтерия и привычный бухгалтеру дизайн позволяет: полная автоматизация учета; до пяти уровней аналитического учета; минимальные изменения в настройке программы под специфику предприятия; бухучет для нескольких предприятий на одном рабочем месте; возможность настройки на любое изменение законодательства; возможность ведения двойной бухгалтерии; возможность работы с любыми валютами.

ABACUS Professional – полный комплекс бухгалтерского учета. Отличительные особенности комплекса – это функциональная полнота и комплексное решение всех задач учета: обработка проводок с детальной аналитической информацией; учет затрат на производство и расчет себестоимости продукции с формированием соответствующих записей в Главной книге; элементы финансового анализа; автоматическое начисление процентов и отчисление налогов; мультивалютные операции; генератор отчетных форм; система аппаратной и программной защиты информации; удобный интерфейс.

1C бухгалтерия предоставляет возможность ручного и автоматического ввода проводок. Все проводки заносятся в журнал операций. Кроме журнала операций программа поддерживает несколько списков справочной информации (план счетов, список видов объектов аналитического учета, списки объектов аналитического учета, констант и т.д.).

В программе существует режим формирования произвольных отчетов, позволяющий описать форму и содержание отчета, включая в него остатки и обороты по счетам и по объектам аналитического учета. С помощью данного режима реализованы отчеты, предоставляемые в налоговые органы, кроме того, данный режим используется для создания внутренних отчетов для анализа финансовой деятельности организации в произвольной форме. Кроме того, программа имеет функции сохранения резервной копии информации и режим сохранения в архиве текстовых документов.

“АУБИ” - это зарегистрированное название интегрированной программной системы «Автоматизации Бухгалтерского Учета» малых, средних и больших предприятий.

“АУБИ” может быть с успехом использована для автоматизации бухгалтерского учета предприятий различного рода деятельности. Программный комплекс представляет одинаковый интерес как для торговых (коммерческих) структур, так и для производственных предприятий. Гибкая система программы позволяет настраивать “АУБИ” на нужды конкретного пользователя. При этом бухгалтер каждого предприятия, исходя из своих собственных потребностей, имеет возможность сформировать план счетов; информационные справочники, содержащие названия предприятий-партнеров и их банковские реквизиты; список материально ответственных лиц и т.д. В зависимости от специфики деятельности предприятия “АУБИ” позволяет вести учет следующих элементов бухгалтерского производства: учет материалов (склад); учет малоценных и быстроизнашивающихся материалов (МБП) на складе и в эксплуатации; основные средства; учет кассовых операций - формирование приходных и расходных кассовых ордеров, ведение кассовой книги; учет банковских операций - платежных поручений, требований и реестров; учет счетов; ведение журнала хозяйственных операций; ведение главной бухгалтерской книги; формирование шахматной и оборотной ведомостей; формирование различных ведомостей аналитического учета и т.д.

# 2. Практическая часть

Разрабатываемая программа предназначена для ведения учета выданных и полученных счетов-фактур предприятия. Программа должна обеспечивать защиту данных от несанкционированного доступа, обладать доступным и понятным интерфейсом, по функциональным качествам не уступать старой программе учета, включая в себя все возможности. Программа должна работать как автономно, так и как модуль общей программы бухгалтерского учета.

На сегодняшний день, спектр отдельных программ по области учета выданных и полученных счетов-фактур предприятия ничтожно мал. В основном это модули больших бухгалтерских программ.

## 2.1 Анализ существующей программы

Существующая программа «Книга покупок» фирмы «ИНФИН» работает под управлением операционной системой MS-DOS. При этом она «вешает» машину при попытке запуска из-под Windows’95, поэтому, для работы с программой приходится перезагружать компьютер в режиме командной строки. Тот факт, что она написана под DOS, уже свидетельствует о неудобном интерфейсе пользователя. Отсутствие поддержки мышки, сложность, запутанность и непонятность назначения некоторых диалоговых окон, отсутствие системы помощи (не говоря гибкой системы контекстной подсказки), неудобство ввода информации и многое другое еще меньше привлекает к программе.

Система управления базой данных программы фирмы «ИНФИН» построена на технологии клиент-сервер. При этом программа может работать как с локальной, так и с сетевой базой данных. Заметим, что при отсутствии доступа к сетевой базе, программа автоматически переключается на локальную базу, не выдавая при этом никаких предупреждений и сообщений. Еще несколько лет назад, среди СУБД наибольшей популярностью пользовались СУБД dBase, Paradox, Rbase, получившие общее название Xbase (созданных на технологии файл-сервер), а в качестве инструментальных средств самыми распространенными были Clipper и FoxPro. Сейчас на рынке этих СУБД распространенны Access, FoxPro, Paradox, dBase. При технологии файл-сервер БД хранится на сервере, а СУБД - на клиентской станции, поэтому клиентская станция должна быть достаточно мощной для обработки полученных данных с сервера и проведения необходимых манипуляций с данными. При обращении к одной записи базы данных считываются целиком все необходимые для этого таблицы, что повышает трафик сети, увеличивает время обработки. В результате получается, что работа ведется с локальной базой данных. Но самый главный недостаток таких СУБД, это то, что только данная конкретная программа способна правильно производить изменения в БД, сохраняя их целостность. Любое стороннее вмешательство в базу данных может привести к полному разрушению данных и потере всей информации.

Сама структура базы данных совсем не правильная, с точки зрения теории построения баз данных, и очень не удобна. Для каждого года и месяца создаются собственные поддиректории с полной базой данных (кроме справочника клиентов) за соответствующий период.

Такой подход очень не удобен, так как при неправильном вводе (например, забыли «перевести» месяц или год) приходится все удалять и вводить заново.

Структура таблиц, которые копируются в директории месяцев, несет в себе много лишней информации, которая только занимает место на диске. Создание таблиц отдельно для полученных, отдельно для выданных счетов-фактур, не только не необходимо, а даже просто не нужно. А по теории нормализации база данных не находится даже в первой нормальной форме (по данной теории, база данных считается хорошей, если она находится в третьей расширенной нормальной форме). Создание отдельных полей для разных сумм просто не логично.



Вообще подход с разделенной базой данных по месяцам резко сужает возможности программы для создания отчетов (период отчета не более чем месяц), возможности экспорта/импорта данных в базу и многое другое.

Программа «Книга покупок» фирмы «ИНФИН» поставляется с ограниченным количеством копий. Но при этом возможности переноса программы на другую машину нет. Все это еще более негативно отражается на отношении пользователей к программе.

Для перевода программы на современную технологию клиент-сервер, необходимо почти полностью переработать базу данных (учитывая все положительные и отрицательные стороны старой программы) для архитектуры клиент-сервер. Необходимо создать удобный пользовательский интерфейс под операционную систему Windows’95 и Windows NT. Предусмотреть гибкую систему помощи, подсказок и отчетности. Также предусмотреть возможность экспорта/импорта данных.

## 2.2 Выбор платформы и программных средств

Сейчас на российском рынке сетевых операционных систем наиболее популярны такие, как Microsoft Windows NT, Novell NetWare, IBM OS/2 Warp и различные версии UNIX. Приведем основные сравнительные характеристики операционных систем Novell NetWare 4.1, Microsoft Windows NT Server 4.0 и Unix. Не будут отражены некоторые известные продукты, такие как IBM OS/2 Warp Server и Banyan VINES. Очень возможно, что это добротные операционные системы, однако если поддержку и сопровождение даже Windows NT и NetWare в России можно назвать весьма слабой, то для OS/2 и VINES она, по существу, отсутствует. А серьезные заказчики никогда не будут приобретать продукты, в поддержке которых они не уверены. Все ведущие поставщики Unix-систем поставляют в качестве дополнительных модулей, а порой и интегрировано в базовом комплекте, такие службы, как NFS (Network File System - сетевая файловая система), NIS (Network Information Service - сетевая информационная служба), X Window System и множество других. Именно они делают Unix полноценной сетевой операционной системой, по функциональности мало, чем уступающей другим. Все приводимые характеристики полностью соответствуют широко распространенным версиям Unix (SCO OpenServer и UnixWare, SunSoft Solaris и Interactive Unix, Hewlett-Packard HP-UX, IBM AIX, Digital Unix, SGI IRIX и др.).

Компания Novell была одной из первых компаний, которые начали создавать ЛВС. Она производила как аппаратные средства, так и программные, однако в последнее время фирма сконцентрировала усилия на программных средствах ЛВС. Операционная система NetWare способна поддерживать рабочие станции, управляемые DOS, OS/2, UNIX, Windows NT и Windows’95, Mac System 7 и другими ОС. Она может надежно работать с большим количеством различных типов сетевых адаптеров и протоколов. Фирма Novell имеет контракты о поддержке NetWare с наиболее крупными и мощными из независимых организаций, таких как Bell Atlantic, DEC, Hewlett-Packard, Intel, Prime, Unisys и Xerox.

Версия ОС NetWare 2.2 может работать на компьютере 80286 (или более поздних моделях), используемом в качестве файлового сервера. Версии NetWare 3.12 и 4.0 ориентированы на 32 разрядные шинные архитектуры и процессоры 80386, 80486 и выше. Существуют версии NetWare, предназначенные для работы под управлением многозадачных, многопользовательских операционных систем OS/2 и UNIX. NetWare 3.12 имеет возможность поддержки до 250 пользователей, а версия 4.0 – до 1000 пользователей. Все версии хорошо совместимы между собой, поэтому в одной и той же компьютерной сети могут находиться файловые серверы с разными версиями ОС NetWare.

Операционная система Advanced NetWare 2.0 была выпущена в 1986 году. Одной из выдающихся особенностей Advanced NetWare 2.0 была способность соединять до четырех различных сетей с одним файловым сервером.

В 1987 году вышла операционная система NetWare SFT, которая отличалась от предыдущей версии повышенной отказоустойчивостью и сохранностью данных.

NetWare 2.15 и NetWare для Macintosh дебютировали в 1988 году. Существенным недостатком этих версий было очень большое время инсталляции - оно включало в себя время тестирования жесткого диска и могло продолжаться день или даже два.

32-разрядная сетевая ОС NetWare 386 была выпущена в сентябре 1989 года. В ней была значительно улучшена система защиты данных, производительность и гибкость.

В NetWare 2.2 фирма Novell собрала все лучшее из ранних версий NetWare. Все варианты версии 2.2 имеют одинаковые возможности и одинаковый уровень отказоустойчивости. Улучшен процесс инсталляции, имеется поддержка VAP (Value Added Processes) – отдельных программных модулей, стыкуемых с ОС NetWare и позволяющих файловому серверу выполнять некоторые дополнительные функции.

NetWare 3.12 использует преимущества процессоров 80386, 80486 или Pentium. Она предоставляет возможности присоединение к одному серверу до 250 пользователей, объем дисковой памяти до 32 терабайт, размер файла до 4Г, один файл может располагаться на нескольких накопителях, одновременно могут быть открыты до 100 тыс. файлов. NetWare 3.12 имеет улучшенную систему защиты данных. Также новой является концепция NLM (*NetWare Laudable Module)* загружаемых (выгружаемых) в процессе работы. В сервере могут храниться файлы для рабочих станций с разными операционными системами (DOS, Macintosh, OS/2, UNIX). Недостатком операционной системы NetWare 3.12 является система помощи и подсказок, где самым слабым местом является пользовательский интерфейс. Еще одна особенность этой версии – новый интерфейс транспортного уровня (*TLI - Transport Layer Interface)*, основанный на ODI. Этот интерфейс предоставляет широкий диапазон возможностей для организации связей, включая IPX/SPX, NetBIOS, LU 6.2 (APPC), именованные каналы связи для рабочих станций, управляемых DOS и OS/2, TCP/IP, интерфейс Berkley 4.3 Sockets и UNIX System V Stream/TLI.

NetWare 4.0 полностью совместима с предыдущими версиями, и пользователь может даже не заметить разницы. Наиболее значительной особенностью NetWare 4.0 является система NDS (*NetWare Directory Service),* представляющая собой иерархически организованную базу данных. Использована также новая система именованных директорий, что позволяет пользователям присоединяться к серверам за одну операцию, при этом доступ возможен к 54 тыс. файловым серверам (раньше эта цифра была равна 8). Новинками версии являются: система кэширования предполагаемого чтения, компрессия данных и компоновка блоков данных, улучшенная система защиты данных и ресурсов.

Операционные системы Microsoft Windows NT и Windows NT Advanced Server появились в продаже в июле 1993 года. Тогда их использовали лишь энтузиасты и крупные компании. Во многом это было связано с довольно высокими требованиями системы к аппаратуре. С выходом версии 3.5, заметно снизившей уровень этих требований и включившей в себя ряд новых функций, начался стремительный рост популярности Windows NT. Сегодня она широко применяется самыми разными организациями, банками, промышленностью и индивидуальными пользователями. Все больше становится поклонников этой удобной и надежной системы и в России. Версия 4.0 – это следующий шаг в распространении Windows NT: новый интерфейс и масса новых полезных свойств, привели к широкому внедрению этой системы на персональных рабочих местах. Сейчас еще рано что-либо говорить о Windows NT 5.0, поскольку только недавно вышла бета-версия операционной системы.

Microsoft не публикует данные об инсталлированной базе Windows NT, однако утверждает, что более 40 компаний собираются использовать Windows NT Workstation в качестве операционной системы более чем на 10 тыс. ПК. К тому же, по данным нескольких консалтинговых фирм, в месяц продаётся более 30 тыс. копий ОС Windows NT Server. Фирмы Compaq, Dell и Getaway готовят NT - серверы на базе микропроцессоров Pentium Pro компании Intel. Ожидается, что Windows NT станет ведущей операционной системой для процессоров Pentium Pro, поскольку Windows’95 содержит 16-ти разрядный код и работает на Pentium Pro не так быстро, как полностью 32-ух разрядная NT.

Ранние версии Windows поддерживали неприоритетную многозадачность, из-за чего работа системы зависела от корректности запущенных задач. Все приложения делили процессорное время путем периодического опроса друг друга. Если какое-либо приложение отказывалось отвечать, система не знала, что в таком случае делать. В Windows NT действует принцип приоритетов, позволяющий приложениям с более высоким приоритетом "вытеснять" имеющие более низкий. Так как система всегда контролирует события, процессорное время используется эффективнее, а "сбойное" приложение не приведет к зависанию системы.

Операционная система UNIX создавалась за несколько этапов. Все началось в 1965-69 гг. в Bell Labs концерна AT&T в рамках проекта MULTICS (Multi-user Timesharing Interactive Computing System) для большой машины General Electric GE-645. В 1969 г. Bell Labs решает выйти из проекта MULTICS, чтобы сосредоточить усилия на создании мобильной операционной среды под условным названием UNIX.

Первоначально UNIX была написана на ассемблере для DEC PDP-7. Затем, в 1973 г., Денис Ритчи, который в то время уже разработал язык В, предложил переписать основную часть UNIX на В. В процессе осуществления этой идеи, язык В настолько усовершенствовался, что преобразился в С. Так было достигнуто невиданное тогда качество – мобильность. В отличие от всех операционных систем, на 100 процентов написанных на ассемблере для определенной машины, UNIX имела только 10 процентов (1000 строк) кода на ассемблере. Для того чтобы работать на произвольной машине, новая ОС нуждалась единственно в написании нескольких страничек на ассемблере и компиляторе языка С. Уже в 1976 г. в первый раз UNIX была перенесена на другую машину - Interdata 8/32.

В 1971 г. торговая марка UNIX была запатентована Bell Labs для серии машин DEC PDP-11/20, наиболее тогда распространенных в университетской среде. За несколько лет UNIX претерпела в Bell несколько изданий, из которых наиболее популярны были шестое (1976 г.) и седьмое (1979 г.).

Нарастающая популярность UNIX заставила Калифорнийский университет в Беркли предложить свой вариант UNIX - BSD (Berkeley Software Distribution), на базе которого по заказу DARPA (Агентство перспективных проектов военного ведомства США) компания BBN реализовала в системе BSD 4.1 протоколы TCP/IP. Так возникла сеть Интернет.

Необходимо вспомнить и разработанную в Массачусетском технологически институте систему X-Window (1984 г.). Основанная на TCP/IP, она обеспечивает мобильный графический интерфейс, к которому прилагается, концепция клиент-сервер, наиболее революционная для своего времени. Сегодня UNIX и X-Window почти неразделимы.

В это же время начались попытки стандартизации. Известный американский институт инженеров по электротехнике и электронике (IEEE), создал рабочую группу 1003, которая разработала стандарт переносимой системы (Portable Operating System). Имя этого стандарта - POSIX, который, конечно же, прежде всего, относится к UNIX. В 1990 г. документ POSIX 1003.1 с редакционными изменениями был принят в качестве стандарта ISO.

Другим опытом стандартизации UNIX является документ Х/Open Portability Guide. Популярно третье издание - XPG3 (1989 г.), которое основано на POSIX 1003.1, но содержит и ряд новых элементов, рассматривающих не только ОС, но и потребительский интерфейс, базы данных, коммуникации.

Шагом к стандартизации UNIX является и появление в 1989 г. ANSI стандарта для языка С (16 лет спустя после его рождения).

Долгим и тернистым был также и путь UNIX на рынок программных средств. Считается, что только с 1 января 1984 г. дочерняя компания AT&T Bell Labs (позднее переименованная в USL - UNIX System Laboratories) вышла на рынок с UNIX в качестве торгового продукта. Под благовидным предлогом стандартизации UNIX, Компания AT&T ввела SVID (System V Interface Definition), и этим ходом вновь отождествила UNIX со своей System V (1983 г.).

Другим важным событием в 1987 г. стало соглашение АТ&Т с ведущими UNIX производителями Sun и Microsoft, о так называемой унификации UNIX. Проект предусматривал создание четвертого издания System V (SVR4), которая объединяла характеристики XENIX Microsoft (другое на звание UNIX для микрокомпьютеров, основанной на седьмом издании и испытавшей сильное влияние System V), SunOS (система UNIX фирмы Sun Microsystems, основанной на BSD) и System V 3.2. В 1987 г. AT&T в первый раз лицензировала имя UNIX.

В мае следующего года несколько ведущих компаний, между которых были Apollo, Bull, HP, IBM и Siemens, учредили Фонд открытых систем OSF (Open Software Foundation) - организацию с целью разработки и распространении открытых программных систем. Она финансировала разработки программного обеспечения в соответствии с наиболее современными требованиями к открытым системам, специфицированными в соответствующих стандартах для разработки: системный интерфейс OSF/AES, графический потребительский интерфейс OSF/Motif, распределенные системы OSF/DCE и т. д. Основой проекта являлся выбор UNIX-технологии. После внимательного изучения выбрали ядро Mach, разработанное в университете Карнеги-Меллон (100 тыс. строк исходного кода; код первой UNIX был на порядок скромнее), а все остальное (OSF-1 имеет, в общем, около 800 тыс. строк кода) предлагалось взять из IBM AIX третьей версии.

Выбор AIX не понравился AT&T. Концерну было предложено принять членство OSF, но он поставил неприемлемое условие заменить AIX на SVR4. В результате AT&T, Sun, UNISYS, XEROX и др. создали в марте 1989 г. организацию UNIX International (UI), которая дала широкую дорогу USL. Разработка и лицензирование софтвера оставалась привилегией USL, но общий контроль был сохранен за AT&T. Взаимные столкновения и несовместимость продуктов OSF и UI явились кульминационным пунктом в так называемых UNIX-войнах.

Между тем уже в наши дни появился Windows NT, который в отличие от своих предшественников DOS и Windows располагает арсеналом средств настоящей ОС. Этот факт заодно с безграничным доверием и поддержкой миллионов клиентов Microsoft всерьез стал угрожать UNIX-производителям. В июне 1992 г. OSF и UI объявили о совместном проекте Destiny, который положил конец UNIX-войнам.

Вдобавок основной конкурент Microsoft в мире персональных компьютеров фирма Novell сделала ряд вполне понятных шагов навстречу UNIX. Компания стала собственником USL (декабрь 1992 г.) и благородным жестом передала торговую марку UNIX в руки Комитета X/Open. X/Open создал XPG4, с которым примирил SVID и AES. Была разработана и серия тестов для отбора в кандидаты открытых систем.

Событием большого значения для будущего открытых систем является проект COSE, созданный ведущими производителями программного обеспечения. Сейчас в этом направлении сделан первый шаг (Common Open Desktop), унифицированный графический потребительский интерфейс. Кроме того, мир открытых систем, безусловно, будет обогащен и развитием объектно-ориентированных технологий.

***Базовые Характеристики.*** Все три представляемые ОС являются многозадачными. Однако в NetWare многозадачность кооперативная, и к тому же процессы на сервере выполняются в режиме ядра системы (отсутствует защита памяти процессов). Это делает применение данной системы потенциально опасным в качестве сервера приложений, поскольку любое некорректно написанное приложение может легко "повесить" сервер. Вместе с тем именно такая функциональная особенность ОС обуславливает исключительно высокую производительность, поскольку минимальны накладные расходы, связанные с переключениями между процессами, а также при переходах из пользовательского режима в режим ядра ОС и наоборот. При всех прочих одинаковых условиях NetWare имеет по определению более высокую, чем Windows NT или Unix, производительность.

Все три ОС поддерживают симметричную многопроцессорную обработку (SMP), но, исходя из специфики этой системы, ценность ее для NetWare 4.1 довольно сомнительна, да и используется она только для узкого круга сертифицированных серверов. Для многопроцессорной обработки лучше применять Windows NT или Unix, причем Windows NT и многие версии Unix прекрасно работают на серверах с несколькими десятками процессоров.

Хорошо себя зарекомендовали и кластерные системы на основе Unix, но, разумеется, далеко не всякий Unix для этого годится. Для Windows NT разработка кластерных технологий полностью еще не закончена.

Отказоустойчивость является на сегодняшний день одной из наиболее важных характеристик, и разработчики операционных систем уделяют особое внимание этому вопросу.

В версиях 2.2, 3.12, 4.0 и 4.10 NetWare применена технология SFT *(System Fault Tolerant- система защиты при отказах оборудования*). Система защиты при отказах оборудования означает бесперебойную работу файлового сервера при различного рода отказах аппаратных средств. Во всех версиях NetWare имеются средства минимизации потерь данных в случае физических повреждений поверхности накопителей. Система SFT пошла дальше в этом отношении, предложив методы *зеркального отображения дисков* и *дублирования дисков.* В ОС NetWare имеется возможность контроля сигналов источника бесперебойного питания.

Многие версии Unix поддерживают различные варианты избыточной аппаратной отказоустойчивости, хотя эта особенность строго индивидуальна и напрямую зависит от используемого аппаратного обеспечения. Однако эталоном отказоустойчивости остаются мэйнфреймы; именно на них равняются при создании отказоустойчивых программно-аппаратных комплексов.

Надёжность и отказоустойчивость Windows NT обеспечиваются архитектурными особенностями, которые защищают прикладные программы от повреждения друг другом и операционной системой. Операционная система использует отказоустойчивую структурированную обработку особых ситуаций на всех архитектурных уровнях, которая включает восстанавливаемую файловую систему NTFS.

Благодаря модульному построению системы обеспечивается *расширяемость* Windows NT, что позволяет гибко осуществлять добавление новых модулей на различные уровни операционной системы. Основными модулями Windows NT являются уровень аппаратных абстракций HAL (Hardware Abstraction Layer), ядро (Kernel), исполняющая система (Executive), защищенные подсистемы (protected subsystems) и подсистемы среды (environment subsystems).

*Уровень аппаратных абстракций* виртуализирует аппаратные интерфейсы, обеспечивая тем самым независимость остальной части операционной системы от конкретных аппаратных особенностей. Подобный подход позволяет обеспечить легкую переносимость Windows NT с одной аппаратной платформы на другую.

*Ядро* является основой модульного строения системы и координирует выполнение большинства базовых операций Windows NT. Этот компонент специальным образом оптимизирован по занимаемому объёму и эффективности функционирования. Ядро отвечает за планирование выполнения потоков, синхронизацию работы нескольких процессоров, обработку аппаратных прерываний и исключительных ситуаций.

*Исполняющая система* включает в свой состав набор программных конструкций привилегированного режима (kernel-mode), представляющих базовый сервис операционной системы подсистемам среды. Исполняющая система состоит из нескольких компонентов, каждая из которых предназначена для поддержки определённого системного сервиса.

*Подсистемы среды* представляют собой защищённые серверы пользовательского режима (user-mode), которые обеспечивают выполнение и поддержку приложений, разработанных для различного операционного окружения (различных операционных систем). Примером подсистем среды могут служить подсистемы Win32 и OS/2.

***Безопасность.*** Система защиты Novell NetWare включает в себя защиту от несанкционированного доступа, а также ограничения на доступ (пользователей с определенными именами в определенное время дня). NetWare 4.1 соответствует требованиям класса безопасности C2, предъявляемым к сетевым конфигурациям ("Красная книга").

Windows NT обеспечивает защиту с помощью встроенной системы безопасности и усовершенствованных методов управления памятью. Система безопасности удовлетворяет спецификациям правительства США и соответствует стандарту безопасности С2. В корпоративной среде критическим приложениям обеспечивается полностью изолированное окружение.

Практически все современные системы Unix либо в базовой поставке, либо, задействуя дополнительный модуль, соответствуют классу безопасности C2 для рабочей станции, а иногда и более высокому. Некоторые версии Unix отвечают также классу C2 и для сетевой конфигурации.

***Многопользовательский Интерфейс.*** Преимуществом Unix является поддержка многопользовательского интерфейса, которая отсутствует у NetWare и Windows NT.

К одной Unix-машине, даже на базе ПК, можно подключать десятки алфавитно-цифровых терминалов. Если же терминалов требуется слишком много, то приобретают мэйнфрейм (причем сегодня, а не во времена царя Гороха). Мэйнфреймов же IBM продает на сумму, которая для любого производителя ПК является пределом мечтаний, но у нас про это говорить как-то не принято. Хороший мэйнфрейм без труда потянет работу нескольких сотен, а то и тысяч терминалов.

Несколько по-другому обстоят дела с сетевым графическим интерфейсом. В мире открытых систем стандартом на сетевую поддержку графического интерфейса является X Window System. В этой системе компьютер, на котором выполняется задание, называется X-клиентом. Устройство (в его качестве может выступать и компьютер, и X-терминал), с клавиатуры и мыши которого осуществляется ввод информации и на дисплей которого производится вывод графического изображения, называется X-сервером. При этом X-клиент и X-сервер могут выполняться как на одном и том же компьютере, так и на различных, порой несовместимых машинах.

Хотя X (сокращенное обозначение X Window System) разрабатывался независимой от конкретной ОС, его реализации могут довольно сильно отличаться в различных операционных системах. X прекрасно адаптирован для Unix, причем имеется множество расширений, значительно увеличивающих возможности стандартной поставки.

Как известно, Microsoft при разработке Windows NT не стала делать графический интерфейс сетевым. Не поставляет она и продукты, реализующие X Window System в этой ОС. Тем не менее, ряд независимых компаний предлагает такие программы.

Для NetWare 4.1 поддержка X вообще отсутствует (кроме простой программы X-console).

***Логическая Организация Сетевых Ресурсов.*** В NetWare 4.1 имеется прекрасное средство организации сетевой инфраструктуры - служба каталогов NetWare (NDS). Она позволяет строить иерархию сетевых ресурсов на глобальном уровне. При этом пользователю для доступа к любому ресурсу довольно один раз зарегистрироваться в сети. Недостатком NDS является недостаточное количество прикладных программ для этой отличной службы.

Что касается реализации NDS на других платформах, то уже сейчас SCO UnixWare (которая раньше принадлежала Novell) поддерживает NDS. Кроме того, фирма Novell заключила соглашение с Hewlett-Packard и SCO о переносе NDS в среду Unix этих фирм.

В Windows NT Server 3.51 и 4.0 сеть строится на основе доменов, что даже по признанию Microsoft менее привлекательно для корпоративных клиентов, чем служба каталогов. В Windows NT 5.0 Microsoft ввела новую службу каталогов NT Directory Service или, как ее еще называют – Active Directory. Она обеспечивает глобальное управление каталогами в сетевой среде масштаба предприятия и сравнима по своим функциям со службой Novell Directory Services (NDS).

Для Unix использование сетевой информационной службы NIS, разработанной фирмой Sun, дело обычное. Эта служба позволяет организовывать сетевые ресурсы на основе доменов, однако, в отличие от Windows NT, она не помогает устанавливать между доменами доверительные отношения. Управление NIS довольно сложная задача: редактировать многочисленные текстовые файлы приходится "врукопашную".

В настоящее время некоторые производители Unix разрабатывают свои службы каталогов, обычно на основе стандарта X.500. Однако представляется гораздо более перспективным для них использование службы каталогов либо фирмы Novell, либо Microsoft.

Если говорить откровенно, самым слабым местом построения корпоративной гетерогенной сети является разобщенность сетевых ресурсов и невозможность применения единой службы (каталогов или доменов), начиная от ПК и кончая мэйнфреймами и суперкомпьютерами.

***Службы Файлов и Печати.*** Судя по тестам, самой производительной сетевой файловой системой обладает NetWare 4.1. Очень близко к ней находится Windows NT, но все-таки при большом количестве клиентов и при подключении сервера к высокопроизводительному сетевому каналу архитектурные особенности NetWare дают ей некоторые преимущества.

Файловый сервер в NetWare является обычным ПК, сетевая ОС которого осуществляет управление работой ЛВС. Функции управления включают координацию рабочих станций и регулирование процесса разделения файлов и принтеров. Сетевые файлы всех рабочих станций хранятся на жестком диске файлового сервера, а не на дисках рабочих станций.

Кроме возможности выполнения приложений, написанных для других операционных систем, в Windows NT поддерживаются и различные типы файловых систем. Можно использовать жесткий диск, отформатированный в одной из трех систем: FAT, HPFS и NTFS (разработанной специально для Windows NT). Очень надежная, файловая система NTFS позволяет применять длинные имена файлов и контролировать доступ к определенным файлам. В 4 версии файловая система HPFS больше не поддерживается, так как не предлагает никаких преимуществ по сравнению с двумя другими.

Сетевая файловая система - самое слабое место в Unix. Стандартной для Unix и общепринятой в мире открытых систем является NFS. Она была разработана Sun более десятка лет назад. С тех исторических времен для Unix было предложено много других, более совершенных спецификаций сетевых файловых систем, но они не получили широкого распространения.

NFS отличается низкой производительностью, ограниченными средствами управления правами и недостаточной защищенностью от несанкционированного доступа. К тому же многие системы Unix имеют ограничения в 2 Гбайт (что слишком мало для современных программ) на максимальный размер файла.

Службы печати во всех трех ОС достаточно близки по своим функциональным возможностям, и, какая из них лучше, оценить трудно. Несомненно, только то, что удобнее всего управлять печатью в Windows NT, а сложнее всего - в Unix.

***Поддержка Аппаратных Платформ.*** NetWare 4.1 может быть установлена только на компьютерах с процессорами Intel x86. Одно время Novell собиралась перенести эту ОС на PowerPC, но, похоже, она отказалась от своих намерений.

ОС Unix имеется, кажется, на всех мыслимых платформах. Но значит ли это, что если вы работаете в SCO OpenServer, то знаете IBM AIX? Конечно же, нет! Более того, многие программы требуют серьезной переработки в исходных текстах для переноса из одной системы Unix в другую. Каждая версия Unix предназначена обычно лишь для одной аппаратной платформы. Единственным, пожалуй, приятным исключением является ОС Solaris, которая реализована для процессоров SPARC, x86 и PowerPC. Однако найти необходимую программу для Solaris x86 или PowerPC - дело явно неблагодарное (слишком их мало).

Windows NT можно установить на самых различных типах компьютеров, список которых продолжает расти. Сегодня поддерживаются Intel-компьютеры с процессорами 386, 486, Pentium и Pentium Pro, а также три типа RISC-процессоров: PowerPC, MIPS R4000 и DEC Alpha. По количеству поддерживаемых платформ она не намного обошла Solaris. Что касается прикладных программ для "неродных" аппаратных платформ, то здесь ситуация полностью аналогична Solaris - программ адаптированных для Windows NT не Intel платформ практически существует.

Все три системы неплохо подходят в качестве сервера удаленного доступа, но лучше для этого использовать не обычный компьютер, а специализированные серверы типа AccessBuilder фирмы 3Com или LanRover/E фирмы Shiva.

В качестве платформы для сервера групповой обработки информации, сервера электронной почты и факс-сервера могут применяться все три ОС, но обычно такие программы для Unix стоят дороже (кроме SMTP, поставляемой бесплатно), а управлять ими сложнее.

Web-серверы существуют на всех трех ОС. Однако чаще всего в качестве платформы Web-серверов используют Unix, в особенности Solaris. Думается все же, что в России главенствующую роль будет играть Windows NT Server 4.0 со своим Internet Information Server, поскольку это приложение поставляется с самой системой бесплатно, а такого рода доводы обычно сильнее всех остальных. Тем не менее, Internet до сих пор остается вотчиной Unix-серверов. Они поддерживают максимальное количество протоколов и приложений TCP/IP. Поэтому, если планируется серьезное использование Internet, стоит обратить пристальное внимание на Unix.

При подключении локальных сетей к Internet особую и важную роль играют брандмауэры. Практически все они реализованы на основе той или иной версии Unix.

***Сервер Баз Данных.*** Из-за отсутствия вытесняющей многозадачности и защиты памяти NetWare вряд ли можно назвать подходящей платформой для сервера БД. Windows NT и Unix-машины хорошо зарекомендовали себя в качестве сервера БД, но, благодаря большей масштабируемости и наличию кластерных технологий, Unix лучше подходит в качестве мощного сервера БД для распределенных сетей.

***Сервер Прикладных Программ.*** По совершенно не понятной причине, когда говорят о сервере приложений, часто под этим термином подразумевают сервер БД. Никто не спорит, сервер БД очень важен, но это только одно из возможных приложений. Зачем же подменять понятие сервера приложений понятием сервера БД (а еще краше SQL-сервером).

Поскольку о серверах БД уже упоминалось, мы поговорим об использовании ОС в качестве программной платформы для общего сервера приложений, каким, например, является сервер вычислений.

Допустим, для каких-либо дорогих и очень ресурсоемких программ (серьезных САПР или, скажем, программ расчета и отображения динамических процессов в ядерном реакторе) понадобилось приобрести мощную вычислительную технику категории "Number Crashing" ("перемалыватель чисел"), допустим 12-процессорный Alpha-Server. Конечно же, на него поставят Unix (или OpenVMS) с тем, чтобы его вычислительной мощью могли пользоваться одновременно несколько пользователей. Для этого обычно задействуют различного типа терминалы, рабочие станции или эмуляторы терминалов на ПК. Из-за отсутствия встроенной поддержки многопользовательского интерфейса Windows NT не очень хорошо подходит для такого сервиса (а о NetWare лучше даже и не заикаться).

Правда в последнее время растет нужда в системах, способных исполнять основное – "тяжелое" – приложение на мощном высокопроизводительном сервере, а результаты деятельности по запросам передавать на маломощные клиентские станции, реализуя модель клиент-сервер. Переход с больших мэйнфреймов на современные системы на базе ПК средней и большой мощности как раз и требует такого решения. Windows NT, изначально построенная по схеме клиент-сервер, отлично приспособлена для работы в системах клиент-сервер в качестве сервера приложений. В первую очередь такими приложениями являются системы управления базами данных, системы информационного обмена, системы управления. Именно поэтому в Microsoft BackOffice входят Microsoft SQL Server – сервер баз данных, Microsoft System Management Server – сервер управления системой, Microsoft Exchange – сервер информационного обмена, SNA Server – сервер связи с мейнфреймами и Internet Information Server – сервер Internet.

Кроме приложений корпорации Microsoft, существует более 2000 разработок других фирм: серверы баз данных (Informix, Oracle, IBM и т.д.), системы управления сетями (HP, DEC), управления производством (SAP), документооборота (Lotus, Saros), финансовые (Platinum) и многие другие системы для бизнеса.

У Windows NT есть поддержка Network OLE. Аналогичные спецификации для Unix пока только в стадии разработки, и еще не понятно, как они будут стыковаться с клиентскими местами на основе Windows.

***Администрирование.*** Простота администрирования зависит не только от продуманности пользовательского интерфейса административных утилит, но и от возможностей самой ОС.

Хотя NetWare 4.1 имеет неплохой набор довольно удобных утилит, все-таки Windows NT Server 4.0 - вне конкуренции. Хорошо продуманный интерфейс плюс богатые возможности утилит делают ее простой в администрировании, и это, не говоря о скрытой мощи.

Тем не менее, в базовом комплекте данных ОС отсутствует много важных утилит (в частности хорошего командного процессора наподобие shell в Unix).

С Unix ситуация несколько иная. Эта ОС имеет такой огромный и богатый набор утилит, который хватило бы на несколько других ОС. Одних общеизвестных командных процессоров в Unix существует, как минимум, три. Но программы Unix, разработанные разными организациями и людьми, обладают порой несовместимыми друг с другом пользовательскими интерфейсами. Многие из них до сих пор работают с командной строки. Хуже того, одна и та же утилита в разных версиях Unix может иметь разные наборы аргументов и опций. Поэтому для того, чтобы администратора Solaris перевести на администрирование AIX, его сначала необходимо переобучать.

Хочется отметить тем не менее, что ситуация с Unix меняется в лучшую сторону. Сейчас в соответствии с общепринятым стандартом многие версии этой ОС снабжаются хорошо проработанными административными утилитами с графическим интерфейсом. И не стоит судить об Unix по тому, как эта ОС выглядела 15 лет назад.

***Поддержка Клиентов.*** Благодаря тому, что NetWare и Windows NT разрабатывались для обслуживания сетей ПК, они обеспечивают хорошую поддержку основных клиентских операционных систем: MS-DOS, Windows 3.х, Windows 95, Windows NT Workstation, OS/2, Macintosh System 7.5. Поддерживаются в качестве клиентов и некоторые версии Unix, но далеко не все и не всегда гладко. Эталоном здесь может служить SCO UnixWare, имеющая прозрачную интеграцию с NetWare 4.1.

Для сетей на основе Unix до недавнего времени проблема поддержки клиентских ОС стояла более серьезно (если не считать клиентов на основе Unix). Но сейчас почти все Unix имеют те или иные средства интеграции клиентов DOS/Windows. Кроме того, Windows’95 и Windows NT Workstation уже поставляются с приложениями ftp, telnet, ping, Internet Explorer и др. Правда, этот набор слишком невелик, да к тому же возможности некоторых из них, мягко говоря, небольшие.

На рынке, однако, имеется очень много программных продуктов различных фирм, реализующих практически весь спектр услуг Unix (и не только Unix) для операционных систем на ПК. Так, Novell поставляет пакет LAN WorkPlace, в который входят: ftp-клиент и ftp-сервер для DOS и Windows; NFS-клиент; telnet для DOS и Windows; TN3270 для Windows; Web-браузер; сервер и клиент X Window System; и др. При этом все приложения данного пакета прекрасно интегрируются с NetWare.

Windows NT имеет встроенные в систему сетевые возможности, что также позволяет обеспечить связь с различными типами host-компьютеров (благодаря поддержке разнообразных транспортных протоколов) и использованию средств клиент-сервер высокого уровня (включая именованные каналы, вызовы удалённых процедур).

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование** | **NetWare 4.1** | **Windows NT Server 4.0** | **Unix** |
| Многозадачность | кооперативная | вытесняющая | вытесняющая |
| Защита памяти отдельного процесса | нет | Есть | есть |
| Многопоточность | есть | Есть | есть |
| Поддержка SMP | есть | Есть | есть |
| Кластеры | нет | Есть | есть |
| Избыточная отказоустойчивость | SFT III | Нет | есть для некоторых версий |
| Сертификация по C2 | сеть | Есть | разные варианты для различных версий |
| Поддержка алфавитно-цифровых терминалов | нет | Нет | есть |
| Сетевой графический интерфейс | нет | у сторонних фирм | есть |
| Логическая организация ресурсов | служба каталогов | Домены  (AD в NT 5.0) | домены (NIS) |
| Быстродействие сетевой файловой системы | отличное | Очень хорошее | низкое (NFS) |

К сожалению, нет в мире совершенства. Ни одна сетевая операционная система не может удовлетворить всем требованиям, предъявляемым при создании корпоративной гетерогенной сети: каждая из них имеет свои плюсы и свои минусы. В гетерогенной среде, где помимо ПК присутствуют Unix-машины, наиболее разумным подходом будет комбинированное использование сетевых ОС (Unix+Windows NT Server 4.0). Тем не менее в сети, где нет Unix-машин, использование Unix-серверов не имеет большого смысла.

Еще один фактор, несомненно влияющий на выбор операционной системы сервера, это операционная система клиентских машин. На сегодняшний день распространение Windows’95 и Windows NT настолько велико, что практически на всех компьютерах установлена одни из версий. Багатый выбор офисных продуктов, удобный и теперь уже привычный интерфейс пользователя, легкость и гибкость настройки и многое другое несомненно привлекает в этих операционных системах.

***Выбор СУБД*** для реализации довольно сложен, хотя после выбора операционной системы сервера достаточно очевиден. Прозрачный и не отегощенный не нужными функциями, Microsoft SQL Server является оптимальным решением. По всем международним тестам он находится среди лидеров реляционных СУБД. Пиковая производительность систем и их стоимость в пересчете на одну транзакцию продолжают быстро улучшаться. В настоящее время лидерство по отношению стоимость/производительность принадлежит комбинации продуктов Compaq/Windows NT/Microsoft SQL .

***Выбор языка программирования.*** Delphi - это потомок Турбо Паскаля, который был выпущен для операционной системы CP/M в 1983 году. В феврале 1994 года Турбо Паскаль был перенесен на операционную систему MS-DOS.

На раннем этапе развития компьютеров IBM PC, Турбо Паскаль являлся одним из наиболее популярных языков разработки программного обеспечения - главным образом потому, что это было вполне серьезный компилятор, который, включая компилятор, редактор и все остальное, стоил всего $19.95 и работал на машине с 64 Kb оперативной памяти.

Под Windows - Турбо Паскаль был перенесен фирмой Borland в 1990 году. А самая последняя версия Borland Pascal 7.0 (имеющая теперь такое название), не считая Delphi, вышла в свет в 1992 году. Разработка Delphi началась в 1993 году. После проведения beta-тестирования Delphi показали на "Software Development '95". И 14 февраля 1995 года официально объявили о ее продаже в США. В торговлю Delphi попала спустя 14 дней, 28 февраля 1995 года.

Delphi обладает рядом отличительных особенностей. *Многократно используемые и расширяемые компоненты*. Delphi устраняет необходимость программировать такие компоненты Windows общего назначения, как метки, пиктограммы и даже диалоговые панели и множество других многократно используемых компонентов, которые позволяют экономить время и программные усилия при разработках для Windows. В Delphi также имеются предварительно определенные визуальные и не визуальные объекты, включая кнопки, объекты с данными, меню и уже построенные диалоговые панели. С помощью этих объектов можно обеспечить вывод данных просто несколькими нажатиями кнопок мыши, не прибегая к программированию. Предоставляемый Delphi внушительный список объектов ставит эту систему во главе сред разработки, предоставляющих архитектуру повторно используемых компонентов.

*Поддержка стандарта* VBX(Visual Basic Extensions). Delphi дает возможность интегрировать VBX-объекты непосредственно в Палитру компонентов (Component Palette) для облегчения доступа к этим объектам и инструментам.

*Шаблоны приложений и форм.* Delphi предоставляет встроенные шаблоны форм и приложений, которые можно использовать для того, чтобы быстро начать разработку собственных прикладных программ. В систему также включены часто используемые диалоговые панели.

*Настрой на среды разработки.* Палитра компонентов, редактор кода, шаблоны приложений и форм – примеры областей, где Delphi может быть полностью настроена в соответствии с пожеланиями программиста.

*Компилируемые программы.* Утверждается, что другие известные визуальные средства разработки приложений Windows также «компилируют» программы. Однако это не совсем верно, в действительности происходит компиляция только части программы и последующая компоновка программы-интерпретатора и Р-кода в исполняемый модуль. Так работали раньше, хотя многие программисты сталкивались с ограничением производительности при использовании этого подхода. Delphi не использует ни интерпретатор, ни Р-код и создает действительно откомпилированные программы, готовые для исполнения. Поэтому программы Delphi столь же быстры, как и программы, написанные на языках третьего поколения. Delphi является самым быстрым в мире инструментом разработки баз данных. Простые программы Delphi могут поставляться в виде единственного исполняемого модуля без дополнительных библиотек DLL, необходимых при использовании иных сред разработки.

Существует два важных различия между файлами EXE, созданными в Delphi, и файлами EXE, созданными VB. Delphi создает чисто машинный код, непосредственно исполняемый компьютером, в то время как VB транслирует исходный код в промежуточную форму (р-код). Файл EXE, сгенерированный VB, в действительности является программой-интерпретатором р-кода с добавленным в конце р-кодом программы пользователя.

"Библиотека времени выполнения" (run-time library) стандартных функций для всех программ VB хранится в файле VBRUN300.DLL. Каждая программа VB, попавшая к конечному пользователю, должна включать этот файл, либо приходится рассчитывать, что такой файл у пользователя уже есть. Дистрибутивный комплект программы должен также содержать файлы VBX для каждого управляющего средства VB, не включенного в VBRUN300.DLL. Программы Delphi включают необходимую часть библиотеки времени выполнения Delphi, а также используемые компоненты. В результате EXE-файл Delphi обычно больше по объему, чем эквивалентный EXE-файл VB, но он не зависит ни от каких внешних файлов.

*Широкие возможности доступа к данным.* В Delphi встроен BDE – механизм работы с базами данных (Borland Database Engine). BDE является тщательно продуманной системой, результатом развития ODAPI и IDAPI. BDE в настоящее время является стандартным промежуточным слоем, используемым для доступа ко всем популярным форматам современных баз данных. BDE также используется в системах «клиент/сервер» и обеспечивает доступ к таким продуктам этого типа, как Sybase SQL Server, Microsoft SQL Server, Oracle и Borland Interbase. Можно утверждать, что BDE просто блистает на фоне ODBC – продукта Microsoft, обеспечивая существенный выигрыш в производительности за счет более тесной связи с форматами баз данных.

## 2.3 Разработка структуры новой БД

Разрабатываемая программа должна работать как автономно, так и как модуль общей бухгалтерской программы. Исходя из таких требований, на проектирование структуры базы данных накладываются некоторые дополнительные требования. С одной стороны, база данных должна нести в себе всю информацию для нормального функционирования. С другой стороны, необходимо предусмотреть возможность интеграции или слияния с базой данных главной программы.

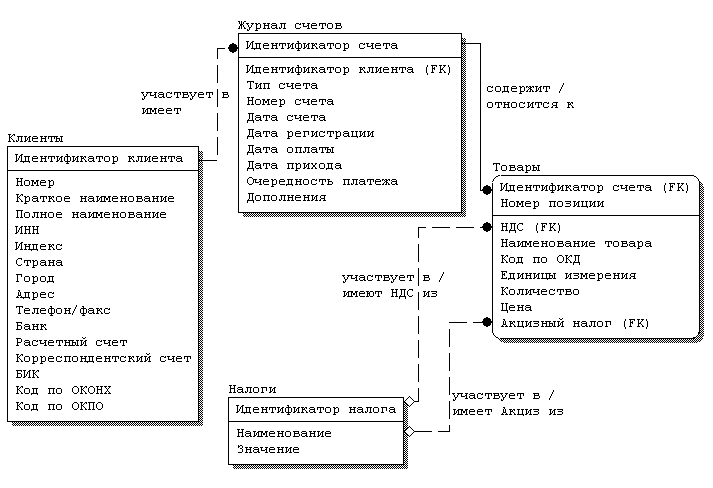
Во-первых, база данных должна хранить информацию о клиентах фирмы, о государственных налогах (налог на добавочную стоимость, акцизный налог). В связи с тем, что значения налогов могут изменяться, заводить эту информацию в программу нельзя. Анализируя структуру базы данных фирмы «ИНФИН», можно заметить, что неразумно разносить отдельно информацию по выданным и полуученым счетам фактуры. Разумнее объединить ее в одну таблицу, добавив еще один признак счета-фактуры – полученный или выданный. Информацию о суммах счета-фактуры хранить не обязательно – ее можно вычислить и таблицы товаров.

Таблица ***Клиенты*** является справочником, в котором хранится вся информация по клиентам фирмы. ***Идентификатор клиента*** является первичным (уникальным) ключом для таблицы. Альтернативным ключом является поле ***ИНН***. По любому и этих ключей можно однозначно определить клиента. Второй справочной таблицей является таблица налогов ***"Налоги"***. Поле ***Идентификатор налога*** является ее первичным ключом. В этой таблице хранится информация, как по НДС, так и по Акцизному налогу.

Основной таблицей в базе данных является таблица ***Журнал счетов***. В таблице хранится информация обо всех выданных/полученных счетах фактуры. В ней хранится только общая информация о счете. Поле ***Идентификатор счета*** является первичным колючем, а поле ***Идентификатор клиента*** является внешним колючем для этой таблицы. По внешнему ключу ***Идентификатор клиента*** можно однозначно определить поставщика/покупателя, и все его реквизиты, для данного счета. Дочерней таблицей для ***Журнала счетов*** является таблица ***Товары***. В ней хранится информация о товарах для каждого счета. ***Идентификатор счета,*** являясь внешним ключом, является вместе с ***Номером позиции*** также первичным ключом, по которому можно однозначно определить параметры товара. Внешними ключами также являются ***НДС*** и ***Акцизный налог*** от таблицы ***Налоги***.

Связи:

***Клиенты – Журнал счетов***. Эта связь не обязательная со стороны ***Клиентов*** и обязательная со стороны ***Журнала счетов***. Это значит, что может существовать клиент, на которого еще не заведено ни одного счета-фактуры. С другой стороны не может существовать счета-фактуры, у которого нет ссылки на клиента или такой ссылки нет. При изменении записей в таблице ***Клиенты***, автоматически изменяются ***Идентификаторы клиента*** в ***Журнале счетов***. Удаление клиента, у которого уже существует счет-фактуры невозможно.

***Налоги – Товары (НДС и Акциз)***. Эта связь не обязательная со стороны ***Налогов*** и необязательная со стороны ***Товаров***. Это значит, что может существовать еще нигде не использованный налог, и может существовать товар, не облагаемый налогом. При изменении значений в таблице ***Налоги***, автоматически изменяется значение в таблице ***Товаров***. С другой стороны удаление какого-нибудь налога не ведет к удалению товара. Ссылка на налог устанавливается в значение NULL.

***Журнал счетов – Товары***. Таблица ***Товары*** является зависимой от таблицы ***Журнал счетов***. Это значит, что не может существовать товара без счета-фактуры. В тоже время у одного счета-фактуры может существовать несколько товаров. При изменении в ***Журнале счетов*** автоматически изменяются записи в таблице ***Товары***. При удалении счета-фактуры, удаляются все товары, связанные с этим счетом-фактуры.

## 2.4 Перенос данных в новую базу данных

Как уже отмечалось структура базы данных старой программы фирмы «ИНФИН» отличается древовидной структурой базы данных, а точнее расположением файлов базы данных. Приблизительная структура показана на рисунке.

Как легко заметить, новая структура базы данных намного легче в понимании и обращении с ней. Ветвистость директорий в базе данных «ИНФИН» никому не нужна. Но, к сожалению, данные из старой базы данных надо переносить. Перенос данных придется осуществлять последовательным перебором всех поддиректорий и выгрузкой данных в SQL сервер.

К сожалению, экспорт информации в Microsoft SQL Server из DBF файлов проводится, не может. Однако существует утилита командной строки DOS позволяющая осуществлять экспорт данных из текстового файла, сформированного определенным образом (формат файла для экспорта показан в приложении А). Поэтому следует проводить экспорт данных из DBF формата в SQL базу данных через промежуточный текстовый файл.



Под форматирование подразумевается приведение таблицы по типу, названию и расположению полей к формату SQL таблицы

Экспорт данных в ТХТ формат

Файл формата таблицы создается единожды для данной таблицы.

При помощи утилиты командной строки

Формат DBF файлов осуществлялся при помощи утилиты для работы с файлами DBF формата – DBU.EXE. В основном, это ручное форматирование структуры файла, перенос данных, отчистка от пустых строк и т.д. Экспорт файла осуществлялся при помощи программы Excel. В Excel осуществлялся вторичный контроль над данными, заполнение незаполненных, но обязательных для заполнения в SQL, полей. Далее данные экспортировались в ТХТ файл с разделителями «;».

При помощи утилиты командной строки bcp.exe создавался fmt файл с форматом SQL таблицы. Это необходимо для дальнейшего экспорта данных. Файл формата таблицы создается единожды для данной таблицы.

К сожалению, при экспорте из Excel, все текстовые строки заключает в кавычки, поэтому приходилось их вручную удалять. Последним этапом был сам экспорт файла, при помощи все той же утилиты командной строки.

Ручное форматирование и преобразование файлов достаточно длительный процесс. Поэтому был предложен альтернативный путь. Был произведен экспорт только справочника клиентов. А далее было предложено перейти на новую программу с нового года, и в новом году работать на новой программе, а в старом – на старой. Это еще обусловлено тем, что Центральный Банк с 1998 года ввел новый план счетов и, как следствие, у всех фирм поменялись реквизиты. Но все равно экспорт остальной информации был проведен в тестовом режиме.

## 2.5 Разработка программы

При запуске программы, первым делом, должна проводится авторизация пользователя. Для этого выводится стандартное диалоговое окно, в котором пользователя просят ввести имя пользователя (Login Name) и пароль. Введенная информация посылается через ODBC-драйвер в Систему Управления Базами Данных, а точнее в службу доступа СУБД, которая проверяет введенные данные и выдает либо положительный, либо отрицательный результат. При отрицательном результате пользователю предлагается повторить попытку регистрации в системе. Если пользователь отказывается от авторизации, выбрав кнопку отмены, программа завершает свою работу, выдав сообщение о невозможности продолжать работу из-за отсутствия возможности получить данные.

После успешной авторизации пользователя системой безопасности СУБД, происходит этап настройки. Для этого программа ищет настроечный файл, в котором хранится информация о предприятии пользователя и индивидуальные настройки для каждой станции сети. При отсутствии файла или при невозможности считать из него информацию, на экран выводится диалоговое окно настроек, в котором пользователю предлагается ввести информацию об его предприятии (полное и краткое наименование, полный адрес и банковские реквизиты). Это диалоговое окно не может быть закрыто и продолжена работа программы, пока пользователь не ввел все реквизиты. При ошибочном вводе или пропуске полей пользователю сообщается об ошибке и курсор автоматически позиционируется на место ошибки. В случае отказа пользователем вводить информацию и выборе кнопки отмены, программа выводит предупреждение, что часть реквизитов пусты и что программа будет аварийно завершена. При отказе пользователя от своих намерений покинуть программу, ему дается возможность исправить допущенные ошибки, а при подтверждении – программа завершается. Программу настроек можно также вызвать уже при роботе из главного окна, при помощи соответствующего пункта главного меню. Но и тогда при неправильном заполнении формы, программа автоматически завершает свою работу, а при успешном заполнении всех реквизитов предприятия программа продолжает свою работу.

Если файл настроек был найден и информация о предприятии пользователя успешно считана, то программа пытается считать из него параметры пользовательского окна и его компонент, а также параметры данных, такие как тип документа, фильтр по датам и расширенный фильтр. При отсутствии искомой информации или невозможности ее прочитать, программа подставляет значения фильтров по умолчанию зашитые в программу, т.е. выданные счета-фактуры, данные за весь период и пустой расширенные фильтр. Все эти значения подставляются в запрос к базе данных, для получения соответствующих данных. При получении данных они анализируются программой и инициализируются соответствующие кнопки на панели инструментов и соответствующие пункты главного и контекстного меню (т.к. может быть возвращен пустой массив данных и, следовательно, удаление, модификация и поиск данных невозможен).

Главное окно разделено на две логически зависимые части. В верхней, главной, находится непосредственно журнал регистрации счетов. Он получается из базы данных при помощи вспомогательной виртуальной таблицы (представления) V\_Facture. Это представление собирает из разных таблиц информацию по документу, в том числе краткое наименовании клиента и суммы документа с налогом и без него. В нижней вспомогательной части окна находится информация о товарах, содержащихся в выбранном документе. При изменении положения курсора в верхнем окне, содержимое нижнего окна соответственно изменяется. Информация о товарах тоже берется из вспомогательной виртуальной таблицы V\_Articles. Это представление содержит в себе кроме информации о товаре, также информацию о процентных ставках товара, и о суммах налогов, чистой сумме, сумме с акцизным налогом и общую сумму товара. Также главное окно имеет меню и панель инструментов, функции которого дублируют некоторые функции главного меню. Также верхняя (главная) часть окна имеет контекстное меню, функции которого дублируют некоторые функции панели инструментов.

После инициализации и получения данных от SQL сервера, программа ожидает дальнейших инструкций пользователя. Это может быть просто перемещение курсора по главной части окна. В таком случае нижняя часть, содержащая информацию о товаре, будет автоматически обновляться. При работе с программой пользователь может выбрать следующие возможности: печать или просмотр журнала регистрации или документа; просмотр и редактирование справочника по клиентам предприятия или государственным налогам; модификация (удаление, добавление, редактирование) счета-фактуры; вызов различных процедур фильтрации, такие как выбор типа документа (выданный или полученный), изменение периода просмотра информации, а также расширенный фильтр.

Вызов справочника, как налогов, так и клиентов предприятия из главной программы осуществляется при помощи выбора в главном меню «Файл» - «Справочники» и далее интересующий справочник. При выборе справочника программа пытается считать из файла настроек параметры соответствующего окна. При отсутствии данных подставляются параметры, зашитые в программу. Каждый справочник представляет собой окно просмотра информации и панели кнопок, с правой стороны, для редактирования, удаления и добавления информации, а также для выхода из справочника. Функции модификации могут быть вызваны также из всплывающего меню. Для удобства понимания и представления информации, в справочнике налогов выводится процентный эквивалент налога. При удалении записи в справочнике, пользователь должен подтвердить свое желание удалить запись. Запрос на удаление записи посылается на SQL сервер, и если, в справочнике клиентов, клиент участвует в каком-нибудь документе, SQL сервер вернет отказ в удалении клиента, при этом выдав соответствующее сообщение на экран о невозможности удаления записи из-за наличия записи в журнале регистраций, связанной с данной записью. При модификации (добавлении или редактировании) информации, пользователю на экран выводится форма (при редактировании уже заполненная данными) и предлагается ее заполнить. При неправильном заполнении или пропуске обязательных к заполнению полей и попытке записать введенную информацию, программа выдает сообщение об ошибке и курсор автоматически позиционируется на место ошибки. При правильном заполнении, программа подготавливает данные и посылает запрос к SQL серверу. После получения подтверждения от сервера о выполнении запроса программа закрывает окно ввода информации и позиционирует курсор на введенную или измененную запись в справочнике. После любой модификации данных программа производит обновление экрана просмотра информации и пере инициализацию кнопок и пунктов меню.

Модификация данных в главном окне зависит от того, было ли выбрано удаление или добавление и редактирование. Удаление записи в журнале регистрации происходит только после подтверждения пользователем. При модификации записи программа раскрывает окно для ввода информации, при этом если было выбрано редактирование, то в поля записывается соответствующая информация. Форма ввода информации разделена на две части: ввод информации по счету и ввод информации по товарам этого счета. Из формы редактирования можно попасть в формы просмотра и редактирования справочников, при этом в формах справочников появляется дополнительная возможность отменить вызов формы, а справочнике налогов также появляется возможность очистить соответствующее поле в форме ввода счета. В форме ввода счета можно вводить товары, включенные в счет, при этом возможна модификация введенной информации, такая как удаление, добавление и обновление введенной информации. Данная часть формы ввода счета содержит в себе окно просмотра введенной информации и форму ввода информации. При изменении позиции курсора в окне просмотра введенной информации по товарам, форма ввода автоматически заменяет свое содержимое. При добавлении и обновлении товаров, в программе предусмотрена проверка на правильность заполнения полей ввода. При ошибке, пользователю выдается сообщение об ошибке и автоматически позиционируется курсор на место ошибки. Если выбран режим добавления счета-фактуры, то ставка НДС по умолчанию устанавливается в 20%. При правильном заполнении всех полей программа подготавливает данные для ввода в базы данных и вводит их при помощи хранимой процедуры SP\_ADD\_FACT. Эта процедура имеет в параметрах все реквизиты документа и реквизиты одного товара. Остальные товары добавляются в базу данных циклически.

Печать документа вызывается из главного окна при помощи выбора соответствующей кнопки на панели инструментов или при выборе пункта в главном меню. При выборе печати из главного меню выводится стандартное окно выбора принтера и дальнейшей печати. Бланк документа зашит в программу и был спроектирован при помощи встроенного в Delphi дизайнера отчетов QuickReport. В документе выбран кириллический шрифт true type, размер и особенности шрифта (такие как наклон, толщина и т.д.) заданы при создании отчета, и не могут быть в дальнейшем изменены пользователем. Часть данных в форму документа передается через результат запроса (перечень товаров счета), а другая при непосредственной записи в соответствующие поля. В зависимости от типа документа (выданый или полученный) поля в документе меняются местами. Данные о предприятии пользователя считываются из файла настроек.

Печать или просмотр журнала регистрации вызывается из главного окна при помощи выбора соответствующей кнопки на панели инструментов или при выборе пункта в главном меню. При выборе печати из главного меню выводится стандартное окно выбора принтера и дальнейшей печати. Бланки отчетов (книга выданных или книга полученных) зашит в программу и был спроектирован при помощи встроенного в Delphi дизайнера отчетов QuickReport. В документе выбран кириллический шрифт true type, размер и особенности шрифта (такие как наклон, толщина и т.д.) заданы при создании отчета, и не могут быть в дальнейшем изменены пользователем. При вызове просмотра или печати журнала регистрации на экран выводится диалоговое окно, в котором пользователю предлагается ввести временной интервал, за который он хочет просмотреть или распечатать журнал регистрации. При этом если данные в главном меню уже отфильтрованы по дате, то программа подставляет этот временной интервал, если же данные небыли отфильтрованы, то подставляется текущий месяц целиком. После модификации пользователем временного интервала, интервал, вместе с типом счета, посылается с запросом в базу данных. При помощи хранимой процедуры SP\_MAKE\_BOOK, SQL сервер возвращает сформированные данные в программу. При отсутствии данных, пользователю выводится сообщение об отсутствии данных и предлагается ввести другой интервал. Пользователь может в любой момент отказаться от печати журнала регистрации. Если SQL сервер возвращает данные, то они подставляются, в зависимости от типа документа, в соответствующую форму.

Фильтрацию данных можно разделить на три части: фильтрация по типу документа, фильтрация по дате документа и расширенная фильтрация.

Фильтрация по типу документа осуществляется автоматически при переключении из журнала выданных документов в журнал полученных документов. Переключение осуществляется в главном окне при помощи кнопок на панели инструментов или при выборе соответствующего пункта главного меню. При переключении между журналами, данные закрываются, и формируется новый запрос к базе данных, при этом фильтры на дату и расширенные фильтры остаются.

Вызов пользователем фильтрации по дате активизирует подпрограмму фильтрации даты. На экран пользователя выводится диалоговое окно, в котором пользователю предлагается ввести новый временной интервал. Также в программе предусмотрена возможность отменить данный тип фильтрации, для чего на диалоговом окне существует соответствующая кнопка. Очистку фильтра пользователь должен подтвердить, так как очистка данного фильтра приведет к выдаче всех документов данного типа на экран. При выводе диалогового окна, программа считывает предыдущий фильтр, и подставляет в окно. Если данный тип фильтрации отсутствует, то программа подставляет по умолчанию только сегодняшний день. После выбора пользователем временного интервала, программа закрывает диалоговое окно и переинициализирует данные в соответствии с новым временным интервалом. После получения новых данных от сервера, программа инициализирует все кнопки на панели инструментов в пункты главного и всплывающего меню.

Вызов расширенного фильтра выводит окно, в котором пользователю предлагается ввести выражения для фильтрации данных. Это выражение должно быль написано на языке Transact-SQL, и программа подставляет его в запрос к базе данных после ключевого слова WHERE и фильтров по типу и дате документа. После этого расширенного фильтра, программа посылает запрос к базе данных и СУБД сама проводит синтаксический анализ введенного выражения. Если СУБД обнаружит ошибку, то пользователю сообщается об этом и предлагается исправить введенное выражение. При инициализации окна расширенного фильтра, программа считывает предыдущий расширенный фильтр и подставляет его в поле для ввода выражения - тем самым можно, очистив поле, очистить расширенный фильтр.

Подпрограмма поиска значений по журналу выводит диалоговое окно, в котором пользователю предлагается ввести искомое значение. При инициализации данного окна, в поле для ввода программа подставляет значение ячейки, на которой был установлен курсор. Поиск осуществляется в обоих направлениях, но тоже только по столбцу, на котором бал установлен курсор. При успешном выполнении поиска, курсор позиционируется на найденной значение, в противном случае остается на месте.

# 3. Отладка

Разрабатываемая программа должна вести журнал регистрации выданных и полученных счетов-фактур. Для ее успешной работы необходимы вспомогательные подпрограммы, поэтому программу можно представить как совокупность нескольких модулей: основная программа с вводом и корректировкой информации о счетах-фактуры, модуль справочников, состоящий из справочника клиентов и справочника налогов и модуль вывода на печать, состоящий из вывода на печать книги покупок/продаж и вывода на печать счета-фактуры.



Модуль справочников и модуль отчетов являются вспомогательными.

Для полного тестирования программы необходимо включить тесты по тестированию правильности функционирования логики базы данных. Тестирование логики работы базы данных отдельно не имеет смысла, так как при тестировании программы, неизбежно будет тестироваться и логика базы данных.

Модульность программы позволяет разбить тестирование программы на несколько этапов: тестирование базы данных, тесты отдельно каждого модуля, тестирование программы в целом и сетевое тестирование, когда несколько пользователей работают с базой данных.

Модуль справочники включает в себя подпрограмму отражения и модификации справочника клиентов фирмы и подпрограмму отражения и модификации справочника налогов. Эти две подпрограммы по своим функциональным требованиям и по логике работы одинаковы, поэтому и разрабатывать тесты нужно только один для обеих подпрограмм.

Первый тест на правильность отражения информации. Программа должна отражать именно ту информацию, которая находится в базе данных. Для проверки правильности функционирования необходимо проводить визуальное сравнение данных на экране и содержимого базы данных. Содержимое базы данных выводится на экран при помощи стандартного SQL запроса (SELECT \* FROM *<имя таблицы>*). Расхождение информации на экране и в базе данных свидетельствует об ошибке в программе при формировании запроса к базе данных или при выводе данных на экран.

Второй тест на отработку логики базы данных. Для этого вводится заведомо неправильная информация, такая как отсутствие или пустые поля, которые не могут быть пустыми; дублирование уникальных полей, т.е. первичного (*Идентификатор клиента*) или альтернативного (*ИНН*) ключей. Для локализации возможной ошибки, сначала тестируется логика работы базы данных, при помощи стандартных запросов INSERT или UPDATE. Отрицательный результат теста обозначает ошибку в логике базы данных. При тестировании программы, возникающая ошибка свидетельствует о неправильном формировании запроса к базе данных, или неправильной проверке вводимых пользователем данных.

Третий тест на правильность введения информации в базу данных. Для этого вводится информация и проверяется правильность ее отражения в базе данных. Отрицательный результат теста означает ошибку при формировании запроса к базе данных.

Модуль отчета включает в себя подпрограмму вывода на печать/просмотр журнала выданных или полученных счетов-фактур за определенных период и подпрограмму печати счета-фактуры. Подпрограмма вывода на журнала счетов-фактур должна, при наличии данных за период, выдавать их на печать, а при отсутствии сообщать об этом. Подпрограмма печати счета-фактуры должна просто печатать счет-фактуры, поэтому тестирование может заключаться в правильности вывода на принтер информации и отработки функции формировании суммы прописью.

Первый тест для подпрограммы выдачи отчета журнала счетов ориентирован в первую очередь на правильность вывода информации. Отрицательный результат теста означает либо ошибку в подпрограмме при формировании запроса или при выводе информации на принтер, либо в хранимой процедуре SP\_MAKE\_BOOK, которая формирует данные для отчета. Последнее можно проверить при помощи стандартной программы SQL Query, поставляемой вместе SQL Server, которая предоставляет интерактивный интерфейс к базе данных.

Второй тест на правильность работы отработки ввода периода. Для этого вводится заведомо неправильный интервал, т.е. интервал в котором нет ни одного счета-фактуры, либо где верхняя граница меньше чем нижняя. Отрицательный результат теста означает ошибку при вводе или обработки интервала.

Главным модулем является модуль регистрации счетов-фактур. Он должен отражать и давать возможность корректировать информацию, при этом не путать счета выданные и полученные. Также обходимо предусмотреть возможность различной фильтрации и поиска данных. Исходя из этого, тестирование можно разбить на две часть, это основную (отражение и корректировка) и дополнительную (фильтрация и поиск).

Так как пустой (т.е. без единого товара) счет-фактуры не имеет смысла, необходимо предусмотреть проверку на наличие хотя бы одного товара. Также товар без цены или количества не имеет смысла.

Первый тест на правильность отражения информации. Программа должна отражать именно ту информацию, которая находится в базе данных, в том числе отражать только те товары, которые относятся к базе данных. Для локализации ошибки, сначала тестируется логика работы базы данных и вспомогательных представлений V\_Articles и V\_Facture. Так как в таблице товары хранится только информация о цене, количестве и ссылка на налоги, а в таблице журнала счетов вообще отсутствует информация о сумме счета, для удобства работы было созданы эти представления. Для проверки правильности функционирования программы необходимо проводить визуальное сравнение экрана и содержимого базы данных. Отрицательный результат теста обозначает ошибку в программе при выборе информации из базы данных или при формировании запроса к ней.

Второй тест для подпрограммы журнала счетов ориентирован в первую очередь на правильность вывода информации. Для этого вводится заведомо неправильная информация: дубликат первичного ключа (идентификатор счета плюс номер товара), отсутствие клиента (внешний ключ) или товара, отсутствие или пустые поля, которые не могут быть пустыми. Отрицательный результат теста означает либо ошибку в подпрограмме, при формировании запроса и выдачи информации, либо в логике базы данных. Последнее можно проверить при помощи стандартной программы SQL Query, поставляемой вместе SQL Server.

Третий тест на правильность отработки фильтрации и поиска. Поиск осуществляется при помощи встроенного в Delphi поиска, поэтому необходимо проверить только правильность передачи аргументов. Фильтрация информации разделена на три компонента: фильтрация по типу счета-фактуры (невидимый для пользователя компонент), фильтрация по дате счета-фактуры и расширенная фильтрация в виде SQL предложения. Для проверки фильтрации информации по дате, необходимо варьировать параметрами даты, и визуально проверять правильность отражения в программе.

Комплексная отладка всех модулей дополнительно проверяет логику работы базы данных, т.к. модули программы функционально независимы.

Первый тест на удаление клиента, на которого уже заведен счет-фактуры. В таком случае программа не отработает ошибку, но ошибку должен выдать SQL Server. Отрицательный результат свидетельствует о нарушении в логике базы данных, в частности в триггере удаления клиента.

Второй тест на каскадное удаление счета-фактуры и товаров связанных с ним. При неправильном удалении могут возникнуть товары-демоны, которые не связаны не с одним счетом-фактуры.

Третий тест на каскадное обновление справочника налогов. При изменении информации в справочнике налогов, в частности изменение идентификатора налога, автоматически должны изменятся все идентификаторы налога в таблице товаров. Отрицательный результат свидетельствует о нарушении в триггере изменения справочника налогов.

Сетевое тесторование программы может выявить ошибки совместного изменения данных. Ошибки могут возникнуть, например, при удалении записи одним пользователем и обращении к ней другим. Устранение подобных ошибок берет на себя SQL Server, которых при записи быдаст сообщение об ошибке.

# Заключение

Заданием являлось создание клиентских частей под SQL базу данных для операционных систем Windows’95 и Windows NT Workstation.

В данной работе был выполнен обзор систем управления базами данных (СУБД), реализованных на технологии «клиент-сервер».

Далее было проведено исследование предметной области. По результатам исследования был сделан вывод, что область бухгалтерского учета, а именно учет счетов-фактур, на технологии «клиент-сервер» отдельно не реализована.

Затем была проанализирована существующая программа учета счетов-фактур фирмы «ИНФИН» и составлен проект переноса структуры базы данных для работы в архитектуре «клиент-сервер». Также был составлен алгоритм переноса необходимых для дальнейшей работы данных.

Для разработки программного обеспечения, платформой для создания серверной части SQL базы данных бала выбрана операционная система сервера Microsoft Windows NT Server 4.0 и система управления базами данных Microsoft SQL Server 6.5. Для разработки пользовательского интерфейса был выбран язык программирования Borland Delphi.

Был разработан и отлажен клиентский интерфейс к SQL базе данных под операционные системы Windows’95 и Windows NT. Разработанное программное обеспечение общим объемом занимает около 2,5 тысяч строк программного кода.

Тестирование разработанной программы проводилось по-модульно и комплексно. Затем была проведена отладка программы в реальных условиях работы, когда с базой данных работают несколько пользователей на разных станциях сети.

Разработанная на технологии «клиент-сервер» программа позволяет ее эксплуатировать как в сетевых условиях, так и на локальной машине, без изменений в структуре базы данных и программы.

Сейчас программа успешно функционирует на предприятии, заменив старую.

# Литература

1. С.Орлов. Windows NT 5.0: что на горизонте?.LAN Спец. выпуск к #5.1997
2. К.Пьянзин. Сетевые ОС в гетерогенной среде. LAN Magazine/ русское издание #7/96
3. Ф.Зубанов. Выбор «профи». М: Издательский отдел «Русская Редакция». 1996
4. К.Стинсон. Эффективная работа в Windows 95. С-П: ПИТЕР. 1997
5. С.Дунаев. UNIX SYSTEM V. Release 4.2. M: Диалог-МИФИ.1995
6. К.Дейта. Введение в системы баз данных. М:Наука.1980
7. Дж.Ульман. Основы систем баз данных.М:Финансы и статистика.1983
8. А.Шуленин. Microsoft SQL Server.СУБД #1/97
9. И.Игнатович. Семейство реляционных баз данных IBM DB2. СУБД #2/97
10. О.Твердова. СУБД Progress. СУБД #2/97
11. С.Бобровски. Oracle7 и вычисления клиент/сервер. М: ЛОРИ. 1996
12. В.Шнитман. Серверы баз данных: проблемы оценки конфигурации системы. СУБД #5-6/96
13. А.Тандоев. Sybase SQL Anywhere Pofessional. СУБД #1/97
14. А.Тандоев. Архитектура Sybase System 11. СУБД #2/96
15. М.Грабер. Введение в SQL. М: ЛОРИ. 1994
16. Р.Ахаян, А.Горев, С.Макашарипов. Эффективная работа с СУБД. С-П: ПИТЕР. 1997
17. К.Маркелов. «Классические» стратегии авоматизации в период нестабильность рынка АБС. Банк и Технологии #3/97
18. К.Маркелов. Опросы Форума как отражение на рынке АБС. Банк и Технологии #4/97
19. А.Сень, Ю.Юшков. Телекоммуникации в банковских системах. Банковские технологии #6/96
20. Дж.Матчо, Д.Фолкнер. Delphi. М: БИНОМ. 1995
21. Т.Сван. Секреты 32-разрядного програмирования в Delphi. К: Диалектика. 1997