

**Современный  
Гуманитарный  
Университет**

**Дистанционное образование**

---

Рабочий учебник

Фамилия, имя, отчество \_\_\_\_\_

Факультет \_\_\_\_\_

Номер контракта \_\_\_\_\_

## **СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ**

ЮНИТА 3

### **СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ СЕТЕЙ ЭВМ**

**МОСКВА 2000**

Разработано А.В. Карпейкиным, к.т.н., с.н.с.

Рекомендовано Министерством  
общего и профессионального  
образования Российской Федерации  
в качестве учебного пособия для  
студентов высших учебных заведений

## **КУРС: СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ**

Юнита 1. Операционные системы.

Юнита 2. Системное программное обеспечение процессов разработки и сопровождения программных комплексов.

Юнита 3. Системное программное обеспечение сетей ЭВМ.

### **ЮНИТА 3**

Рассмотрены распределенные базы данных и технология “клиент-сервер”, средства доступа к ресурсам глобальных сетей и средства разработки приложений в сетях ЭВМ, а также прикладные интерфейсы. Рассмотрены основные сетевые операционные системы и на примере операционной системы Windows NT вопросы, связанные со службой каталогов, файловой системой, администрированием и мониторингом, безопасностью и отказоустойчивостью, а также работа в глобальных сетях.

Для студентов Современного Гуманитарного Университета

Юнита соответствует образовательной профессиональной программе № 1

## ОГЛАВЛЕНИЕ

ДИДАКТИЧЕСКИЙ ПЛАН .....	5
ЛИТЕРАТУРА .....	6
ПЕРЕЧЕНЬ УМЕНИЙ .....	7
ТЕМАТИЧЕСКИЙ ОБЗОР .....	8
1. Системное ПО разработки и сопровождения приложений в сетях ЭВМ .....	8
1.1. Технология «клиент-сервер» .....	8
1.1.1. Архитектуры баз данных .....	8
1.1.2. Архитектура «клиент-сервер» .....	11
1.1.3. SQL-сервер Borland InterBase и его основные компоненты .....	15
1.1.4. Многозвенная архитектура «клиент-сервер» .....	18
1.2. Распределенные базы данных .....	19
1.2.1. Понятие распределенных БД .....	19
1.2.2. Целостность данных .....	22
1.2.3. Прозрачность расположения .....	23
1.2.4. Обработка распределенных запросов .....	24
1.2.5. Межоперабельность .....	24
1.2.6. Технология тиражирования данных .....	25
1.2.7. Распределенные системы .....	27
1.3. Средства доступа к ресурсам глобальных сетей .....	28
1.3.1. Характеристика основных информационных ресурсов Интернет .....	28
1.3.2. Принципы функционирования Интернет .....	31
1.3.3. Технология World Wide Web .....	35
1.3.4. Программы-клиенты WWW .....	38
1.3.5. Стратегия поиска информации в Интернет .....	39
1.3.6. Электронная почта в Интернет .....	42
1.4. Средства разработки приложений в сетях ЭВМ .....	47
1.5. Интерфейсы .....	51
2. Операционные системы сетей ЭВМ .....	54
2.1. Основные сетевые ОС .....	54
2.2. Windows NT .....	61
2.2.1. Архитектурные модули Windows NT .....	63
2.2.2. Уровень аппаратных абстракций .....	64
2.2.3. Ядро .....	65
2.2.4. Исполняющая система Windows NT .....	66
2.2.5. Подсистемы среды .....	77

2.3. Файловая система . . . . .	83
2.3.1. Файловая система FAT . . . . .	83
2.3.2. Файловая система HPFS . . . . .	85
2.3.3. Файловая система NTFS . . . . .	86
2.3.4. Целостность и восстановимость данных в файловых системах . . . . .	89
2.4. Служба каталогов . . . . .	92
2.5. Безопасность и отказоустойчивость . . . . .	94
2.5.1. Модель безопасности Windows NT . . . . .	94
2.5.2. Механизмы отказоустойчивости Windows NT . . . . .	106
2.6. Администрирование и мониторинг . . . . .	109
2.6.1. Администрирование Windows NT . . . . .	109
2.6.2. Мониторинг Windows NT . . . . .	113
2.7. Работа в глобальных сетях . . . . .	124
2.7.1. Сетевые средства . . . . .	124
2.7.2. Средства BackOffice . . . . .	127
ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ . . . . .	132
ТРЕНИНГ УМЕНИЙ . . . . .	133
ГЛОССАРИЙ*	

---

\* Глоссарий расположен в середине учебного пособия и предназначен для самостоятельного заучивания новых понятий.

## **ДИДАКТИЧЕСКИЙ ПЛАН**

**Системное ПО разработки и сопровождения приложений в сетях ЭВМ.** Технология “клиент-сервер”. Распределенные базы данных. Средства доступа к ресурсам глобальных сетей. Средства разработки приложений в сетях ЭВМ. Интерфейсы

**Операционные системы сетей ЭВМ.** Основные сетевые ОС. Windows NT. Служба каталогов. Файловая система. Администрирование и мониторинг. Безопасность и отказоустойчивость. Работа в глобальных сетях.

## **ЛИТЕРАТУРА**

### **Базовая**

- \*1. Острейковский В.А., Информатика. – М.: Высшая школа, 1999.
- \*2. Ресурсы Windows NT. Microsoft Press, 1996.

### **Дополнительная**

- 3. Корпоративные технологии. Учебный курс/ Пер. с англ. – М.: Издательский отдел «Русская редакция», – 1998.
- 4. Блэк Ю., Сети ЭВМ: протоколы, стандарты, интерфейсы. – М.: Мир, 1990.
- 5. Верлань А.Ф., Широчкин В.П., Информатика и ЭВМ. Киев: Техника, 1987.

---

Примечание. Знаком (\*) отмечены работы, на основе которых составлен тематический обзор.

### ПЕРЕЧЕНЬ УМЕНИЙ

№ п/п	Умение	Алгоритмы
1.	Написание SQL запроса на выборку данных из БД	<ol style="list-style-type: none"> <li>1. Определить БД для выборки данных.</li> <li>2. Определить поля для выборки данных.</li> <li>3. Определить условия выборки данных.</li> <li>4. Полностью написать SQL запрос.</li> </ol>
2.	Определение и использование протокола, имени домена и адреса узла в домене	<ol style="list-style-type: none"> <li>1. Считать из браузера полный адрес узла.</li> <li>2. По адресу определить используемый протокол, имя домена или адрес узла в домене.</li> </ol>
3.	Установка прав доступа к дискам, томам, каталогам и файлам (только для файловой системы NTFS)	<ol style="list-style-type: none"> <li>1. Выбрать (выделить) требуемые диски, тома, каталоги и/или файлы.</li> <li>2. Открыть окно «Свойства» выбранных объектов.</li> <li>3. Выбрать закладку «Безопасность».</li> <li>4. В панели «Разрешения» нажать кнопку «Разрешения».</li> <li>5. Установить права доступа для нужных групп пользователей.</li> </ol>
4.	Установка прав аудита для групп пользователей (только для файловой системы NTFS)	<ol style="list-style-type: none"> <li>1. Выбрать (выделить) требуемые диски, тома, каталоги и/или файлы.</li> <li>2. Открыть окно «Свойства» выбранных объектов.</li> <li>3. Выбрать закладку «Безопасность».</li> <li>4. В панели «Аудит» нажать кнопку «Аудит».</li> <li>5. Установить параметры аудита для нужных групп пользователей.</li> </ol>

## ТЕМАТИЧЕСКИЙ ОБЗОР\*

### 1. СИСТЕМНОЕ ПО РАЗРАБОТКИ И СОПРОВОЖДЕНИЯ ПРИЛОЖЕНИЙ В СЕТЯХ ЭВМ

#### 1.1. Технология «клиент-сервер»

##### 1.1.1. Архитектуры баз данных

Общий состав средств, необходимых для работы с базами данных (БД), приведен на рис. 1.1. Согласно этой общей схеме имеется цепочка *Приложение* → *Машина БД* → *БД*. Приложение является собственно программой для работы с БД, Машина БД – это набор сервисных библиотек, выполняющий действия по доступу к БД (с учетом типа БД и места их расположения) и проверке их правильности. Существует несколько типов машин БД в зависимости от используемых типов БД – например, фирмы Borland (BDE – Borland Database Engine).

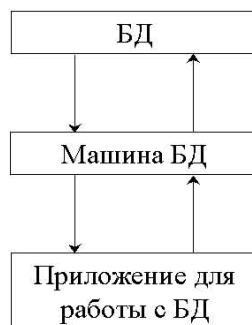


Рис. 1.1

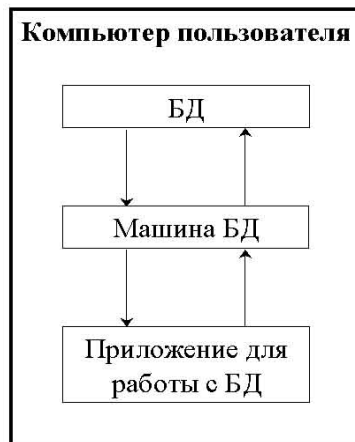
Между тем, местоположение отдельных элементов данной цепочки определяется используемой архитектурой. В настоящее время выделяют следующие архитектуры БД: локальные БД; архитектура «файл-сервер»; архитектура «клиент-сервер»; многозвенная архитектура.

Под **сервером** понимается компьютер, предоставляющий свои ресурсы для совместного использования в сети. **Клиент** – это компьютер, осуществляющий доступ к ресурсам другого компьютера, предоставленным в совместное использование. **Ресурсом** является любая часть компьютерной системы (диск, принтер, память и т.д.), которая может быть предоставлена программе во время работы.

---

\* Жирным шрифтом выделены новые понятия, которые необходимо усвоить. Знание этих понятий будет проверяться при тестировании.

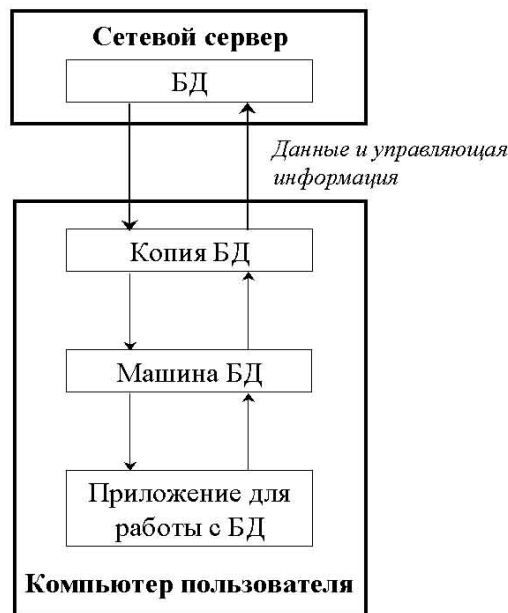
При работе с локальными базами данных (рис. 1.2) сами БД расположены на том же компьютере, что и приложения, осуществляющие доступ к ним. Работа с БД идет в однопользовательском режиме, а всю ответственность за выполнение запросов к БД и поддержание ее целостности несет приложение.



**Рис. 1.2**

Для осуществления режима коллективного доступа к данным (многопользовательского режима работы) была предложена архитектура «файл-сервер» (рис. 1.3). При работе в такой архитектуре БД и приложение расположены на файловом сервере сети (например, Novell NetWare). В этом случае возможна многопользовательская работа с одной и той же БД, когда каждый пользователь со своего компьютера запускает приложение, расположенное на сетевом сервере (это необходимо для соблюдения всеми пользователями принятых и одинаковых правил работы с БД). Тогда на компьютере пользователя запускается копия приложения. По каждому запросу к БД из приложения все данные из таблиц БД передаются на компьютер пользователя, причем независимо от того, сколько реально нужно данных для выполнения запроса.

Каждый пользователь имеет на своем компьютере локальную копию данных, время от времени обновляемых из реальной БД, расположенной на сетевом сервере. При этом изменения, которые каждый пользователь вносит в БД, могут быть до определенного момента неизвестны другим пользователям, что делает актуальной задачу систематического обновления данных на компьютере пользователя из реальной БД. Другой актуальной задачей является блокирование записей, которые изменяются одним из пользователей (это необходимо для того, чтобы



**Рис. 1.3**

в это время другой пользователь не внес изменения в эти же данные).

В архитектуре «файл-сервер» вся тяжесть выполнения запросов к БД и управления целостностью БД ложится на приложение пользователя. БД на сервере являются пассивным источником данных. Кардинальных отличий с точки зрения архитектуры между однопользовательской архитектурой и архитектурой «файл-сервер» нет. И в том, и в ином случае применяются локальные СУБД (приложение – машина БД). Сама БД в этом случае представляет собой набор таблиц, индексных файлов и пр. Характерными примером использования архитектуры «файл-сервер» являлись системы типа FoxPro, Clipper, Paradox и др. В ходе эксплуатации таких систем были выявлены *общие недостатки файл-серверного подхода при обеспечении многопользовательского доступа к БД*. Они состоят в следующем:

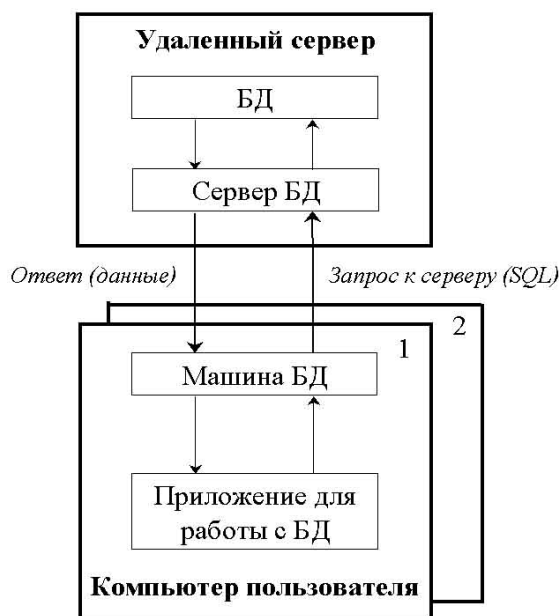
- Вся тяжесть вычислительной нагрузки при доступе к БД ложится на приложение клиента, что является следствием принципа обработки информации в системах «файл-сервер» – при выдаче запроса на выборку информации из таблицы на клиентское место копируется вся таблица и только затем осуществляется выборка нужной информации.
- Локальные СУБД используют так называемый навигационный подход, ориентированный на работу с отдельными записями.

- Не оптимально расходуются ресурсы клиентского компьютера и сети. Например, если в результате запроса необходимо получить 5 записей из таблицы объемом 10000 записей, все 10000 записей будут скопированы с файл-сервера на клиентский компьютер. В результате возрастает сетевой трафик и увеличиваются требования к аппаратным мощностям клиентского компьютера. Заметим, что потребности в постоянном увеличении вычислительных мощностей клиентского компьютера обуславливаются не только развитием ПО как такового, но и возрастанием обрабатываемых объемов информации.
- Обеспечение целостности БД производится из приложений. Это потенциальный источник ошибок, нарушающих физическую и логическую целостность БД, поскольку различные приложения могут производить контроль целостности по-разному, взаимно-исключающими способами, или не производить такого контроля вовсе (гораздо эффективнее управлять БД из единого места и по единым законам). Поэтому безопасность при работе в архитектуре «файл-сервер» невысока (по сути, осуществляется только на уровне файловой системы) и всегда присутствует элемент неопределенности.
- Секретность и конфиденциальность при работе с БД в архитектуре «файл-сервер» обеспечить также тяжело – любой, кто имеет доступ в каталог сетевого сервера или клиента, где хранится БД, может изменить таблицы БД любым образом, копировать их, заменять и т.п.

### 1.1.2. Архитектура «клиент-сервер»

Приведенные недостатки решаются при переводе приложений из архитектуры «файл-сервер» в архитектуру «клиент-сервер», которая знаменует собой следующий этап в развитии СУБД. Характерной особенностью архитектуры «клиент-сервер» является перенос вычислительной нагрузки на *сервер БД (SQL-сервер)* и максимальная разгрузка приложения клиента от вычислительной работы, а также существенное укрепление безопасности данных – как от злонамеренных, так и просто ошибочных изменений. То есть архитектура «клиент-сервер» разделяет функции приложения пользователя (называемого клиентом) и сервера (рис. 1.4). Приложение-клиент формирует запрос к серверу, на котором расположена БД, на структурном языке запросов SQL (Structured Query Language), являющемся промышленным стандартом в мире реляционных БД. Удаленный сервер принимает запрос и переадресует его SQL-серверу БД. SQL-сервер – специальная программа, управляющая удаленной базой данных. SQL-сервер обеспечивает интерпретацию запроса, его выполнение в базе данных, формирование результата выполнения запроса и выдачу его приложению-клиенту. При этом

ресурсы клиентского компьютера не участвуют в физическом выполнении запроса. Клиентский компьютер лишь отправляет запрос к серверной БД и получает результат, после чего интерпретирует его необходимым образом и представляет пользователю. Так как клиентскому приложению посылается результат выполнения запроса, по сети «путешествуют» только те данные, которые необходимы клиенту. В итоге снижается нагрузка на сеть. Поскольку выполнение запроса происходит там же, где хранятся данные (на сервере), нет необходимости в пересылке больших пакетов данных. Кроме того, SQL-сервер, если это возможно, оптимизирует полученный запрос таким образом, чтобы он был выполнен в минимальное время с наименьшими накладными расходами. Все это повышает быстродействие системы и снижает время ожидания результата запроса.



**Рис. 1.4**

При выполнении запросов сервером существенно повышается степень безопасности данных, поскольку правила целостности данных определяются в базе данных на сервере и являются едиными для всех приложений, использующих эту БД. Таким образом, исключается возможность определения противоречивых правил поддержания целостности. Мощный аппарат транзакций, поддерживаемый SQL-серверами, позволяет исключить одновременное изменение одних и

тех же данных различными пользователями и предоставляет возможность откатов к первоначальным значениям при внесении в БД изменений, закончившихся аварийно.

Таким образом, функциями приложения-клиента являются: 1 – посылка запросов к серверу; 2 – интерпретация результатов запросов, полученных от сервера, и представление их пользователю в требуемой форме; 3 – реализация интерфейса пользователя.

SQL-сервер – это программа, расположенная на компьютере сетевого сервера. SQL-сервер должен быть загружен на момент принятия запроса от клиента. Функциями сервера БД являются: 1 – прием запросов от приложений-клиентов, интерпретация запросов, выполнение запросов в БД, отправка результата выполнения запроса приложению-клиенту; 2 – управление целостностью БД, обеспечение системы безопасности, блокировка неверных действий приложений-клиентов; 3 – хранение бизнес-правил, часто используемых запросов в уже интерпретированном виде; 4 – обеспечение одновременной безопасной и отказоустойчивой многопользовательской работы с одними и теми же данными.

В архитектуре «клиент-сервер» используются так называемые «удаленные» (или промышленные) СУБД. Промышленными они называются из-за того, что именно СУБД этого класса могут обеспечить работу информационных систем масштаба среднего и крупного предприятия, организации, банка. Локальные СУБД предназначены для однопользовательской работы или для обеспечения работы информационных систем, рассчитанных на небольшие группы пользователей.

К разряду промышленных СУБД принадлежат Oracle, Gupta, Informix, Sybase, MS SQL Server, DB2, InterBase и ряд других. Как правило, SQL-сервер управляется отдельным сотрудником или группой сотрудников (администраторы SQL-сервера). Они управляют физическими характеристиками баз данных, производят оптимизацию, настройку и переопределение различных компонентов БД, создают новые БД, изменяют существующие и т.д., а также выдают привилегии (разрешения на доступ определенного уровня к конкретным БД, SQL-серверу) различным пользователям.

Кроме этого, существует отдельная категория сотрудников, называемых администраторами баз данных. Как правило, это администраторы сервера, разработчики БД или пользователи, имеющие привилегии на создание, изменение, настройку оптимальных параметров отдельных серверных БД. Администраторы БД также отвечают за предоставление прав на разноуровневый доступ к сопровождаемым ими БД для других пользователей.

*Преимущества архитектуры «клиент-сервер»:*

- большинство вычислительных процессов происходит на сервере; таким образом, снижаются требования к вычислительным мощностям компьютера клиента;

- снижается сетевой трафик за счет посылки сервером клиенту только тех данных, которые он запрашивал; например, если необходимо сделать из таблицы объемом 10 000 записей выборку, результатом которой будут всего две записи, сервер выполнит запрос и перешлет клиенту набор данных (НД) из двух записей;
- упрощается наращивание вычислительных мощностей в условиях развития программного обеспечения и возрастания объемов обрабатываемых данных. Проще и дешевле усилить мощности на сетевом сервере или полностью заменить сервер на более мощный, нежели наращивать мощности или полностью заменять 100-500 клиентских компьютеров;
- БД на сервере представляет собой, как правило, единый файл, в котором содержатся таблицы БД, ограничения целостности и другие компоненты БД. Взломать такую БД, даже при наличии умысла, тяжело; значительно увеличивается защищенность БД от ввода неправильных значений, поскольку сервер БД проводит автоматическую проверку соответствия вводимых значений наложенным ограничениям и автоматически выполняет необходимые бизнес-правила. Кроме того, сервер отслеживает уровни доступа для каждого пользователя и блокирует осуществление попыток выполнения неразрешенных для пользователя действий, например, изменения или просмотр таблиц; все это позволяет говорить о значительно более высоком уровне обеспечения безопасности БД и ссылочной и смысловой целостности информации;
- сервер реализует управление транзакциями и предотвращает попытки одновременного изменения одних и тех же данных; различные уровни изоляции транзакций позволяют определить поведение сервера при возникновении ситуаций одновременного изменения данных;
- безопасность системы возрастает за счет переноса большей части бизнес-правил на сервер; падает удельный вес противоречащих друг другу бизнес-правил в клиентских приложениях, выполняющих разные действия над БД. Определить такие противоречивые бизнес-правила в приложениях клиента все еще можно, однако намного труднее их выполнить ввиду автоматического отслеживания сервером БД правильности данных.

Для реализации архитектуры применяют так называемые промышленные («удаленные») СУБД, такие как Borland InterBase, Oracle, Informix, Sybase, DB2, MS SQL Server. Рассмотрим более подробно сервер Borland InterBase.

### 1.1.3. SQL-сервер Borland InterBase и его основные компоненты

SQL-сервер Borland InterBase является промышленной СУБД, предназначенной для хранения и выдачи больших объемов данных при использовании архитектуры «клиент-сервер» в условиях одновременной работы с БД множества клиентских приложений. Масштаб информационной системы при этом произволен – от системы уровня рабочей группы (под управлением Novell Netware или Windows NT на базе IBM PC) до системы уровня большого предприятия (на базе серверов IBM, Hewlett-Packard, SUN).

Рассмотрим ряд компонентов InterBase, использование которых обеспечивает максимальную вычислительную разгрузку клиентского приложения и гарантирует высокую безопасность и целостность информации.

Для задания ссылочной и смысловой целостности в БД определяются:

- отношения подчиненности между таблицами БД путем определения *первичных* (PRIMARY) *ключей* у родительских и *внешних* (FOREIGN) *ключей* у дочерних таблиц;
- ограничения на значения отдельных столбцов путем определения *ограничений* (CONSTRAINT); при этом условия ограничений могут быть весьма разнообразны – от требования попадания значения в определенный диапазон или соответствия маске до определенного отношения с одной или несколькими записями из другой таблицы (или многих таблиц) БД;
- бизнес-правила при помощи *триггеров* (TRIGGER) – подпрограмм, автоматически выполняемых сервером до или (и) после события изменения записи в таблице БД;
- уникальные значения нужных полей путем создания и использования *генераторов* (GENERATOR).

Для ускорения работы клиентских приложений с удаленной БД могут быть определены хранимые процедуры (STORED PROCEDURE), которые представляют собой подпрограммы, принимающие и возвращающие параметры и могущие выполнять запросы к БД, условные ветвления и циклическую обработку. Хранимые процедуры пишутся на специальном алгоритмическом языке. В хранимых процедурах программируются часто повторяемые последовательности запросов к БД. Текст процедур хранится на сервере в откомпилированном виде. Преимущества в использовании хранимых процедур очевидны:

- отпадает необходимость синтаксической проверки каждого запроса и его компиляции перед выполнением, что убыстряет выполнение запросов;
- отпадает необходимость реализации в приложении запросов, определенных в теле хранимых процедур;

- увеличивается скорость обработки транзакций, так как вместо подчас длинного SQL-запроса по сети передается относительно короткое обращение к хранимой процедуре.

В составе записи БД могут определяться *BLOB-поля* (Binary Large Object, большой двоичный объект), предназначенные для хранения больших объемов данных в виде последовательности байтов. Таким образом могут храниться текстовые и графические документы, файлы мультимедиа, звуковые файлы и т.д. Интерпретация *BLOB-поля* выполняется в приложении, однако разработчик может определить так называемые BLOB-фильтры для автоматического преобразования содержимого *BLOB-поля* к другому виду.

InterBase дает возможность использовать *определяемые пользователем функции* (User Defined Function, *UDF*), в которых могут реализовываться функциональности, отсутствующие в стандартных встроенных функциях InterBase (вычисление максимума, минимума, среднего значения, преобразование типов и приведение букв к заглавным). Например, в UDF можно реализовать извлечение из значения даты номера дня, года; определение длины символьного значения; усечение пробелов; разные математические алгоритмы и другое. Функция пишется на любом алгоритмическом языке, позволяющем разрабатывать DLL (библиотеки динамического вызова), например на Object Pascal.

InterBase может посылать уведомления клиентским приложениям о наступлении какого-либо *события* (EVENT). Одновременно работающие приложения могут обмениваться сообщениями через сервер БД, вызывая хранимые процедуры, в которых реализована инициация нужного события.

Для обеспечения скорости выполнения запросов и снятия с клиентского приложения или необходимости выдавать такие запросы в БД можно определить *виртуальные таблицы* (или *просмотры*, VIEW), в которых объединяются записи из одной или более таблиц, соответствующих некоторому условию. Работа с просмотром из клиентского приложения ничем не отличается от работы с обычной таблицей. Просмотры могут быть изменяемыми и не допускающими внесения в них изменений.

Для доступа к БД используется утилита *Windows Interactive SQL (WISQL)*. Она работает с БД напрямую через InterBase API, минуя BDE. В WISQL можно выдавать любые запросы, будь то создание БД, таблиц, изменение структуры данных, извлечение данных из БД или их изменение, а также назначение прав доступа к информации для отдельных пользователей.

Для управления SQL-сервером в целом и отдельными БД, в частности, используется утилита *InterBase Server Manager*. Здесь можно определять параметры SQL-сервера, производить сохранение, восстановление БД, сборку «мусора», определять новых пользователей, их пароли и т.д.

Для просмотра БД, работы с таблицами, индексами, доменами, ограничениями и др. могут использоваться утилиты *Database Desktop* (весьма ограниченно) и *SQL Explorer*.

Для просмотра и анализа реальных процессов, происходящих на сервере при реализации пользовательского запроса, используется утилита *SQL Monitor*.

В России InterBase используется с 1993 г., но интерес к этому SQL-серверу возрос только в последнее время, в связи с включением его локальной (или 4-х пользовательской версии) в состав Delphi Client/Server Suite. Внимание разработчиков БД и приложений InterBase привлек, во-первых, потому, что это «родной» продукт Borland (а средства разработки приложений этой компании давно зарекомендовали себя с положительной стороны), во-вторых, потому, что InterBase весьма прост в установке,

Таблица 1.1

Характеристика	Значение
Максимальный размер одной БД	Рекомендуется не выше 10 Гбайт. Однако известны случаи объема одной БД в 10-20 Гбайт
Максимальное число таблиц в одной БД	65 536
Максимальное число полей (столбцов) в одной таблице	1000
Максимальное число записей в одной таблице	Не ограничено
Максимальная длина записи	64 Кбайт (не считая полей BLOB)
Максимальная длина поля	32 Кбайт (кроме полей BLOB)
Максимальная длина поля BLOB	Не ограничена
Максимальное число индексов в БД	65 536
Максимальное число полей в индексе	16
Максимальное число вложенностей SQL-запроса	16
Максимальный размер хранимой процедуры или триггера	48 Кбайт
Максимальное количество UDF в базе данных	Длина имени UDF – не более 31 символа. Каждый UDF должен иметь уникальное имя. Максимальное число UDF ограничивается только требованием уникальности имени

настройке и – главное – в администрировании по сравнению с другими SQL-серверами, и, в-третьих, потому, что он обладает прекрасными функциональными возможностями. В таблице 1.1 приведены некоторые технические характеристики сервера.

Кроме того, локальный InterBase может использоваться для отладочных целей. После того, как приложение отлажено на локальной версии SQL-сервера, происходит масштабирование приложения (*upsizing*). БД переносится на сетевой сервер, а изменения в клиентских приложениях при этом минимальны – необходимо изменить псевдоним БД и, может быть, скорректировать некоторые параметры соединения приложения с сервером.

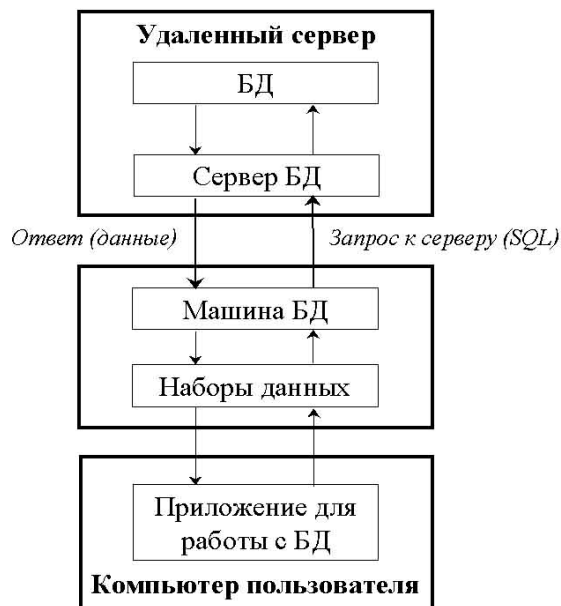
При переносе приложений, ранее разработанных для применения в архитектуре «файл-сервер», требуется не только частично или полностью переписывать приложения клиентов, но и преобразовывать локальную БД в серверную. Для этого под управлением серверной СУБД (например, InterBase) создают БД на сервере, куда затем «перекачивают» данные из локальных СУБД, реализованных, например, с помощью Paradox. Основная проблема, встающая в этом случае, – несовместимость некоторых форматов данных или их отсутствие. При возникновении подобных проблем следует изучить вопросы совместимости типов данных локальной СУБД и выбранной серверной СУБД.

#### **1.1.4. Многозвенная архитектура «клиент-сервер»**

Развитие идей архитектуры «клиент-сервер» привело к появлению многозвенной архитектуры доступа к базам данных (в литературе ее также называют трехзвенной архитектурой, N-tier или multi-tier архитектурой). Архитектура «клиент-сервер» является двухзвенной. Первым звеном является приложение клиента, вторым – сервер БД и сама БД. В трехзвенной архитектуре наборы данных, бывшие ранее «собственностью» клиентских приложений, выделяются в отдельное звено, называемое сервером приложений (рис. 1.5).

Итак, модули данных в трехзвенной архитектуре «клиент-сервер» выделяются в отдельный «сервер приложений». Совместно с ним располагается BDE. Теперь, при изменении бизнес-правил, нет необходимости изменять приложения клиентов и обновлять их у всех пользователей-клиентов, как это было ранее, когда часть бизнес-правил хранилась в приложении клиента.

Сервер приложений разделяется несколькими клиентами. Он формирует запрос к удаленной БД (т.е. к SQL-серверу). На нем расположены реальные наборы данных (в двухзвенной архитектуре «клиент-сервер» располагавшиеся в приложении клиента). В клиентском приложении размещается «клиентский набор данных». Он представляет собой локальную копию данных с сервера приложений. Таким образом, все изменения, вносимые пользователем в данные при помощи



**Рис. 1.5**

клиентского приложения, вносятся в локальную копию НД. При обновлении удаленного НД клиентское приложение посылает серверу приложений только изменившиеся записи. Сервер приложений, в свою очередь, отсылает эти изменения SQL-серверу, который вносит их в удаленную БД.

Взаимодействие клиентского приложения (в многозвенном приложении называемого «тонким клиентом», thin client) с сервером приложений осуществляется при помощи так называемых брокеров данных.

## **1.2. Распределенные базы данных**

### **1.2.1. Понятие распределенных БД**

Распределенные базы данных невозможно рассматривать вне контекста более общей и более значимой темы распределенных информационных систем. На это повлияли процессы децентрализации и информационной интеграции, происходящие во всем мире. Россия, в силу своего географического положения и размеров, «обречена» на преимущественное использование распределенных систем. Это направление может успешно развиваться лишь при выполнении двух

главных условий – адекватном развитии глобальной сетевой инфраструктуры и применении реальных технологий создания распределенных информационных систем.

Под **распределенной базой данных** (Distributed DataBase – DDB) обычно подразумевают совокупность баз данных, включающую фрагменты из нескольких баз данных, которые располагаются на различных узлах сети компьютеров, и, возможно, управляются различными СУБД, оставаясь доступными для совместного использования в различных приложениях (рис. 1.6). Распределенная база данных выглядит с точки зрения пользователей и прикладных программ как обычная локальная база данных. В этом смысле слово «распределенная» отражает способ организации базы данных, но не внешнюю ее характеристику («распределенность» базы данных невидима извне).

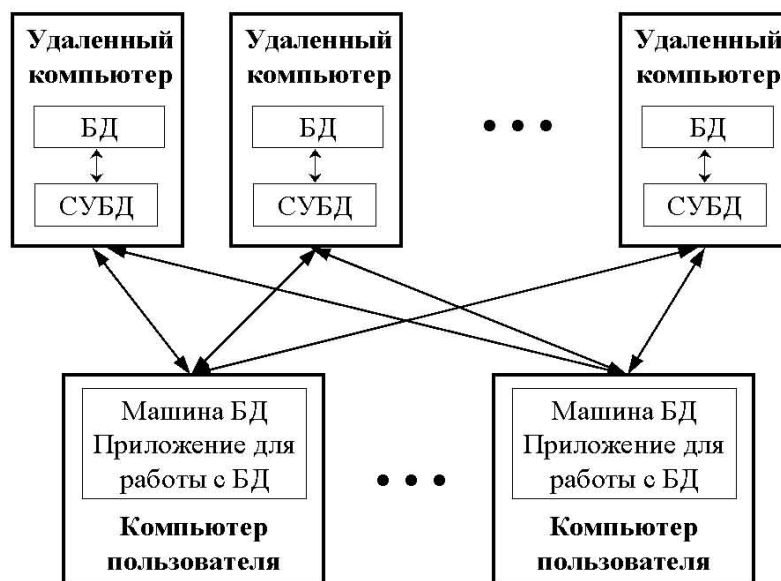


Рис. 1.6

Можно выделить 12 основных свойств или качеств идеальной DDB:

- **Локальная автономия.** Это качество означает, что управление данными на каждом из узлов **распределенной системы** (системы, состоящей из взаимосвязанных сетевых ЭВМ, позволяющей выполнять обработку в распределенной БД) выполняется локально. База данных, расположенная на одном из узлов, является неотъемлемым компонентом распределенной системы. Будучи фрагментом общего пространства данных, она

в то же время функционирует как полноценная локальная база данных: управление ею выполняется локально и независимо от других узлов системы.

- *Независимость узлов.* В идеальной системе все узлы равноправны и независимы, а расположенные на них базы являются равноправными поставщиками данных в общее пространство данных. База данных на каждом из узлов самодостаточна – она включает полный собственный словарь данных и полностью защищена от несанкционированного доступа.
- *Непрерывные операции.* Это качество можно трактовать как возможность непрерывного доступа к данным (известное «24 часа в сутки, семь дней в неделю») в рамках DDB вне зависимости от их расположения и вне зависимости от операций, выполняемых на локальных узлах.
- *Прозрачность расположения.* Это свойство означает полную прозрачность расположения данных. Пользователь, обращающийся к DDB, ничего не должен знать о реальном, физическом размещении данных в узлах информационной системы. Все операции над данными выполняются без учета их местонахождения. Транспортировка запросов к базам данных осуществляется встроенными системными средствами.
- *Прозрачная фрагментация.* Это свойство трактуется как возможность распределенного (то есть на различных узлах) размещения данных, логически представляющих собой единое целое. Существует фрагментация двух типов: горизонтальная и вертикальная. Первая означает хранение строк одной таблицы на различных узлах (фактически, хранение строк одной логической таблицы в нескольких идентичных физических таблицах на различных узлах). Вторая означает распределение столбцов логической таблицы по нескольким узлам.
- *Прозрачное тиражирование.* Тиражирование данных – это асинхронный (в общем случае) процесс переноса изменений объектов исходной базы данных в базы, расположенные на других узлах распределенной системы. В данном контексте прозрачность тиражирования означает возможность переноса изменений между базами данных средствами, невидимыми пользователю распределенной системы. Данное свойство означает, что тиражирование возможно и достигается внутрисистемными средствами.
- *Обработка распределенных запросов.* Это свойство DDB трактуется как возможность выполнения операций выборки над распределенной базой данных, сформулированных в рамках обычного запроса на языке SQL. То есть операцию выборки из DDB можно сформулировать с помощью тех же языковых средств, что и операцию над локальной базой данных.

- *Обработка распределенных транзакций.* Это качество DDB можно трактовать как возможность выполнения операций обновления распределенной базы данных, не разрушающее целостность и согласованность данных. Эта цель достигается применением двухфазового или двухфазного протокола фиксации транзакций (two-phase commit protocol), ставшего фактическим стандартом обработки распределенных транзакций. Его применение гарантирует согласованное изменение данных на нескольких узлах в рамках распределенной (или, как ее еще называют, глобальной) транзакции.
- *Независимость от оборудования.* Это свойство означает, что в качестве узлов распределенной системы могут выступать компьютеры любых моделей и производителей – от мэйнфреймов до «персоналок».
- *Независимость от операционных систем.* Это качество вытекает из предыдущего и означает многообразие операционных систем, управляющих узлами распределенной системы.
- *Прозрачность сети.* Доступ к любым базам данных может осуществляться по сети. Спектр поддерживаемых конкретной СУБД сетевых протоколов не должен быть ограничением системы с распределенными базами данных. Данное качество формулируется максимально широко – в распределенной системе возможны любые сетевые протоколы.
- *Независимость от баз данных.* Это качество означает, что в распределенной системе могут мирно сосуществовать СУБД различных производителей, и возможны операции поиска и обновления в базах данных различных моделей и форматов. DDB можно рассматривать как слабосвязанную сетевую структуру, узлы которой представляют собой локальные базы данных. Локальные базы данных автономны, независимы и самоопределены; доступ к ним обеспечивается СУБД, в общем случае от различных поставщиков. Связи между узлами – это потоки тиражируемых данных. Топология DDB варьируется в широком диапазоне – возможны варианты иерархии, структур типа «звезда» и т.д. В целом топология DDB определяется географией информационной системы и направленностью потоков тиражирования данных.

Далее рассмотрим некоторые наиболее важные свойства DDB.

### **1.2.2. Целостность данных**

В DDB поддержка целостности и согласованности данных, ввиду свойств 1-2 (см. выше), представляет собой сложную проблему. Ее решение – синхронное и согласованное изменение данных в нескольких локальных базах данных, составляющих DDB, – достигается приме-

нием протокола двухфазной фиксации транзакций. Если DDB однородна, то есть на всех узлах данные хранятся в формате одной базы и на всех узлах функционирует одна и та же СУБД, то используется механизм двухфазной фиксации транзакций данной СУБД. В случае же неоднородности DDB для обеспечения согласованных изменений в нескольких базах данных используют менеджеры распределенных транзакций. Это, однако, возможно, если участники обработки распределенной транзакции – СУБД, функционирующие на узлах системы, поддерживают XA-интерфейс, определенный в спецификации DTP консорциума X/Open. В настоящее время XA-интерфейс имеют CA-OpenIngres, Informix, Microsoft SQL Server, Oracle, Sybase.

Если в DDB предусмотрено тиражирование данных, то это сразу предъявляет дополнительные жесткие требования к дисциплине поддержки целостности данных на узлах, куда направлены потоки тиражируемых данных. Проблема в том, что изменения в данных иницируются как локально – на данном узле, – так и извне, посредством тиражирования. Неизбежно возникают конфликты по изменениям, которые необходимо отслеживать и разрешать.

### **1.2.3. Прозрачность расположения**

Это качество DDB в реальных продуктах должно поддерживаться соответствующими механизмами. Разработчики СУБД придерживаются различных подходов. Рассмотрим пример из Oracle. Допустим, что DDB включает локальную базу данных, которая размещена на узле в Лондоне. Создадим вначале ссылку (database link), связав ее с символическим именем (london\_unix), транслируемым в IP-адрес узла в Лондоне.

Теперь мы можем явно обращаться к базе данных на этом узле, запрашивая, например, в операторе SELECT таблицу, хранящуюся в этой базе. Очевидно, однако, что данный запрос, зависит от расположения базы данных, поскольку в нем явно использовали ссылку. Однако можно определить переменную (например, customer) и адрес (например, customer@london.com) как синонимы с помощью оператора CREATE SYNONYM customer FOR customer@london.com, и в результате можем написать полностью независимый от расположения базы данных запрос.

Задача решается с помощью оператора SQL CREATE SYNONYM, который позволяет создавать новые имена для существующих таблиц. При этом оказывается возможным обращаться к другим базам данных и к другим компьютерам.

Во многих СУБД задача управления именами объектов DDB решается путем использования глобального словаря данных, хранящего информацию о DDB: расположение данных, возможности других СУБД (если используются шлюзы), сведения о скорости передачи по сети с различной топологией и т.д.

#### 1.2.4. Обработка распределенных запросов

Выше уже упоминалось это качество DDB. Обработка распределенных запросов (Distributed Query – DQ) – задача, более сложная, нежели обработка локальных запросов, и она требует интеллектуального решения с помощью особого компонента – оптимизатора DQ. Обратимся к базе данных, распределенной по двум узлам сети. Таблица detail хранится на одном узле, таблица supplier – на другом. Размер первой таблицы – 10 000 строк, размер второй – 100 строк (множество деталей поставляется небольшим числом поставщиков). Допустим, что выполняется запрос:

```
SELECT detail_name, supplier_name, supplier_address  
FROM detail, supplier  
WHERE detail.supplier_number = supplier.supplier_number;
```

Данная конструкция операторов служит для выбора из БД данных, соответствующих определенным условиям. За оператором SELECT следует перечень полей из таблицы (таблиц) БД (названия БД, из которых необходимо выбрать данные, определяются следующим оператором – FROM). Если необходимо выбрать все поля из таблицы, то за оператором SELECT следует знак «\*». Условия выборки определяются оператором WHERE.

Результирующая таблица представляет собой объединение таблиц detail и supplier, выполненное по столбцу detail.supplier\_number (внешний ключ) и supplier.supplier\_number (первичный ключ).

Данный запрос – распределенный, так как затрагивает таблицы, принадлежащие различным локальным базам данных. Для его нормального выполнения необходимо иметь обе исходные таблицы на одном узле. Следовательно, одна из таблиц должна быть передана по сети. Очевидно, что это должна быть таблица меньшего размера, то есть таблица supplier. Следовательно, оптимизатор DQ запросов должен учитывать такие параметры, как, в первую очередь, размер таблиц, статистику распределения данных по узлам, объем данных, передаваемых между узлами, скорость коммуникационных линий, структуры хранения данных, соотношение производительности процессоров на разных узлах и т.д. От интеллекта оптимизатора DQ напрямую зависит скорость выполнения распределенных запросов.

#### 1.2.5. Межоперабельность

В контексте DDB межоперабельность означает две вещи. Во-первых, – это качество, позволяющее обмениваться данными между базами данных различных поставщиков. Как, например, тиражировать данные из базы данных Informix в Oracle и наоборот? Известно, что штатные

средства тиражирования в составе данной конкретной СУБД позволяют переносить данные в однородную базу. Так, средствами CA-Ingres/Replicator можно тиражировать данные только из Ingres в Ingres. Как быть в неоднородной DDB? Ответом стало появление продуктов, выполняющих тиражирование между разнородными базами данных.

Во-вторых, это возможность некоторого унифицированного доступа к данным в DDB из приложения. Возможны как универсальные решения (стандарт ODBC), так и специализированные подходы. Очевидный недостаток ODBC – недоступность для приложения многих полезных механизмов каждой конкретной СУБД, поскольку они могут быть использованы в большинстве случаев только через расширения SQL в диалекте языка данной СУБД, но в стандарте ODBC эти расширения не поддерживаются.

Специальные подходы – это, например, использование шлюзов, позволяющее приложениям оперировать над базами данных в «чужом» формате так, как будто это собственные базы данных. Вообще, цель шлюза – организация доступа к унаследованным (legacy) базам данных – служит для решения задач согласования форматов баз данных при переходе к какой-либо одной СУБД. Так, если компания долгое время работала на СУБД IMS и затем решила перейти на Oracle, то ей, очевидно, потребуется шлюз в IMS. Следовательно, шлюзы можно рассматривать как средство, облегчающее миграцию, но не как универсальное средство межоперабельности в распределенной системе. Вообще, универсального рецепта решения задачи межоперабельности в этом контексте не существует – все определяется конкретной ситуацией, историей информационной системы и массой других факторов. DDB конструирует архитектор, имеющий в своем арсенале отработанные интеграционные средства, которых на рынке сейчас очень много.

### **1.2.6. Технология тиражирования данных**

Принципиальная характеристика тиражирования данных (Data Replication – DR) заключается в отказе от физического распределения данных. Суть DR состоит в том, что любая база данных (как для СУБД, так и для работающих с ней пользователей) всегда является локальной; данные размещаются локально на том узле сети, где они обрабатываются; все транзакции в системе завершаются локально.

Тиражирование данных – это асинхронный перенос изменений объектов исходной базы данных в базы, принадлежащие различным узлам распределенной системы. Функции DR выполняет, как правило, специальный модуль СУБД – сервер тиражирования данных, называемый репликатором (так устроены СУБД CA-OpenIngres и Sybase). В Informix-OnLine Dynamic Server репликатор встроен в сервер, в Oracle7 для использования DR необходимо приобрести дополнительно к Oracle7 DBMS опцию Replication Option.

В качестве базиса для тиражирования выступает транзакция к базе данных. В то же время возможен перенос изменений группами транзакций, периодически или в некоторый момент времени, что дает возможность исследовать состояние принимающей базы на определенный момент времени.

Детали тиражирования данных полностью скрыты от прикладной программы; ее функционирование никак не зависит от работы репликатора, который целиком находится в ведении администратора базы данных. Следовательно, для переноса программы в распределенную среду с тиражируемыми данными не требуется ее модификации. В этом, собственно, и состоит качество б (прозрачное тиражирование – см. стр. 21).

Синхронное обновление DDB и DR-технология – в определенном смысле антиподы. Краеугольный камень первой – синхронное завершение транзакций одновременно на нескольких узлах распределенной системы, то есть синхронная фиксация изменений в DDB. Ее «ахиллесова пята» – жесткие требования к производительности и надежности каналов связи. Если база данных распределена по нескольким территориально удаленным узлам, объединенным медленными и ненадежными каналами связи, а число одновременно работающих пользователей составляет сотни и выше, то вероятность того, что распределенная транзакция будет зафиксирована в обозримом временном интервале, становится чрезвычайно малой. В таких условиях (характерных, кстати, для большинства отечественных организаций) обработка распределенных данных практически невозможна.

DR-технология не требует синхронной фиксации изменений, и в этом ее сильная сторона. В действительности далеко не во всех задачах требуется обеспечение идентичности БД на различных узлах в любое время. Достаточно поддерживать тождественность данных лишь в определенные критичные моменты времени. Можно накапливать изменения в данных в виде транзакций в одном узле и периодически копировать эти изменения на другие узлы.

Налицо преимущества DR-технологии. Во-первых, данные всегда расположены там, где они обрабатываются, – следовательно, скорость доступа к ним существенно увеличивается. Во-вторых, передача только операций, изменяющих данные (а не всех операций доступа к удаленным данным), и к тому же в асинхронном режиме позволяет значительно уменьшить трафик. В-третьих, со стороны исходной базы для принимающих баз репликатор выступает как процесс, инициированный одним пользователем, в то время как в физически распределенной среде с каждым локальным сервером работают все пользователи распределенной системы, конкурирующие за ресурсы друг с другом. Наконец, в-четвертых, никакой продолжительный сбой связи не в состоянии нарушить передачу изменений. Дело в том, что тиражирование предполагает буферизацию потока изменений (транзакций); после восстановления

связи передача возобновляется с той транзакции, на которой тиражирование было прервано.

DR-технология данных не лишена недостатков. Например, невозможно полностью исключить конфликты между двумя версиями одной и той же записи. Он может возникнуть, когда вследствие все той же асинхронности два пользователя на разных узлах исправят одну и ту же запись в тот момент, пока изменения в данных из первой базы данных еще не были перенесены во вторую. При проектировании распределенной среды с использованием DR-технологии необходимо предусмотреть конфликтные ситуации и запрограммировать репликатор на какой-либо вариант их разрешения. В этом смысле применение DR-технологии – наиболее сильная угроза целостности DDB. DR-технологии нужно применять крайне осторожно, только для решения задач с жестко ограниченными условиями и по тщательно продуманной схеме, включающей осмысленный алгоритм разрешения конфликтов.

### 1.2.7. Распределенные системы

Сегодня можно считать, что распределенные базы данных – тема достаточно локальная и далеко не так актуальная, как архитектура распределенных систем. В DDB-технологии за последние 2-3 года не было каких-либо существенных новаций (за исключением, быть может, технологии тиражирования данных). Можно считать, что в этой сфере информатики все более или менее устоялось и каких-либо революционных шагов не предвидится. Более интересное направление (включающее DDB) – архитектура, проектирование и реализация распределенных информационных систем. «Горячие» темы в этом направлении – системы с трехзвенной архитектурой, продукты класса middleware, объектно-ориентированные средства разработки распределенных приложений в стандарте CORBA. Их активное применение будет доминировать в отечественной информатике в ближайшие 3-5 лет и станет технологической базой реальных интеграционных проектов.

Технологический взрыв в Интернет (Internet), создание и супербурное развитие Всемирной паутины, технология Java неизбежно отразятся на организации инфраструктуры корпораций. Очевидные преимущества гипертекстовой организации данных (гибкость, открытость, простота развития и расширения) перед жесткими структурами реляционных баз данных, по своей природе плохо приспособленными для расширения, предопределяют использование **HTML** (специальный язык разметки гипертекстовых документов) в качестве одного из основных средств создания информационного пространства компании. Подход, опирающийся на гипертексты, позволяет без особых проблем интегрировать уже существующие информационные массивы, хранящиеся в базах данных. То, что сейчас называют Intranet, – это прообраз будущей корпоративной информационной системы.

### 1.3. Средства доступа к ресурсам глобальных сетей

#### 1.3.1. Характеристика основных информационных ресурсов Интернет

Как упоминалось выше, одним из видов интеграции уже существующих информационных массивов, хранящихся в базах данных, является Internet. Информация в Интернет хранится на так называемых **Web-серверах** (серверах, на которых хранятся мультимедиа документы, связанные между собой гипертекстовыми ссылками). Таким образом, Интернет является глобальной информационной сетью – **открытой системой** (АИС различного назначения, размещенных в различных узлах телекоммуникационной сети, работу которых как единого интегрированного целого обеспечивает система отраслевых, государственных и международных стандартов в области информационных технологий, специфицирующих интерфейсы, услуги и поддерживающих форматы данных для достижения взаимодействия и переносимости приложений, данных и персонала).

В силу колоссального объема и разнородности организации информационных ресурсов в сети возникает ряд естественных проблем. Каждый ресурс имеет структуру определенного типа, базируется на машине со своей операционной системой (платформой) и специальной программой обслуживания доступа к ней – программой-сервером. Машину, непрерывно функционирующую в сети, где выполняется такая программа, также часто называют сервером (Web-сервером). Само соединение пользователя с сервером происходит с помощью соответствующей программы, запускаемой на его компьютере (программы-клиента, или **навигатора в Интернет (browser)** – пользовательской программы, используемой для поиска информационных ресурсов в сетях ЭВМ и доступа к ним), и выполняется такое соединение на основе заранее определенного свода правил, или протокола взаимодействия между клиентом и сервером.

Отметим отдельно, что пользователь Интернет может получить доступ к ресурсам других сетей благодаря существованию межсетевых шлюзов. Под *шлюзом (gateway)* принято понимать специализированный узел (рабочую станцию, компьютер) локальной сети, обеспечивающий доступ других узлов данной локальной сети к внешней сети передачи данных и другим вычислительным сетям. Говоря о *межсетевом шлюзе*, часто подразумевают и аппаратные, и программные средства, обеспечивающие межсетевую связь.

Передача информации в Интернет происходит небольшими порциями данных, имеющими строго определенную структуру и называемыми пакетами. Сообщение может быть разбито на несколько пакетов, размер которых может варьироваться, но, как правило, не превышает 1500 байт.

Важнейшим моментом при функционировании Интернет является стандартизированный свод правил передачи пакетов данных в Сети и за ее пределы в рамках межсетевого обмена, закрепленный базовым транспортным протоколом TCP (Transmission Control Protocol) и межсетевым протоколом IP (Internet Protocol). Протокол TCP дает название всему семейству протоколов TCP/IP, главной задачей которых является объединение в сети пакетных подсетей через шлюзы. Каждая сеть работает по своим собственным законам, но предполагается, что шлюз может принять пакет из другой сети и доставить его по указанному адресу. Реально пакет из одной сети передается в другую подсеть через последовательность шлюзов, что становится возможным благодаря реализации во всех узлах сети протокола межсетевого обмена IP.

Величину потока информации (объем последней измеряется в битах или байтах и единицах, им кратных), прошедшего за определенный промежуток времени через выделенный канал связи, шлюз или другую систему, принято называть *трафиком*.

В Интернет каждой машине (host) приписан определенный адрес, по которому к ней и осуществляется доступ в рамках одного из стандартных протоколов, причем существует одновременно как числовая адресация (так называемый IP-адрес, состоящий из набора четырех трёхразрядных чисел, разделенных точками, например 144.206.160.32), так и более удобная для восприятия человеком система осмысленных *доменных имен* (например, apollo.polyn.kiae.su). Пользователь для обращения к машине может использовать как ее IP-адрес, так и ее имя. Для упрощения работы в сети используется специальная система DNS (Domain Name System), представляющая собой базу данных, которая обеспечивает преобразования доменных имен компьютеров в числовые IP-адреса, поскольку базовым элементом адресации для семейства протокола TCP/IP являются IP-адреса, а доменная адресация выполняет роль сервиса.

Информационные ресурсы Интернет – это вся совокупность информационных технологий и баз данных, доступных при помощи этих технологий и существующих в режиме постоянного обновления. К их числу относятся, например:

- *электронная почта* (с помощью **почтовых серверов** – серверов, обеспечивающих передачу и хранение электронной почты);
- *система телеконференций Usenet* (с помощью **серверов новостей** – средства для оперативного обмена информацией между пользователями сети Интернет). Система построена по принципу электронных досок объявлений, когда любой пользователь может поместить свою информацию в одну из групп новостей Usenet и эта информация станет доступной другим пользователям, которые на данную группу новостей подписаны;
- *система файловых архивов FTP* (протокол передачи данных, который позволяет в Интернет перемещать файлы любого

формата с одного компьютера на другой). Объем программного обеспечения в архивах FTP составляет терабайты информации, и ни один пользователь или администратор сети не может просто физически обозреть эту информацию. Организован **фонд свободного программного обеспечения – FSF** (программистская организация, занимающаяся устранением ограничений по копированию, распространению, изучению и модификации программ для компьютеров; для достижения этой общей задачи FSF стимулирует разработку и использование свободного программного обеспечения, ориентированного на широкий класс применений). Практически любой архив строится как иерархия директорий. Многие архивы дублируют информацию из других архивов (так называемые зеркала – mirrors);

- *базы данных WWW*. World Wide Web (WWW) представляет удобный доступ к большинству информационных архивов Сети. Особенностью системы является механизм гипертекстовых ссылок, который позволяет просматривать материалы в порядке выбора этих ссылок пользователем. Система универсальных адресов позволяет проадресовать практически все информационные ресурсы Интернет. В WWW существует большое количество различного рода каталогов, которые позволяют ориентироваться в сети; кроме этого, пользователи могут выполнить даже удаленные программы или смотреть фильмы по сети. Такой сервис не обеспечивается другими информационными системами Интернет;
- *базы данных Gopher* (клиентов Интернет, предназначенных для поиска и получения текстовых файлов от Gopher-серверов). Внешне Gopher выглядит как огромная файловая система, которая расположена на машинах сети. Gopher считается простой системой, легкой в установке, администрировании, достаточно надежной и защитной;
- *базы данных WAIS*. В основу системы положен принцип поиска информации с использованием логических запросов, основанных на применении ключевых слов. Клиент «обшаривает» все серверы WAIS на предмет наличия в них документов, удовлетворяющих запросу. WAIS широко применяется как поисковая машина в других информационных серверах Интернет, например WWW и Gopher;
- *информационные ресурсы LISTSERV*. LISTSERV – это система почтовых списков сети BIT-NET (сеть образовательных учреждений). LISTSERV специально ориентирован на применение в качестве транспорта электронной почты. Доступ к нему в интерактивном режиме затруднен;
- *справочная служба WHOIS*. WHOIS-служба содержит информацию о пользователях сети, их электронные и обычные адреса, идентификаторы и реальные имена. WHOIS – распределенная система;

- *информационные ресурсы TRICKLE*. TRICKLE – это доступ по почте к архивам FTP, который организован через специальный шлюз;
- *поисковые машины Open Text Index, Alta Vista, Yahoo, Lycos и др.* Они представляют собой мощные информационно-поисковые системы, размещенные на серверах свободного доступа, специальные программы которых непрерывно в автоматическом режиме сканируют информацию Сети на основе заданных алгоритмов, проводя индексацию документов.

Интернет – это, главным образом, возможность получить информацию в тот же момент, когда она нужна, т.е. в режиме on-line. Но если нет возможности работать в on-line, то для доступа к услугам большинства информационных серверов Интернет можно воспользоваться электронной почтой, хотя в этом случае все будет происходить не так быстро, как в стандартном режиме telnet, ftp или WWW, о которых будет сказано ниже.

### 1.3.2. Принципы функционирования Интернет

**ИЕРАРХИЯ ПРОТОКОЛОВ ИНТЕРНЕТ.** *Протокол* – это некоторый свод четко определенных правил, которые одинаково реализованы в различных системах (программах, шлюзах, пакетах данных и др.). Благодаря этому в местах взаимодействия этих систем, например, при инициировании соединения программы-клиента с программой-сервером или при попадании передаваемого пакета данных на машину-шлюз, все происходит по заранее определенному сценарию.

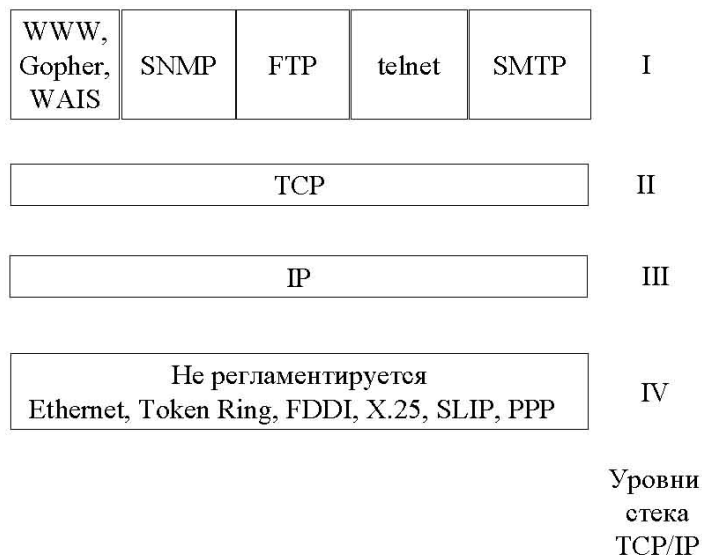
По мере продвижения пакета данных по сети на каждом этапе его взаимодействия с другими сетевыми элементами отработывают протоколы разных уровней. Полную совокупность таких протоколов, необходимых для успешного взаимодействия разных элементов в рамках сети данного типа, принято называть семейством или стеком. Интернет работает под семейством протоколов TCP/IP, которое имеет многоуровневую структуру.

Структура протоколов TCP/IP имеет четыре уровня и приведена на рис. 1.7.

Самый нижний (уровень IV) соответствует уровню доступа к сети. В протоколах TCP/IP он не регламентируется, но поддерживает все популярные стандарты протоколов физического и канального уровня, такие как Ethernet, Token Ring, SLIP, PPP и другие. Протоколы данного уровня обеспечивают передачу пакетов данных в сети на уровне аппаратных средств.

Следующий уровень (уровень III) – это уровень межсетевого взаимодействия, который обеспечивает передачу пакетов данных из одной подсети в другую. В качестве протокола в стеке используется протокол IP.

Следующий уровень (уровень II) называется основным. На этом



**Рис. 1.7**

уровне функционирует протокол управления передачей TCP, который обеспечивает надежную передачу сообщений между удаленными друг от друга различными прикладными программами за счет образования виртуальных соединений между ними.

Все перечисленные выше протоколы с легким сердцем можно отнести к «уровню секретарей», т.е. к обслуживающим интерфейс взаимодействия с пользователем. Для конечного пользователя («начальника») наиболее необходима компетентность на самом верхнем уровне (уровень I), который называется на языке стека TCP/IP прикладным.

За долгие годы использования в сетях различных стран и организаций стек TCP/IP накопил большое количество протоколов и сервисов прикладного уровня. Предметом нашего детального рассмотрения будут следующие четыре: протокол копирования файлов FTP (File Transfer Protocol); протокол эмуляции терминала telnet; протокол Gopher; для доступа к ресурсам всемирного пространства GopherSpace и наиболее популярный на данный момент протокол HTTP (Hyper Text Transfer Protocol) доступа к удаленным гипертекстовым базам данных во всемирный паутине WWW (World Wide Web).

Для того чтобы предотвратить путаницу в терминах, связанную с одинаковыми именами протоколов, программ и информационных ресурсов, сделаем несколько замечаний.

Так, под термином FTP понимается как сам стандарт протокола, так и программа-клиент на компьютере пользователя, которая

инициирует соединение с одноименной (или с добавлением буквы «d» – ftpd) программой-сервером. Последняя, в свою очередь, исполняется на машине-сервере и пребывает в режиме постоянного ожидания запроса от клиента. Кроме того, о самих ресурсах, доступ к которым осуществляется по протоколу ftp, принято говорить как об ftp-архивах. Аналогичная ситуация характерна и для остальных протоколов.

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

Итак, в основу построения адреса ресурса в сети оказались заложены следующие понятия и принципы:

- *Расширяемость* – новые адресные схемы должны были легко вписываться в существующий синтаксис URI (Universal Resource Identifier – универсальный индикатор ресурса).
- *Полнота* – по возможности, любая из существующих схем должна описываться посредством URI.
- *Читаемость* – адрес должен быть легко читаем человеком, что вообще характерно для технологии WWW.

Расширяемость была достигнута за счет выбора определенного порядка интерпретации адресов, который базируется на понятии «адресная схема». Идентификатор схемы стоит перед остатком адреса, отделен от него двоеточием и определяет порядок интерпретации остатка.

Полнота и читаемость порождали коллизию, связанную с тем, что в некоторых схемах используется двоичная информация. Эта проблема была решена за счет формы предоставления такой информации. Символы, которые несут служебные функции, и двоичные данные отображаются в URI в шестнадцатеричном коде и предваряются символом «%».

Прежде, чем рассмотреть различные схемы представления адресов, приведем пример простого адреса URI: <http://polyn.net.kiae.su/polyn/index.html>.

Перед двоеточием стоит идентификатор схемы адреса – «http». Это имя отделено двоеточием от остатка URI, который называется «путем». В данном случае путь состоит из доменного адреса машины, на которой установлен сервер HTTP, и пути от корня дерева сервера к файлу «index.html».

Кроме представленной выше полной записи URI существует упрощенная. Она предполагает, что к моменту ее использования многие параметры адреса ресурса уже определены (протокол, адрес машины в сети, некоторые элементы пути).

При таких предположениях автор гипертекстовых страниц может указывать только относительный адрес ресурса, т.е. адрес относительно определенных базовых ресурсов.

**СХЕМЫ АДРЕСАЦИИ РЕСУРСОВ ИНТЕРНЕТ.** В RFC-1630 (Request for Comment – документы с таким названием содержат в себе материалы по Интернет-технологии, которые доведены до уровня стандарта или близки к этому уровню) рассмотрено восемь схем адресации Интернет и указаны две, синтаксис которых находится в стадии обсуждения.

**Схема HTTP.** Это основная схема для WWW. В схеме указываются ее идентификатор, адрес машины, TCP-порт, путь в директории сервера, поисковый критерий и метка.

Следует отличать понятие TCP-порта от физического разъема на задней стенке системного блока компьютера. В Интернет принято идентифицировать конкретную прикладную программу с определенным числом, или портом (это понятие абсолютно не связано с названием физического устройства ввода-вывода компьютера). Всякий раз передаваемый по сети от одного компьютера к другому пакет данных содержит информацию о том, какой именно протокол используется и с какой прикладной программой машины пытается установить связь. Номер порта и обозначает эту прикладную программу.

Приведем несколько примеров URI для схемы HTTP: <http://polyn.net.kiae.su/polyn/mamfest.html>.

Это наиболее распространенный вид URI, применяемый в документах WWW. Вслед за именем схемы (http) следует путь, состоящий из доменного адреса машины и полного адреса HTML-документа в дереве сервера HTTP.

В качестве адреса машины допустимо использование и IP-адреса: <http://144.206.160.40/risk/risk.html>.

Если сервер протокола HTTP запущен на другой, отличный от 80 порт TCP, то это отражается в адресе: <http://144.206.130.137-8080/altai/index.html>.

При указании адреса ресурса возможна ссылка на точку внутри файла HTML. Для этого вслед за его именем может быть указана метка внутри документа: <http://polyn.het.kiae.su/altai/volume4.html#first>. Символ # отделяет имя документа от имени метки.

**Схема FTP.** Данная схема позволяет адресовать файловые архивы FTP из программ-клиентов World Wide Web. При этом программа должна поддерживать протокол FTP. В данной схеме возможно указание не только имени схемы, адреса FTP-архива, но и идентификатора пользователя и даже его пароля. Наиболее часто используется для доступа к публичным архивам FTP: <ftp://polyn.net.kiae.su/pub/0index.txt>.

**Схема Gopher.** Данная схема используется для ссылки на ресурсы распределенной информационной системы Gopher. Схема состоит из идентификатора и пути, в котором указывается адрес Gopher-сервера,

тип ресурса и команда Gopher: `gopher://gopher.kiae.su:70:/7/kuku`.

В этом примере осуществляется доступ к gopher-серверу `gopher.kiae.su` через порт 70 для поиска (тип 7) слова «kuku». Следует заметить, что gopher-тип, в данном случае 7, передается не перед командой, а вслед за ней.

**Схема MAILTO.** Данная схема предназначена для отправки почты по стандарту RFC-822 (стандарт почтового сообщения). Общий вид схемы выглядит так: `mailto:paul@quest.polyn.kiae`.

**Схема TELNET.** По этой схеме осуществляется доступ к ресурсу в режиме удаленного терминала. Обычно клиент вызывает дополнительную программу для работы по протоколу telnet. При использовании этой схемы необходимо указывать идентификатор пользователя, допускается использование пароля. Реально доступ осуществляется к публичным ресурсам, и идентификатор и пароль являются общеизвестными, например, их можно узнать в базах данных Hytelnet. `telnet://guest:password@apollo.polyn.kiae.su`.

**Схема FILE.** WWW-технология используется как в сетевом, так и в локальном режимах. Для локального режима используют схему FILE: `file:/C:/text/html/inaex.htm`.

В данном примере приведено обращение к локальному документу на персональном компьютере MS-DOS или MS-Windows.

Существует еще несколько схем, которые на практике используются редко или находятся в стадии разработки, поэтому останавливаться на них не будем.

Из приведенных выше примеров видно, что спецификация адресов ресурсов URI является довольно общей и позволяет адресовать практически любой ресурс Интернет. При этом число ресурсов может расширяться за счет создания новых схем. Они могут быть похожими на существующие, а могут и отличаться от них. Реальный механизм интерпретации идентификатора ресурса, опирающийся на URI, называется URL (Uniform Resource Locator), и пользователи WWW имеют дело именно с ним.

### 1.3.3. Технология World Wide Web

WWW – это аббревиатура от «World Wide Web» («Всемирная паутина»). Официальное определение World Wide Web звучит как мировая виртуальная файловая система – «широкомасштабная гипермедиа-среда, ориентированная на предоставление универсального доступа к документам».

Проект WWW возник в начале 1989 г. в Европейской Лаборатории физики элементарных частиц в Швейцарии. Используя популярный программный интерфейс, проект WWW изменил процесс просмотра и создания информации. Идея заключается в том, что по всему миру хаотично разбросаны тысячи информационных серверов и любую

машину, подключенную к Internet в режиме on-line, можно преобразовать в сервер и начинить его информацией. С любого компьютера, подключенного к Internet, можно свободно установить сетевое соединение с таким сервером и получать от него информацию.

Информационный WWW-сервер использует гипертекстовую технологию. Для записи документов в гипертексте используется специальный, но очень простой язык HTML (HyperText Markup Language), который позволяет управлять шрифтами, отступами, вставлять цветные иллюстрации, поддерживает вывод звука и анимации. В стандарт языка также входит поддержка математических формул.

Внешне гипертекст отличается от обычного текста тем, что часть слов или целые строки в нем, будучи выделены особым шрифтом или цветом, оказываются чувствительными к появлению на них указателя манипулятора «мышь». При попадании на такую область текста указатель (часто стрелочка) изменяет первоначальный вид, становясь, например, ладошкой. Щелчок «мыши» в таком положении приводит к инициированию какого-либо события, чаще всего к загрузке в программу просмотра нового документа, привязанного так называемой гипертекстовой ссылкой к выделенной строке текста. В результате у пользователя появляется возможность самому выбирать порядок просмотра тех или иных страниц, двигаясь по перемежающимся между собой нитям – паутинкам ссылок. Если при этом компьютер подключен к глобальной сети Интернет, то в сценарий просмотра могут входить ресурсы всего мира, доступ к которым происходит по протоколу работы с гипертекстом – HTTP (Hyper Text Transfer Protocol). После сказанного становится понятным представление об этих ресурсах как о Всемирной паутине.

Поскольку нетривиальный характер взаимодействия клиента и сервера по протоколу HTTP с удаленными ресурсами сети скрыт от конечного пользователя за интерфейсом дружественной программы просмотра гипертекстовых страниц (броузером, от англ. browse – просматривать), начало работы в Web не представляет больших проблем.

Итак, гипертекст не может корректно отображаться обычным текстовым редактором, хотя последний вполне пригоден для его приготовления. Специально разработанный язык гипертекстовой разметки HTML позволяет превращать нужные элементы документа, включая не только текстовые поля, но и графику, в области «мышечувствительности», или в гипертекстовые ссылки. Существует ряд серьезных причин, по которым необходимо остановиться на этом языке ниже чуть более подробно.

Для удобства ввода информации предусмотрены специальные формы, меню. Программы просмотра позволяют получать доступ не только к WWW-серверам, но и к другим службам Internet. С их помощью можно путешествовать по Gopher-серверам, искать информацию в WAIS-базах, получать файлы с файловых серверов по протоколу FTP. Поддерживается протокол обмена сетевыми новостями Usenet NNTP.

WWW – это в настоящее время самый популярный и самый

интересный сервис Интернет, самое популярное и удобное средство работы с информацией. Самое распространенное имя для компьютера в Интернет сегодня – www, больше половины потока данных Интернет приходится на долю WWW. Количество серверов WWW сегодня нельзя оценить сколько-либо точно, но по некоторым оценкам их более 300 тысяч. Скорость роста WWW даже выше, чем у самой сети Интернет.

WWW работает по принципу клиент-сервер, точнее, клиент-серверы: существует множество серверов, которые по запросу клиента возвращают ему гипермедийный документ – документ, состоящий из частей с разнообразным представлением информации, в котором каждый элемент может являться ссылкой на другой документ или его часть. Ссылки эти в документах WWW организованы таким образом, что каждый информационный ресурс в глобальной сети Интернет однозначно адресуется, и документ, который вы читаете в данный момент, способен ссылаться как на другие документы на этом же сервере, так и на документы (и вообще на ресурсы Интернет) на других компьютерах Интернет. Таким образом, программные средства WWW являются универсальными для различных сервисов Интернет, а сама информационная система WWW играет интегрирующую роль.

Подключение к Internet производится посредством сетевого адаптера или другого сетевого устройства, например модема или платы ISDN (Integrated Services Digital Network – цифровая сеть с интеграцией сервиса). Скорость передачи информации в Internet выражается в битах в секунду. Для подключения к Интернету необходим ISP (Internet Service Provider – поставщик услуг Интернета). ISP предоставляет клиентам доступ к Интернету по телефонным или другим линиям. Кроме того, ISP предоставляет такие услуги, как аренда пространства на сервере и создание Web-страниц.

Обращаясь к ISP, необходимо указать сервисы и потребность в полосе пропускания. После заключения контракта ISP сообщит ваш адрес IP, маску подсети, имена серверов DNS, проинструктирует о подключении его к сети и порекомендует любое необходимое дополнительное оборудование.

При выборе ISP основные критерии – местоположение, цена, надежность и набор предоставляемых сервисов.

*Регистрация имени домена.* Домены в Интернете различаются по уровням иерархии, например в iae.ltiae – домен второго уровня, а lt – верхнего. Создавая домен, необходимо зарегистрировать его в руководящей организации, тогда имя домена будет включено в имя ее домена. Домены верхнего уровня классифицируют организации по типам (используется в США): gov (government – государственные), edu (educational – образовательные), org (organization – организации), net (главные центры поддержки сети), mil (военные группы), int (международные), com (commercial – коммерческие), <country code> (любая страна, географическая единица).

Имя домена должно иметь смысл, легко запоминаться и вводиться с клавиатуры, а также не использоваться другой организацией на Интернете. Выбранное подходящее имя регистрируется. Обычно для этого из области Registration Web-страницы InterNIC получают текстовый бланк и заполняют его в любом редакторе или текстовом процессоре или заполняют форму WWW, используя программу просмотра Web.

Необходимо сообщить InterNIC о себе некоторые данные. Во-первых, кто будет контактировать с ней по административным, техническим или финансовым вопросам, касающимся домена. Во-вторых, имена и IP-адреса серверов DNS, поддерживающих домен. Заполненная форма отсылается электронной почтой в InterNIC. Через некоторое время поступают два ответа: первый – подтверждение получения запроса, второй – разрешение на использование имени домена.

#### **1.3.4. Программы-клиенты WWW**

Наиболее распространенными программами этого типа являются Mosaic, Netscape Navigator, Internet Explorer (графический интерфейс) и Lynx для алфавитно-цифрового режима доступа. Приведем здесь их краткие характеристики.

Lynx – полноэкранный интерфейс доступа к WWW. Данный интерфейс обеспечивает доступ к WWW с алфавитно-цифровых устройств типа терминала vt100. Интерфейс поддерживает все возможности языка HTML 2.0, за исключением графики.

Internet Explorer (Microsoft) и Netscape Navigator (Netscape Communications) – близкие по своим возможностям многопротокольные графические интерфейсы доступа к WWW и другим ресурсам Сети, интерпретирующие язык гипертекстовой разметки HTML 3.2 (речь идет о версиях продуктов 1997 г.) и поддерживающие средства работы с объектами гипермедиа.

Учитывая высокую популярность броузера Netscape Navigator, а также тот факт, что им в отличие от Internet Explorer не поддерживается меню на русском языке (начиная с версии 4.0 такая поддержка введена), более подробно обсудим здесь возможности, предоставляемые этой программой.

Netscape Navigator реализован для таких платформ, как UNIX, Windows, Macintosh и является мощной многопротокольной программой, позволяющей эффективно организовать доступ ко многим ресурсам Сети. В своей четвертой версии он существует в виде одного из компонентов интегрированного многофункционального пакета Netscape Communicator.

Интерфейс программы для Windows 95/NT можно представить следующим образом. В самом верхнем поле окна отображаются название программы и имя текущего документа, которое указывается в его заголовке. Чуть ниже располагаются элементы главного меню,

которые раскрываются по щелчку «мыши» на их именах. Сразу под ними размещена панель с кнопками быстрого доступа к наиболее часто используемым командам. В принципе весь спектр возможностей программы можно извлечь из главного меню, а все остальные поля дублируют пункты главного меню и служат для ускорения работы с пакетом, их можно сделать невидимыми.

На следующем уровне представлена иконка Bookmarks, являющаяся точкой входа в сервис работы с закладками. Сбоку от нее находится, пожалуй, самое главное поле программы – окно Location, предназначенное для ввода адреса ресурса (его URL). Стрелочка-указатель в правой части окна Location предлагает раскрывающееся меню, содержащее до 14 URL наиболее часто просматриваемых пользователем страниц.

Уровнем ниже размещена персональная панель пользователя, которую он может конструировать по своему усмотрению. Далее следует обширная область, в которую загружается документ (Web-страница). Система прокрутки (в виде полосы справа и снизу от текста) позволяет просматривать содержимое страницы, не поместившееся в один экран. Для того чтобы различать ссылки в тексте на уже просмотренные страницы от еще не просмотренных, для первых по умолчанию используется синий цвет, для вторых – фиолетовый. Копирование части текста страницы можно выполнить с помощью меню или штатных средств Windows.

В нижней части окна располагаются поле состояния загрузки, поле статуса сообщения и панель иконок, составляющих программ-компонентов пакета. При этом поле состояния загрузки показывает количество загруженной части текущего документа в процентах. Поле сообщения статуса отображает текст, относящийся к загружаемому документу, в том числе и текущую скорость передачи информации. При наложении указателя мыши на гипертекстовую ссылку в документе в этом поле отображается ее URL, а при работе с картой чувствительного изображения – текущие координаты указателя.

Сообщение в поле статуса типа Document Done означает лишь, что загружен очередной объект страницы, например, картинка, а не документ в целом. Поскольку на экране может появиться только часть страницы до ее полной загрузки, то и полоса прокрутки появляется только при необходимости и возможности движения вверх-вниз по документу. О полной загрузке страницы сообщает только поле состояния загрузки. Панель составляющих программ-компонентов позволяет запустить программы пакета Navigator (броузер). Messenger Mailbox (электронная почта), Collabra Discussion Groups (просмотр новостей телеконференций) и Page Composer (редактор HTML-документов).

### **1.3.5. Стратегия поиска информации в Интернет**

Как разыскать в Сети необходимую информацию в условиях, когда ее поток непрерывно и до определенной степени бесконтрольно

возрастает. Наиболее убедительной попыткой обуздать информационный хаос в Интернет является культивирование поисковых машин самого широкого профиля. В их функции входит автоматическое или полуавтоматическое сканирование (просмотр) узлов сети, сопровождающееся индексированием (созданием баз данных) и классификацией их ресурсов (построением каталогов, структурированных по различным критериям) с возможностью последующего обслуживания поисковых запросов клиентов. Общий вид функциональной цепочки обобщенной поисковой машины представлен на рис. 1.8.



**Рис. 1.8**

***Сканирование.*** В процессе сканирования ресурсов Сети принимают участие специальные программы, в WWW их часто называют «паучками». Работа таких программ обычно происходит в автоматическом режиме и состоит в последовательном обходе узлов Сети на основе заданного алгоритма, который может отдавать определенные предпочтения тем или иным хостам (узлам) как на основе их географической или профильной принадлежности, так и частоты изменения находящихся на них ресурсов. Кроме того, учитываются интересы компаний, стремящихся включить свои серверы в индексную базу данной поисковой машины и проинформировать о них широкий круг пользователей сети. В отношении сказанного важной характеристикой машины является число уже отсканированных узлов и скорость работы сканирующих программ.

***Индексирование*** предполагает формирование базы данных поисковой машины, организованной по определенным принципам. В первую очередь, безусловно, предметом сканирования являются текстовые документы. В результате такой операции для каждого документа формируется набор ключевых слов, по которым затем на стадии обслуживания поискового запроса пользователю выдаются адреса заиндексированных ресурсов.

Информационные объекты нетекстового характера (графика, видео, аудио) в общем случае также могут идентифицироваться и быть представлены в соответствующих базах данных.

*Классификация* ресурсов является дополнительной функцией поисковой машины, которая предполагает, например, присвоение при индексировании пометки о принадлежности данного информационного объекта к определенному типу.

*Обслуживание* пользователя той или иной поисковой машиной строится на разработке информационно-поискового языка, естественным образом связанного со структурой базы данных. Типичными являются два основных подхода: пользователю предоставляется возможность вести поиск интересующей его информации либо путем осмысленного на каждом шаге перемещения по дереву иерархического каталога, уже построенного и жестко определенного системой, либо путем реализации собственного поискового запроса в рамках поддерживаемого системой поискового языка. Конечной точкой обоих путей является локализация и извлечение соответствующего информационного объекта.

В процессе сканирования поисковой машине приходится получать доступ к ресурсам сети, естественно, что такой доступ реализуется в рамках одного из протоколов прикладного уровня. В связи с этим принято различать поисковые машины по области сканирования, прежде всего это – гипертекстовые базы данных Web, ресурсы всемирного пространства GopherSpace, FTP-архивы.

В мире Интернет технологии WWW произвели революцию, следствием которой стали следующие факторы:

- неуклонное нарастание числа серверов в Сети, реализующих http-протокол;
- перенесение наиболее востребуемых ресурсов на Web-узлы с серверов, поддерживающих другие протоколы доступа;
- разработка системы межпротокольных шлюзов WWW-Gopher, WWW-FTP, WWW-Telnet.

Существование шлюзов между протоколами прикладного уровня позволяет, например, поисковой машине WWW сканировать ресурсы FTP-архивов, тем не менее, инфраструктуры межпротокольных шлюзов оказывается явно недостаточно для формирования однородного информационного пространства. В результате для исчерпывающего профессионального поиска информации в Сети следует прибегать к специальным поисковым средствам, характерным для среды того или иного протокола, а не ограничиваться наиболее развитыми сегодня средствами поисковых машин WWW, полагаясь на полноту охвата остальной части Интернет благодаря шлюзам.

По этой причине поиск информации в Интернет сегодня поднимается на уровень технологии. Подводя итог сказанному, сегодня можно говорить о развитии информационно-поисковых систем в двух направлениях:

1. Возрастание чувствительности поисковых программ к полям сканируемого документа, что фактически приводит к их внедрению в технологию WWW и подразумевает активное использование операторов языка HTML для идентификации значимых для поиска полей документа.
2. Развитие возможностей сужения поиска путем усложнения запроса (применение логических операторов, операторов близости и т. д. для наложения связей на элементы запроса – ниже будут подробно обсуждены).

### **1.3.6. Электронная почта в Интернет**

Электронная почта является чрезвычайно важным информационным ресурсом Интернет. Помимо того, что она представляет собой самое массовое средство электронных коммуникаций, через нее можно принять или послать сообщения еще в два десятка международных компьютерных сетей, часть из которых вовсе не имеют on-line сервиса (т.е. прямого подключения к Интернет).

Электронная почта во многом похожа на обычную почтовую службу. Корреспонденция подготавливается пользователем на своем рабочем месте либо программой подготовки почты, либо обычным текстовым редактором. Затем пользователь должен вызвать программу отправки почты (программа подготовки почты вызывает программу отправки автоматически), которая посылает сообщение на почтовый сервер отправителя. Тот, в свою очередь посылает его на почтовый сервер адресата, где специальная программа занимается сортировкой почты и рассылкой ее по ящикам конечных пользователей. После запуска программы получения почты адресат устанавливает соединение со своим почтовым сервером и организует пересылку всех полученных на свое имя сообщений. Отметим, что почтовые серверы постоянно подключены к сети, тогда как компьютеры участников переписки могут устанавливать соединение с ними по мере необходимости. Кроме того, получить и отправить почту можно через разные серверы Интернет. При настройке программы работы с электронной почтой независимо от ее интерфейса необходима следующая информация от провайдера: имя сервера исходящей почты, имя сервера входящей почты, имя пользователя и пароль, а также типы протоколов, используемые при почтовом обмене.

**ПРОТОКОЛ SIMPLE MAIL TRANSFER PROTOCOL.** Для работы электронной почты в Интернет специально разработан протокол Simple Mail Transfer Protocol (SMTP), который является протоколом прикладного уровня и использует транспортный протокол TCP. Однако совместно с этим протоколом используется и Unix-Unix-CoPy (UUCP) протокол. UUCP хорошо подходит для использования телефонных линий связи. Разница

между SMTP и UUCP заключается в том, что при использовании первого протокола почтового обмена программа, функционирующая на сервере, пытается найти машину получателя почты и установить с ней взаимодействие в режиме on-line для того, чтобы передать почту в ее почтовый ящик. В случае использования SMTP почта достигает почтового ящика получателя за считанные минуты, и время получения сообщения зависит только от того, как часто получатель просматривает свой почтовый ящик. При использовании UUCP почта передается по принципу «stop-go», т.е. почтовое сообщение передается по цепочке почтовых серверов от одной машины к другой, пока не достигнет машины-получателя или не будет отвергнута по причине отсутствия абонента-получателя. С одной стороны, UUCP позволяет доставлять почту по плохим телефонным каналам, так как не требуется поддерживать линию все время доставки от отправителя к получателю, а с другой стороны, время доступа к адресату значительно возрастает. В целом же общие рекомендации таковы: если имеется возможность надежно работать в режиме on-line и это является нормой, то следует настраивать почту для работы по протоколу SMTP, если линии связи плохие или on-line используется чрезвычайно редко, то лучше использовать UUCP.

Основой любой почтовой службы является система адресов. Без точного адреса невозможно доставить почту адресату. В Интернет принята система адресов, которая базируется на доменном адресе машины. Например, для пользователя tala машины с адресом citmgu.ru почтовый адрес будет выглядеть так: tala@citmgu.ru

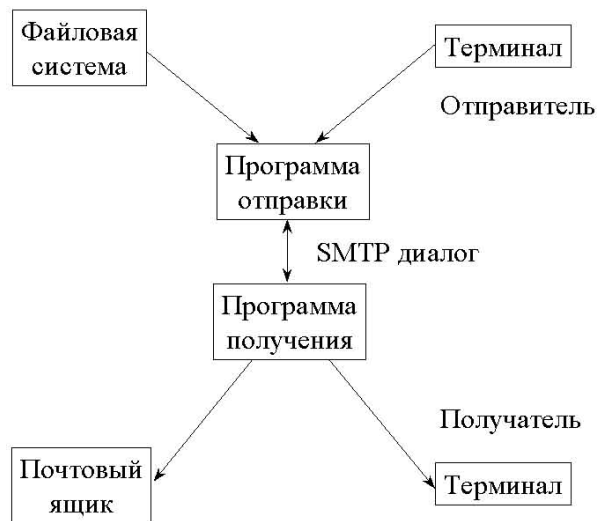
Таким образом, адрес состоит из двух частей: идентификатора пользователя, который записывается перед знаком «коммерческого эй» – «@», и доменного адреса машины, который записывается после знака «@».

Протокол SMTP был разработан для обмена почтовыми сообщениями в сети Интернет, он не зависит от транспортной среды и может использоваться для доставки почты в сетях с протоколами, отличными от TCP/IP.

Взаимодействие в рамках SMTP строится по принципу двусторонней связи, которая устанавливается между отправителем и получателем почтового сообщения. При этом отправитель инициирует соединение и посылает запросы на обслуживание, а получатель на эти запросы отвечает. Фактически, отправитель выступает в роли клиента, а получатель – сервера (см. рис. 1.9).

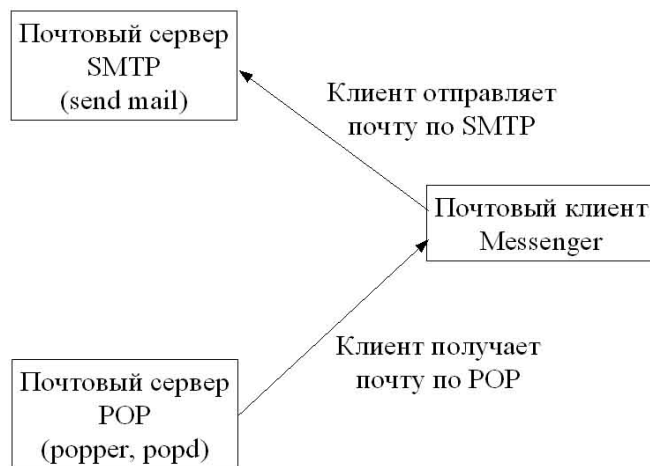
Канал связи устанавливается непосредственно между отправителем и получателем сообщения. При таком взаимодействии почта достигает абонента в течение нескольких секунд после отправки.

**ПРОТОКОЛ POP3.** Протокол обмена почтовой информацией POP3 (Post Office Protocol, версия 3) предназначен для разбора почты из почтовых ящиков пользователей на их рабочие места при помощи



**Рис. 1.9**

программ-клиентов. Если по протоколу SMTP пользователи отправляют корреспонденцию через Интернет, то по протоколу POP3 они получают корреспонденцию из своих почтовых ящиков на почтовом сервере в локальные файлы (см. рис. 1.10).



**Рис. 1.10**

Такая схема предполагает, что пользователь имеет почтовый ящик на машине-сервере, которая не выключается круглосуточно. Все почтовые сообщения складываются в этот почтовый ящик. По мере необходимости пользователь из своего почтового клиента обращается к почтовому ящику и забирает из него пришедшую на его имя почту. При отправке программа-клиент обращается непосредственно к серверу рассылки почты и передает отправляемые сообщения на этот сервер для дальнейшей рассылки.

Формат почтового сообщения Интернет определен в документе RFC-822 (Standard for ARPA Internet Text Message). Почтовое сообщение состоит из трех частей: конверта, заголовка и тела сообщения. Пользователь видит только заголовок и тело сообщения. Конверт используется только программами доставки. Заголовки всегда находятся перед телом сообщения и отделен от него пустой строкой. RFC-822 регламентирует содержание заголовка сообщения. Заголовок состоит из полей. Поля состоят из имени поля и содержания поля. Имя поля отделено от содержания символом «:». Минимально необходимыми являются поля Date, From, cc или To, например:

```
Date: 26Aug761429EDT
From: Jones@Registry.org
cc: Robert@Registry.org
```

Поле Date определяет дату отправки сообщения, поле From – отправителя, а поля cc и To – получателя (ей). Чаще заголовок содержит дополнительные поля:

```
Date: . 26Aug76 1429EDT
From: George Jones<Jones@Registry.org>
Sender: Secy@SHOST
To: Smith@Registry.org
Message-ID: <4231.629.XYzi-What@Regictry.org>
```

В данном случае поле Sender указывает, что George Jones не является автором сообщения. Он только переслал сообщение, которое получил из Secy@SHOST. Поле Message-ID содержит уникальный идентификатор сообщения и используется программами доставки почты. Следующее сообщение демонстрирует все возможные поля заголовка:

```
Date: 27 Aug 76 0932
From: Ken Davis <Kdavis@This-Host .This.net>
Subject: Re: The Syntax in the RFC
Sender: Ksecy@0ther-host
Reply-To: Sam. Irvine@Reg. Organization
```

To: George Jones <Jones@Registry.org>  
cc: Important folks:  
Tom Softwood <Balsa@Tree.Root>,  
«Sam Irving»2@Other-Host;  
Standard Distribution:  
/main/davis/people/standard@Other-Host Comment:  
Comment Sam is away on business.  
In-Reply-To: <some.string@DBM.Group>, George's message  
X-Special-action: This is a sample of user-defined field-names.  
Message-ID: <4331.629.XYzi-What@Other-Host

Поле Subject определяет тему сообщения, Reply-To – пользователя, которому отвечают, Comment – комментарий, In-Reply-To – показывает, что сообщение является тем, которое выслано «В ответ на Ваше сообщение, отвечающее на сообщение, отвечающее...». X-Special-action – поле, определенное пользователем, которое не определено в стандарте.

Следует сказать, что формат сообщения постоянно дополняется и совершенствуется, кроме того, хотелось бы отметить, что возможности почты не ограничиваются только пересылкой корреспонденции. По почте можно получить доступ ко многим ресурсам Интернет, которые имеют почтовых работников (специальные программы автоматического обслуживания), отвечающих на запросы.

**СТАНДАРТ MIME.** Стандарт MIME (Multipurpose Internet Mail Extension), или в нотации Интернет – документ RFC-1341, предназначен для описания тела почтового сообщения Интернет. Предшественником MIME является стандарт почтового сообщения АКРА (RFC822). Стандарт RFC822 был разработан для обмена текстовыми сообщениями. С момента опубликования стандарта возможности аппаратных средств и телекоммуникаций ушли далеко вперед, и стало ясно, что многие типы информации, которые широко используются в сети, невозможно передать по почте без специальных ухищрений. Так, в тело сообщения нельзя включить графику, аудио, видео и другие типы информации. Естественно, что при использовании RFC822 не может быть и речи о передаче размеченного текста для отображения его различными стилями. Ограничения RFC822 становятся еще более очевидными, когда речь заходит об обмене сообщениями в разных почтовых системах.

В некотором смысле стандарт MIME ортогонален стандарту RFC822. Если последний подробно описывает в заголовке почтового сообщения текстовое тело письма и механизм его рассылки, то MIME главным образом сориентирован на описание в заголовке письма структуры тела почтового сообщения и возможности составления письма из информационных единиц различных типов.

В стандарте зарезервировано несколько способов представления

разнородной информации. Для этой цели используются специальные поля заголовка почтового сообщения:

- поле версии MIME, которое используется для идентификации сообщения, подготовленного в новом стандарте;
- поле описания типа информации в теле сообщения, которое позволяет обеспечить правильную интерпретацию данных;
- поле типа кодировки информации в теле сообщения, указывающее на тип процедуры декодирования;
- два дополнительных поля, зарезервированных для более детального описания тела сообщения.

Стандарт MIME разработан как расширяемая спецификация, в которой подразумевается, что число типов данных будет расти по мере развития форм представления данных. При этом следует учитывать, что анархия типов (безграничное их увеличение) тоже не допустима. Каждый новый тип в обязательном порядке должен быть зарегистрирован в IANA (Internet Assigned Numbers Authority).

#### **1.4. Средства разработки приложений в сетях ЭВМ**

Говоря о том или ином средстве разработки приложений всегда хочется понять, какие тенденции приводят к его появлению. К середине 90-х годов имелись следующие основные направления развития средств разработки приложений, в том числе и в сетях ЭВМ.

Одно направление – *объектно-ориентированный подход*, хорошо структурирующий задачу как таковую, так и ее решение в виде прикладной системы. Другое направление, возникшее во многом благодаря объектной ориентации, – *визуальные средства быстрой разработки приложений* (RAD – Rapid Application Development), основанные на компонентной архитектуре. Третья тенденция – использование *компиляции*, а не интерпретации. Это объясняется тем, что скоростные характеристики компилируемых приложений в десятки раз лучше, чем у систем, использующих интерпретатор. При этом повышается легкость отчуждаемости готовых систем, так как отпадает необходимость «таскать за собой» сам интерпретатор (run-time), выполненный обычно в виде динамической библиотеки и занимающий в лучшем случае несколько сотен килобайт (а большинстве случаев – два-три мегабайта). Отсюда и меньшая ресурсоемкость у скомпилированных систем. Четвертая тенденция – возможность работы с базами данных *универсальными (единообразными) методами*. Если мы попытаемся оценить процент систем, которые так или иначе требуют обработки структурированной информации (как для внутрикорпоративного использования, так и для коммерческого или иного распространения), то окажется, что цифра 60-70 % может представлять лишь нижнюю границу. Важным свойством средств обеспечения доступа к базам данных является их *масштабируемость* как возможность не только количественного, но и качественного

роста системы. Например, обеспечение перехода от локальных, в том числе файл-серверных данных к архитектуре “клиент-сервер” или тем более к многоуровневой N-tier схеме.

Говоря об инструментах, ориентированных на создание систем корпоративных масштабов, мы должны абсолютно четко представлять предъявляемые к ним требования. Попытаемся сформулировать некоторые из них.

1. Крупные информационные системы требуют гибкости инструмента с точки зрения возможности наращивания функциональности повторно используемого программного кода и реализации нестандартных решений (пользовательский интерфейс, межпрограммное взаимодействие, интеграция с унаследованными системами – legacy systems, доступ к системным ресурсам и т.п.). Полнота реализации объектной модели (неограниченные возможности расширения иерархии наследования объектов) плюс возможность изменения функциональности объектов без создания новых объектных типов – классов (обработчики событий).
2. Создание корпоративных систем требует коллективной работы. Поддержка групповой разработки (системы контроля версий, разделяемые словари данных и репозитории объектов) плюс разделение работ за счет абстрагирования задач и конструирования приложений из функционально полных объектов – компонентов, создаваемых членами коллектива для совместного использования.
3. «Единство противоположностей»: Нейтральность по отношению к используемым форматам БД плюс поддержка специфики конкретных способов хранения/доступа к данным. Универсальный механизм доступа к данным.
4. Требования к производительности. Компиляция, в случае платформи-зависимых решений.
5. Охватывание всех этапов разработки – от проектирования до создания отчуждаемых приложений (дистрибутивов), через кодирование и отладку. Открытость среды разработки, в плане возможностей интеграции с другими продуктами.

В настоящее время основными средствами разработки приложений в сетях ЭВМ являются Borland Delphi, C и C++, а также широкий набор специализированных CASE-средств.

Рассмотрим более подробно Delphi. Delphi создавался как продукт, в полной мере реализующий описанные тенденции, с архитектурой, открытой для расширения спектра поддерживаемых стандартов и подходов.

Рассмотрим, насколько Delphi удовлетворяет вышеперечисленным требованиям.

1. Delphi использует язык 3-го поколения Object Pascal, обладающий полной реализацией основных признаков объектной ориентации

(инкапсуляция, наследование, полиморфизм), поддержкой RTTI – RunTime Type Information и встроенной обработкой исключительных ситуаций (Exception handling). Компонентная архитектура Delphi является прямым развитием поддерживаемой объектной модели. Все компоненты являются объектными типами (классами), с возможностью неограниченного наследования. Компоненты Delphi поддерживают PME-модель (Property, Method, Events), позволяющую изменять поведение компонентов без необходимости создания новых классов.

2. Delphi Client/Server Suite включает систему контроля версий Intersolv PVCS, поддерживает работу со словарем данных (Data Dictionary) и репозитарием объектов (Object Repository). Среда визуальной разработки Delphi позволяет единообразно работать как с предопределенными, так и с пользовательскими компонентами, которые разрабатываются на том же языке (Object Pascal), на котором создаются и конечные приложения.
3. Borland Database Engine (BDE) обеспечивает единообразную работу с локальными данными (Paradox, dBase) и серверами БД (Oracle, Sybase, MS SQL Server, InterBase и т.д.) за счет применения навигационных методов доступа к серверным СУБД (двунаправленные курсоры, закладки и т.п.) и SQL – к локальным форматам (подмножество Local SQL).
4. Компилятор Delphi является самым быстрым; имеет общий генератор кода с Borland C++ (Delphi & BC++ 5). Компилятор Delphi (точнее, Object Pascal) является продолжением линии компиляторов Turbo Pascal/Borland Pascal.
5. Открытые интерфейсы Delphi – Open Tools API – обеспечивают контроль над средой разработки «извне» и доступ к информации о проекте.
  - Delphi Client/Server Suite включает CASE Expert, позволяющий импортировать данные из ведущих CASE в словарь данных Delphi, интегрировать IDE (Integrated Development Environment) с генераторами кода (например, Silver Run RDM компании CSA, WithClass 3.0 и т.п.).
  - «Эксперты» (программные модули, встраиваемые в IDE) позволяют использовать Delphi как «скелет» – общую среду разработки – для всего комплекса используемых инструментов.
  - Delphi включает «генератор дистрибутивов» Install Shield Express.

При построении систем масштаба предприятия практически невозможно избежать неоднородности (разные ОС, СУБД, промежуточное ПО и т.п.). Встает вопрос о средствах объединения разных технологических платформ. Достаточно четко можно разбить архитектурно грамотную информационную систему на три «модуля» – клиентский, сервер приложений и БД.

В рамках новой инициативы Golden Gate, Borland объединяет уже

имеющиеся технологии с достижениями Open Environment Corporation – ОЕС (приобретена компанией Borland) в области средств для построения многоуровневых, распределенных систем. Продукт ОЕС OLEnterprise обеспечивает распределенные вычисления на базе технологий OLE-automation/RPC (Remote Procedure Call) поверх D-COM и в отсутствие такового на всех платформах Windows (в том числе Win16). Полная автоматизация импорта/экспорта объектов в сети позволяет избежать необходимости изменения кода приложений для их взаимодействия на разных участках сети.

В силу того, что Delphi полностью поддерживает OLE-automation и предоставляет высокоуровневые средства работы с этими механизмами (специализированные классы, эксперты, языковые расширения), вариант совместного использования Delphi & OLEnterprise может оказать решающее воздействие на архитектуру системы => распределенные вычисления и локальные рабочие места – все в одном коде.

Так как Delphi обеспечивает создание «чистого» (native) кода посредством компиляции (например в самодостаточную – без интерпретатора – динамическую библиотеку DLL), возможна тонкая интеграция полученных программных модулей не только с тремя клиентскими приложениями, но и с серверами приложений и баз данных на платформах Windows (в большей степени Windows NT, как следствие ее приспособленности для поддержки серверных звеньев). В качестве примера можно привести построение определяемых пользователем функций UDF для серверов БД Borland InterBase (например, для специфической обработки BLOB-полей).

Главной целью Golden Gate является объединение лучших черт архитектуры «клиент-сервер» и модели intranet. И первым этапом ее реализации является добавление средств интеграции с Internet-технологиями в уже имеющиеся средства разработки. Delphi не является исключением. Выпущенная летом 1996 года обновленная версия Delphi 2.01 включает поддержку модулей сопряжения с Internet для Windows 95/NT – WinINET; возможность построения блоков расширения Microsoft Information Server через интерфейсы ISAPI & ISAPI Filter; 8 элементов ActiveX, полностью реализующих логику поддержки основных Internet-протоколов и HTML (обработка + отображение => построение браузеров) в виде повторно используемых компонентов. Отметим, что в 1999 году вышла версия Delphi 5.0, имеющая гораздо более мощные возможности.

С этих точек зрения, гибкость такого инструмента корпоративного разработчика, как Delphi, становится не менее важным фактором, чем возможность стандартизации бизнес-логики и организации бизнес-процессов.

## 1.5. Интерфейсы

В современных ОС существует понятие системных интерфейсов, обеспечивающих взаимодействие программ с различными видами сервиса – коммуникационного, информационного и др. По типу предоставляемого сервиса они могут быть разделены на прикладные программные интерфейсы (API – Application Program Interface), графические интерфейсы, телефонные API и ряд других.

Каждый интерфейс должен удовлетворять ряду требований (стандартам, определенным Международной организацией стандартизации – ISO). Более подробно остановимся на требованиях к интерфейсу API.

**API** – это интерфейс, связывающий **прикладное программное обеспечение** (собственно прикладные программы, данные, а также документация и средства обучения пользователей), использующее средства языков программирования, для вызова сервисных средств операционной системы (такой, например, как Windows). Эти средства могут включать процедуры или операции, объекты общих данных и др. Для поддержки приложений может требоваться широкий объем сервисных услуг API.

Информация в API определяется синтаксисом и семантикой специфического языка программирования так, чтобы пользователь этого языка мог обращаться к услугам, обеспеченным прикладной платформой. **Прикладная платформа** состоит из аппаратной платформы и программного обеспечения – операционной системы, компиляторов, СУБД, графических систем, т.е. всех средств, составляющих операционную среду для прикладных систем. Это подразумевает технические требования отображения функций, делающих доступными прикладной платформе синтаксис и семантику языка программирования, а также взаимодействие с внешней средой.

**Внешняя среда** включает в себя все системные компоненты, которые являются внешними по отношению к прикладной платформе и прикладному обеспечению – это утилиты и подсистемы, реализуемые на других (удалённых) платформах, а также периферийные устройства. Существует также понятие область **интерфейсов внешней среды** (EEI), описывающих взаимодействие между прикладной платформой и внешней средой.

Спецификации API документируют сервис и/или сервисные методы доступа, которые являются доступными в интерфейсе между приложением и прикладной платформой.

Спецификации API имеют следующие формы:

1. Спецификации программирования, которые являются описанием языка, определенного в пределах программы работы SC22, типа ФОРТРАНА, Ады и С.
2. Языково-независимые спецификации API, которые являются описанием набора функциональных возможностей в терминах

семантики (в абстрактном синтаксисе) и абстрактных типов данных, которые могут быть связаны с множеством языков программирования.

3. Языково-зависимые спецификации API, которые являются описанием набора функциональных возможностей в терминах синтаксиса и типов данных некоторого языка программирования.

Языково-независимые спецификации API нужны при определении технических требований для вызова услуг в API. Эти спецификации необходимы, прежде всего, для обслуживания различных платформ языков программирования. Однако должно также существовать большое количество привязок к языкам программирования типа КОБОЛА или С. Языково-зависимые спецификации API используются программистами, пишущими программы в специфическом языке программирования, для вызова сервиса, обеспечиваемого прикладной платформой. Они могут использоваться программами для вызова сервиса, предлагаемого другими прикладными программами (т.е. для обмена информацией между различными программами).

Концепция «Уровень Абстракции» является комплексной, с различными вариантами использования (по возможности не находящимися в противоречии). Использование «Уровня Абстракции» подразумевает вариации в количестве функциональных возможностей, предлагаемых программе при каждом запросе сервисных функций.

Одинаковый вид сервиса можно обеспечивать различными спецификациями API, которые отличаются по уровню абстракции. Например, менее абстрактные спецификации API для услуг X.400 электронной почты могут обеспечить прикладного программиста существенным контролем над подробностями его взаимодействия с серверами почты. С другой стороны, более абстрактные спецификации API могут обеспечить простой, единственный вызов подпрограммы для отправки файла как сообщения почты к почтовому ящику.

При этом более абстрактные спецификации API более легки для использования, чем менее абстрактные спецификации при условии, что принятые соглашения в осуществлении обслуживания соответствуют требованиям приложения. Менее абстрактные спецификации API используются там, где имеются специфические требования, касающиеся особенностей взаимодействия или выполнения программы.

Другое использование «Уровня Абстракции» отражает степень видимости/невидимости при выполнении запросов спецификации API.

Спецификации API могут отражать чистую абстракцию, управляемую только в соответствии со служебными требованиями (такими как, например, независимый сетевой протокол API), или отражать детали выполнения. Эти детали могут быть связаны с одним из нескольких альтернативных методов для служебного сервиса (например, OSI, TCP/IP или ISDN коммуникационного сервиса API). Подробности могут также отражать аспекты выполнения на альтернативных платформах ОС. В связи

с этим использование более абстрактных спецификаций API позволяет обеспечить большую быстроту и независимость выполнения, в то время как менее абстрактные спецификации API позволяют обеспечить больший контроль или большее качество сервиса.

Уровень абстракций спецификаций API изменяется с языком программирования и абстракциями, присущими определенному виду сервиса. Поэтому «однородный» уровень абстракций не соответствует спецификациям API.

Стандартные спецификации API определяют связь между языком программирования и особенностями специфического обслуживания, и обеспечивают доступ приложений, написанных в специфическом языке программирования, к обслуживанию.

Стандартные спецификации API могут быть частью стандарта, определенного языком программирования, связанного с определенным видом обслуживания или отдельным стандартом, связывающим язык программирования и определенный вид обслуживания. Таким образом, стандарты языка программирования могут рассматриваться как один из видов стандартных спецификаций API.

Определён ряд требований к спецификациям API:

1. Стандартные спецификации API должны идентифицировать стандарты, которые определяют язык программирования и обслуживание, связанное с ним, если они не определены стандартными техническими требованиями API непосредственно.
2. Стандартные спецификации API должны быть совместимы и должны избегать дублирования требований, связующих обслуживание и стандарт языка программирования.
3. Если в API ожидается поддержка большого числа функций сервиса языка программирования, то должны быть определены все требования по взаимодействию и совместимости между языком программирования и API, включая требования по обмену данными.
4. Если API должен поддерживать большое число функций сервиса для языка программирования, то должны быть определены все требования по взаимодействию и совместимости между API и языком программирования, включая требования по координации имен идентификаторов. То есть необходимо перечислить требования для правильного взаимодействия между сервисом, оказываемым языку программирования.
5. Если проект стандартизации спецификации API включает множество привязок к языку программирования с общими характеристиками интерфейса, должны использоваться языково-независимые спецификации API.
6. Языково-независимые технические требования API будут прогрессировать вместе, по крайней мере, с одной привязкой к языку, которая зависит от языково-независимых технических требований API.

8. API должен иметь соответствующие уровни соответствия языкам программирования или служебным спецификациям, с которыми он связан с помощью интерфейса.
9. Требования соответствия, указанные в стандартных спецификациях API, должны работать на разных аппаратных платформах и на соответствующих приложениях.
10. Спецификации API должны соответствовать требованиям контрольных тестов и быть прозрачны для проверок (контрольных тестов).

## 2. ОПЕРАЦИОННЫЕ СИСТЕМЫ СЕТЕЙ ЭВМ

### 2.1. Основные сетевые ОС

Большое разнообразие типов компьютеров, используемых в вычислительных сетях, влечет за собой разнообразие операционных систем: для рабочих станций, для серверов сетей уровня отдела и серверов уровня предприятия в целом. К ним могут предъявляться различные требования по производительности и функциональным возможностям, желательно, чтобы они обладали свойством совместимости, которое позволило бы обеспечить совместную работу различных ОС.

Сетевые ОС могут быть разделены на две группы: масштаба отдела и масштаба предприятия. ОС для отделов или **рабочих групп** (совокупности компьютеров, сгруппированных для удобства просмотра их сетевых ресурсов) обеспечивают набор сетевых сервисов, включая разделение файлов, приложений и принтеров. Они также должны обеспечивать свойства отказоустойчивости, например, работать с RAID-массивами, поддерживать кластерные архитектуры. Сетевые ОС отделов обычно более просты в установке и управлении по сравнению с сетевыми ОС предприятия, у них меньше функциональных свойств, они меньше защищают данные и имеют более слабые возможности по взаимодействию с другими типами сетей, а также худшую производительность.

Сетевая операционная система масштаба предприятия, прежде всего, должна обладать основными свойствами любых корпоративных продуктов, в том числе: масштабируемостью, то есть способностью одинаково хорошо работать в широком диапазоне различных количественных характеристик сети; совместимостью с другими продуктами, то есть способностью работать в сложной гетерогенной среде интерсети в режиме plug-and-play.

Корпоративная сетевая ОС должна поддерживать более сложные сервисы. Подобно сетевой ОС рабочих групп сетевая ОС масштаба предприятия должна позволять пользователям разделять файлы, приложения и принтеры, причем делать это для большего количества пользователей и объема данных и с более высокой производительностью.

Кроме того, сетевая ОС масштаба предприятия обеспечивает возможность соединения разнородных систем – как рабочих станций, так и серверов. Например, даже если ОС работает на платформе Intel, она должна поддерживать рабочие станции UNIX, работающие на RISC-платформах. Аналогично, серверная ОС, работающая на RISC-компьютере, должна поддерживать DOS, Windows и OS/2. Сетевая ОС масштаба предприятия должна поддерживать несколько стеков протоколов (таких как TCP/IP, IPX/SPX, NetBIOS, DECnet и OSI), обеспечивая простой доступ к удаленным ресурсам, удобные процедуры управления сервисами, включая агентов для систем управления сетью.

Важным элементом сетевой ОС масштаба предприятия является централизованная справочная служба, в которой хранятся данные о пользователях и разделяемых ресурсах сети. Такая служба, называемая также службой каталогов, обеспечивает единый логический вход пользователя в сеть и предоставляет ему удобные средства просмотра всех доступных ему ресурсов. Администратор, при наличии в сети централизованной справочной службы, избавлен от необходимости заводить на каждом сервере повторяющийся список пользователей, а значит, избавлен от большого количества рутинной работы и от потенциальных ошибок при определении состава пользователей и их прав на каждом сервере.

Важным свойством справочной службы является ее масштабируемость, обеспечиваемая распределенностью базы данных о пользователях и ресурсах.

Такие сетевые ОС, как Banyan Vines, Novell NetWare 4.x, IBM LAN Server, Sun NFS, Microsoft LAN Manager и Windows NT Server, могут служить в качестве операционной системы предприятия, в то время как ОС NetWare 3.x, Personal Ware, Artisoft LANtastic больше подходят для небольших рабочих групп.

Критериями для выбора ОС масштаба предприятия являются следующие характеристики:

- органичная поддержка многосерверной сети;
- высокая эффективность файловых операций;
- возможность эффективной интеграции с другими ОС;
- наличие централизованной масштабируемой справочной службы;
- хорошие перспективы развития;
- эффективная работа удаленных пользователей;
- разнообразные сервисы: файл-сервис (в т.ч. **предоставление**

**файлов в совместное использование** – способность использовать часть или всю свою локальную файловую систему совместно с другими компьютерами), принт-сервис, безопасность данных и **отказоустойчивость** (способность компьютера и операционной системы адекватно реагировать на катастрофические события – пропадания напряжения или отказ техники; обычно под отказоустойчивостью понимают способность системы продолжать функционировать без потери данных

или закрытие системы с перезапуском и последующим восстановлением всех процессов, присутствующих до момента аварии), архивирование данных, служба обмена сообщениями, разнообразные базы данных и другие;

разнообразные программно-аппаратные хост-платформы: IBM SNA, DEC NSA, UNIX;

разнообразные транспортные протоколы: TCP/IP, IPX/SPX, NetBIOS, AppleTalk;

поддержка многообразных операционных систем конечных пользователей: DOS, UNIX, OS/2, Mac;

поддержка сетевого оборудования стандартов Ethernet, Token Ring, FDDI, ARCnet;

наличие популярных прикладных интерфейсов и механизмов вызова удаленных процедур RPC;

возможность взаимодействия с системой контроля и управления сетью, поддержка стандартов управления сетью SNMP.

Конечно, ни одна из существующих сетевых ОС не отвечает в полном объеме перечисленным требованиям, поэтому выбор сетевой ОС, как правило, осуществляется с учетом производственной ситуации и опыта. В таблице 2.1 приведены основные характеристики популярных и доступных в настоящее время сетевых ОС.

Таблица 2.1

Тип сетевой ОС	Характеристика сетевой ОС
Novell NetWare 4.1	<p>Специализированная операционная система, оптимизированная для работы в качестве файл-ового сервера и принт-сервера.</p> <p>Ограниченные средства для использования в качестве сервера приложений: не имеет средств виртуальной памяти и вытесняющей многозадачности, а поддержка симметричного мультипроцессирования отсутствовала до самого недавнего времени.</p> <p>Отсутствуют API основных операционных сред, используемых для разработки приложений, – UNIX, Windows, OS/2.</p>

Тип сетевой ОС	Характеристика сетевой ОС
	<p>Серверные платформы: компьютеры на основе процессоров Intel, рабочие станции RS/6000 компании IBM под управлением операционной системы AIX с помощью продукта NetWare for UNIX.</p> <p>Поставляется с оболочкой для клиентов: DOS, Macintosh, OS/2, UNIX, Windows (оболочка для Windows NT разрабатывается компанией Novell в настоящее время, хотя Microsoft уже реализовала клиентскую часть NetWare в Windows NT).</p> <p>Организация одноранговых связей возможна с помощью ОС PersonalWare.</p> <p>Имеет справочную службу NetWare Directory Services (NDS), поддерживающую централизованное управление, распределенную, полностью реплицируемую, автоматически синхронизируемую и обладающую отличной масштабируемостью.</p> <p>Поставляется с мощной службой обработки сообщений Message Handling Service (MHS), полностью интегрированную (начиная с версии 4.1) со справочной службой.</p> <p>Поддерживаемые сетевые протоколы: TCP/IP, IPX/SPX, NetBIOS, AppleTalk.</p> <p>Поддержка удаленных пользователей: ISDN, коммутируемые телефонные линии, frame relay, X.25 – с помощью продукта NetWare Connect (поставляется отдельно).</p> <p>Безопасность: аутентификация с помощью открытых ключей метода шифрования RSA (<b>шифрование данных</b> – изменение формата данных так, чтобы их нельзя было прочесть); сертифицирована по уровню C2.</p> <p>Хороший сервер коммуникаций.</p> <p>Встроенная функция компрессии диска.</p> <p>Сложное обслуживание.</p>

Тип сетевой ОС	Характеристика сетевой ОС
<p>Banyan VINES 6.0 и ENS (Enterprise Network Services) 6.0</p>	<p>Серверные платформы ENS for UNIX: работает на RISC-компьютерах под управлением SCO UNIX, HP-UX, Solaris, AIX ENS for NetWare: работает на Intel-платформах под управлением NetWare 2.x, 3.x, 4.x</p> <p>VINES работает на Intel-платформах.</p> <p>Клиентские платформы: DOS, Macintosh, OS/2, UNIX, Windows for Workgroups, Windows NT.</p> <p>Хороший сервер приложений: поддерживаются вытесняющая многозадачность, виртуальная память и симметричное мультипроцессирование в версии VINES и в ENS-версиях для UNIX.</p> <p>Поддерживаются прикладные среды UNIX, OS/2, Windows.</p> <p>Поддержка одноранговых связей – отсутствует.</p> <p>Справочная служба – Streetwork III, наиболее отработанная из имеющихся на рынке, с централизованным управлением, полностью интегрированная с другими сетевыми службами, распределенная, реплицируемая и автоматически синхронизируемая, отлично масштабируемая.</p> <p>Согласованность работы с другими сетевыми ОС: хорошая; серверная оболочка работает в средах NetWare и UNIX; пользователи NetWare, Windows NT и LAN Server могут быть объектами справочной службы Streetwork III.</p> <p>Служба сообщений – Intelligent Messaging, интегрирована с другими службами.</p> <p>Поддерживаемые сетевые протоколы: VINES IP, TCP/IP, IPX/SPX, Appletalk.</p> <p>Поддержка удаленных пользователей: ISDN, коммутируемые телефонные линии, X.25.</p> <p>Служба безопасности: поддерживает электронную подпись (собственный алгоритм), избирательные права доступа, шифрацию; не сертифицирована.</p>

Тип сетевой ОС	Характеристика сетевой ОС
	<p>Простое обслуживание.</p> <p>Хорошо масштабируется.</p> <p>Отличная производительность обмена данными между серверами, хуже при обмене сервер-ПК.</p>
Microsoft LAN Manager	<p>Широкая распространенность, работает под OS/2 и UNIX, поддерживает мощные серверные платформы, один сервер может поддерживать до 2000 клиентов.</p>
Microsoft Windows NT Server 3.51 и 4.0	<p>Серверные платформы: компьютеры на базе процессоров Intel, PowerPC, DEC Alpha, MIPS.</p> <p>Клиентские платформы: DOS, OS/2, Windows, Windows for Workgroups, Macintosh.</p> <p>Организация одноранговой сети возможна с помощью Windows NT Workstation и Windows for Workgroups.</p> <p>Windows NT Server представляет собой отличный сервер приложений: он поддерживает в вытесняющую многозадачность (вытесняющая или <b>приоритетная многозадачность</b> – это принцип приоритетов, позволяющий приложениям с более высоким приоритетом вытеснять приложения, имеющие более низкий приоритет. Так как система всегда контролирует события, процессорное время используется эффективнее, а сбойное приложение не приведет к зависанию системы), виртуальную память и симметричное мультипроцессирование, а также прикладные среды DOS, Windows, OS/2, POSIX.</p> <p>Справочные службы: доменная для управления учетной информацией пользователей (Windows NT Domain Directory service), справочные службы имен WINS и DNS.</p> <p>Хорошая поддержка совместной работы с сетями NetWare: поставляется клиентская часть (редиректор) для сервера NetWare (версий 3.x и 4.x в режиме эмуляции 3.x, справочная служба NDS поддерживается начиная с версии 4.0), выполненная в виде шлюза в Windows NT Server или как отдельная компонента для Windows NT</p>

Тип сетевой ОС	Характеристика сетевой ОС
	<p>Workstation; недавно Microsoft объявила о выпуске серверной части NetWare как оболочки для Windows NT Server.</p> <p>Служба обработки сообщений – Microsoft Mail, основанная на DOS-платформе, в ближайшее время ожидается версия для платформы Windows NT – Microsoft Message Exchange, интегрированная с остальными службами Windows NT Server.</p> <p>Поддерживаемые сетевые протоколы: TCP/IP, IPX/SPX, NetBEUI, AppleTalk.</p> <p>Поддержка удаленных пользователей: ISDN, коммутируемые телефонные линии, frame relay, X.25 – с помощью встроенной подсистемы Remote Access Server (RAS).</p> <p>Служба безопасности: мощная, использует и собирает <b>права доступа</b> (разрешения процессу определённым образом воздействовать на определённый объект. Различные типы объектов поддерживают различные права доступа, хранящиеся в списках контроля доступа) и доверительные отношения между <b>доменами</b> (объединения нескольких компьютеров, использующих единую базу учетных записей и <b>политику безопасности</b> (состоит из политики ведения учётных записей, политики привилегий пользователей, политики аудита и политики доверительных отношений); домены имеют уникальное имя); узлы сети, основанные на Windows NT Server, сертифицированы по уровню C2.</p> <p>Простота установки и обслуживания.</p> <p>Отличная масштабируемость.</p>
IBM LAN Server 4.0	<p>Серверные платформы: операционные системы MVS и VM для мэйнфреймов; AS/400 с OS/400, рабочие станции RS/6000 с AIX, серверы Intel 486 или Pentium под OS/2.</p> <p>Поставляется с оболочками для клиентов: DOS, Macintosh, OS/2, Windows, Windows NT, Windows for Workgroups.</p>

Тип сетевой ОС	Характеристика сетевой ОС
	<p>Серверы приложений могут быть организованы с помощью LAN Server 4.0 в операционных средах MVS, VM, AIX, OS/2, OS/400. В среде OS/2 поддерживаются: вытесняющая многозадачность, виртуальная память и симметричное мультипроцессирование.</p> <p>Организация одноранговых связей возможна с помощью ОС Warp Connect.</p> <p>Справочная служба – LAN Server Domain, то есть основа на доменном подходе.</p> <p>Поддерживаемые сетевые протоколы: TCP/IP, NetBIOS, AppleTalk.</p> <p>Безопасность – избирательные права доступа, система не сертифицирована.</p> <p>Служба обработки сообщений – отсутствует.</p> <p>Высокая производительность.</p> <p>Недостаточная масштабируемость.</p>
IBM и NCR LAN Manager	<p>LAN Manager for UNIX хорошо распространена (15% объема мировых продаж сетевых ОС).</p> <p>LAN Manager for AIX поддерживает RISC-компьютеры System/6000 в качестве файлового сервера.</p> <p>Работает под UNIX, имеет все преимущества, связанные с использованием этой ОС</p>

## 2.2. Windows NT

Windows NT является 32-разрядной операционной системой с приоритетной многозадачностью. В качестве фундаментальных компонентов в состав операционной системы входят средства обеспечения безопасности и развитый сетевой сервис. Windows NT обеспечивает совместимость со многими другими операционными и файловыми системами, а также сетями. Windows NT способна функционировать как на компьютерах, оснащенных CISC-процессорами со сложной системой команд (complex instruction set computing), так и на компьютерах с RISC-процессорами, имеющими сокращенный набор инструкций (reduced instruction set computing). Операционная система Windows NT также поддерживает высокопроизводительные системы с симметричной мультипроцессорной конфигурацией.

Система Windows NT не является дальнейшим развитием ранее

существовавших продуктов. Стремясь обеспечить *совместимость* (compatibility) новой операционной системы, разработчики Windows NT сохранили привычный интерфейс Windows и реализовали поддержку существующих файловых систем (таких как FAT. **Файловая система FAT** базируется на таблице размещения файлов, обслуживаемой ОС и отслеживающей состояние различных сегментов диска, используемых для хранения файлов) и различных приложений (написанных для MS-DOS, OS/2, Windows 3.x и POSIX). Разработчики также включили в состав Windows NT средства работы с различными сетевыми средами. Достигнута *переносимость* (portability) системы, которая может теперь работать как на CISC, так и на RISC-процессорах. К CISC относятся Intel-совместимые процессоры 80386 и выше; RISC представлены системами с процессорами MIPS R4000 или Digital Alpha AXP. *Масштабируемость* (scalability) означает, что Windows NT не привязана к однопроцессорной архитектуре компьютеров, а способна полностью использовать возможности, предоставляемые симметричными мультипроцессорными системами. В настоящее время Windows NT может функционировать на компьютерах с числом процессоров от 1 до 32. Кроме того, в случае усложнения стоящих перед пользователями задач и расширения предъявляемых к компьютерной среде требований, Windows NT позволяет легко добавлять более мощные и производительные серверы и рабочие станции к корпоративной сети. Дополнительные преимущества дает использование единой среды разработки и для серверов, и для рабочих станций. Windows NT имеет однородную *систему безопасности* (security), удовлетворяющую спецификациям правительства США. В корпоративной среде критическим приложениям обеспечивается полностью изолированное окружение. *Распределенная обработка* (distributed processing) означает, что Windows NT имеет встроенные в систему сетевые возможности. Windows NT также позволяет обеспечить связь с различными типами хост-компьютеров благодаря поддержке разнообразных транспортных протоколов и использованию средств «клиент-сервер» высокого уровня, включая именованные каналы, вызовы удаленных процедур (RPC – remote procedure call) и Windows-сокеты. *Надежность и отказоустойчивость* (reliability and robustness) обеспечиваются архитектурными особенностями, которые защищают прикладные программы от повреждения друг другом и операционной системой. Windows NT использует отказоустойчивую структурированную обработку особых ситуаций на всех архитектурных уровнях, которая включает восстанавливаемую **файловую систему NTFS** (улучшенная файловая система, разработанная специально для Windows NT. Поддерживает средства восстановления файловой системы, допускает использование чрезвычайно больших носителей данных, а также различных функций подсистемы POSIX. Поддерживает объектно-ориентированные приложения, обрабатывая все файлы как объекты с определяемой пользователем системой атрибутами) и обеспечивает защиту с помощью

встроенной системы безопасности и усовершенствованных методов управления памятью. Возможности *локализации* (localization) предоставляют средства для работы во многих странах мира на национальных языках, что достигается применением стандарта ISO Unicode (разработан Международной организацией по стандартизации). Благодаря модульному построению системы обеспечивается *расширяемость* (extensibility) Windows NT, что, как будет показано в следующем разделе, позволяет гибко осуществлять добавление новых модулей на различные уровни операционной системы.

### 2.2.1. Архитектурные модули Windows NT

Как показано на рис. 2.1, Windows NT представляет из себя модульную (более совершенную, чем монолитная) операционную систему, которая состоит из отдельных взаимосвязанных, относительно простых модулей. Основными модулями Windows NT являются (перечислены в

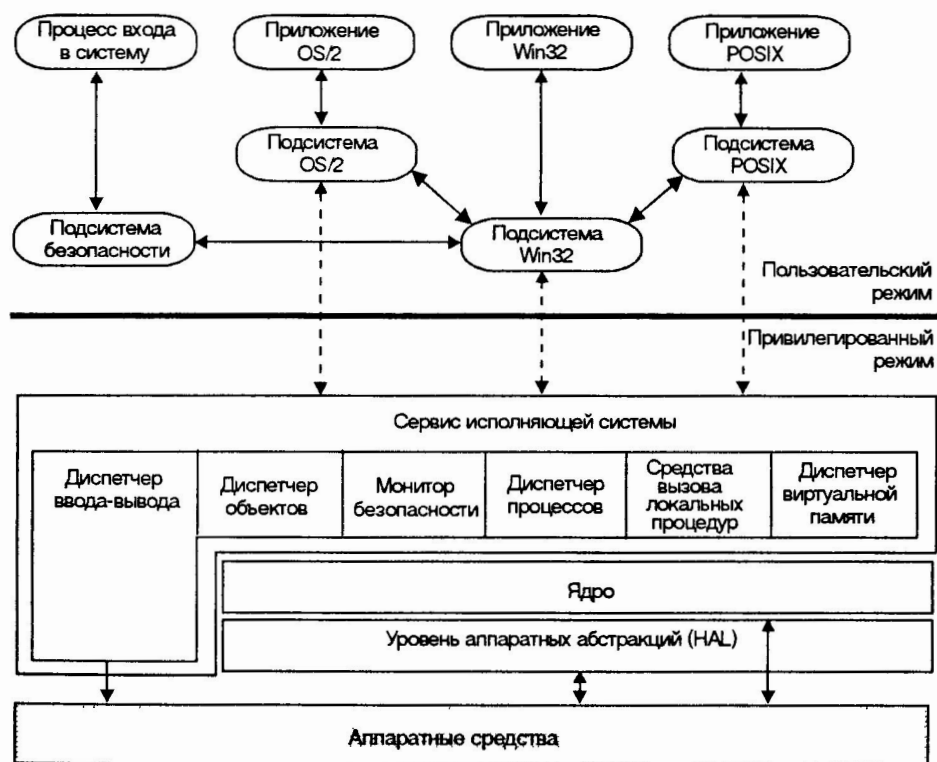


Рис. 2.1

порядке следования от нижнего уровня архитектуры к верхнему): 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 объему 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), предоставляющих базовый сервис операционной системы подсистемам среды. Исполняющая система состоит из нескольких компонентов; каждая из них предназначена для поддержки определенного системного сервиса. Так, один из компонентов – монитор безопасности (Security Reference Monitor) – функционирует совместно с защищенными подсистемами и обеспечивает реализацию модели безопасности системы. *Подсистемы среды* представляют собой защищенные 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.

### **2.2.2. Уровень аппаратных абстракций**

Уровень аппаратных абстракций (HAL) представляет собой создаваемый производителями аппаратных средств слой программного обеспечения, который скрывает (или абстрагирует) особенности и различия аппаратуры от верхних уровней операционной системы. При создании уровня аппаратных абстракций ставилась задача подготовки процедур, которые позволяли бы единственному драйверу конкретного устройства поддерживать функционирование этого устройства для всех платформ. HAL ориентирован на большое число разновидностей аппаратных платформ с однопроцессорной архитектурой; таким образом, для каждого из аппаратных вариантов не требуется отдельной версии операционной системы.

Процедуры HAL вызываются как средствами операционной системы (включая ядро), так и драйверами устройств. При работе с драйверами устройств уровень аппаратных абстракций обеспечивает

поддержку различных технологий ввода-вывода (вместо традиционной ориентации на одну аппаратную реализацию или требующей значительных затрат адаптации под каждую новую аппаратную платформу). Уровень аппаратных абстракций позволяет также «скрывать» от остальных уровней операционной системы особенности аппаратной реализации симметричных мультипроцессорных систем.

### 2.2.3. Ядро

Ядро (Kernel) является «сердцем» Windows NT и работает в тесном контакте с уровнем аппаратных абстракций. Этот модуль, в первую очередь, занимается планированием действий компьютерного процессора. В случае если компьютер содержит несколько процессоров, ядро синхронизирует их работу с целью достижения максимальной производительности системы. Ядро осуществляет диспетчеризацию *нитей управления* (threads – иногда называются подзадачами, ответвлениями или потоками), которые являются основными объектами в планируемой системе. Нити управления определяются в контексте процесса; процесс включает адресное пространство, набор доступных процессу объектов и совокупность выполняемых в контексте процесса нитей управления. Объектами являются управляемые операционной системой ресурсы. Ядро производит диспетчеризацию нитей управления таким образом, чтобы максимально загрузить процессоры системы и обеспечить первоочередную обработку нитей с более высоким приоритетом. (Всего существует 32 значения приоритета, которые сгруппированы в два класса приоритетов: real-time и variable). Подобный подход позволяет достичь максимальной эффективности операционной системы.

Подкомпоненты исполняющей системы, такие как диспетчер ввода-вывода и диспетчер процессов, используют ядро для синхронизации действий. Они также взаимодействуют с ядром для более высоких уровней абстракции, называемых *объектами ядра*; некоторые из этих объектов экспортируются внутри пользовательских вызовов интерфейса прикладных программ (API).

Ядро управляет двумя типами объектов:

1. *Объекты диспетчеризации* (dispatcher objects) характеризуются сигнальным состоянием (signaled или nonsignaled) и управляют диспетчеризацией и синхронизацией системных операций. Эти объекты включают события, мутанты, мутэксы, семафоры, нити управления и таймеры (events, mutants, mutexes, semaphores, threads, timers).
2. *Управляющие объекты* (control objects) используются для операций управления ядра, но не воздействуют на диспетчеризацию или синхронизацию. Управляющие объекты включают в себя асинхронные вызовы процедур, прерывания, уведомления

и состояния источника питания, процессы и профили (asynchronous procedure calls, interrupts, power notifies, power statuses, processes, profiles). Существует также понятие **локального профиля** – файла, содержащего информацию о параметрах окружения пользователя, например о сетевых подключениях, группах программ, положении и размерах окон, сохраняемых при выходе из ОС. Локальные профили сохраняются там же, где и база данных реестра. **Персональный профиль пользователя** создаётся администратором и назначается одному пользователю. В персональный профиль записываются все изменения, сделанные пользователем в течение сеанса работы. Эти изменения сохраняются при выходе пользователя из системы. При последующей регистрации на любой рабочей станции загружается персональный профиль (файл с расширением \*.USR) и выставляются сохранённые параметры окружения.

В табл. 2.2 показано, каким образом операционная система использует каждый тип объекта диспетчеризации, а в табл. 2.3 показано, каким образом операционная система использует каждый тип управляющего объекта.

В основном, ядро не обеспечивает проведение в жизнь какой-либо политики, так как за это отвечает исполняющая система. Однако ядро производит формирование политики по перемещению процессов из памяти. Ядро выполняется полностью в привилегированном режиме и неперемещаемо (nonpagable) в памяти. Программное обеспечение ядра не является выгружаемым (preemptible), и, следовательно, для него не может производиться переключение контекста (context-switched); большая часть программного обеспечения вне ядра почти всегда может быть выгружена и использует переключение контекста. Ядро может выполняться одновременно на всех процессорах в мультипроцессорной конфигурации, соответствующим образом синхронизируя доступ к критическим областям.

#### 2.2.4. Исполняющая система Windows NT

Исполняющая система (Executive), в состав которой входят ядро и уровень аппаратных абстракций HAL, обеспечивает общий сервис системы, который могут использовать все подсистемы среды. Каждая группа сервиса находится под управлением одной из отдельных составляющих исполняющей системы: диспетчера объектов (Object Manager); диспетчера виртуальной памяти (Virtual Memory Manager); диспетчера процессов (Process Manager); средств вызова локальных процедур (Local Procedure Call Facility); диспетчера ввода-вывода (I/O Manager); монитора безопасности (Security Reference Monitor). Монитор безопасности совместно с процессом входа в систему (Logon) и защищенными подсистемами реализует модель безопасности Windows NT.

Таблица 2.2

## Объекты диспетчеризации

Тип объекта	Описание
Event	Используется для записи местонахождения события и синхронизации его с некоторым выполняемым действием.
Mutant	Один из двух объектов, используемых ядром для контроля над общим монопольным доступом к ресурсу. Этот тип объекта применяется для обеспечения в пользовательском режиме механизма взаимного исключения, который имеет семантику монопольного использования. Может также использоваться в привилегированном режиме.
Mutex	Второй объект, используемый ядром для контроля над общим монопольным доступом к ресурсу. Этот тип объекта может быть использован только в режиме ядра и предназначен для обеспечения безтупикового механизма взаимного исключения с семантикой монопольного использования и другой специальной семантикой системы.
Semaphore	Используется для управления доступом к ресурсу, но не обязательно в режиме взаимного исключения. Объект Semaphore действует как клапан, через который может проходить одновременно некоторое число нитей управления (до определенного ограничения). Клапан открыт (состояние signaled) до тех пор, пока имеются доступные ресурсы. Когда число используемых ресурсов достигает ограничивающей отметки, клапан закрывается (состояние non signaled).
Thread	Выполняет программный код и управляется ядром. Каждая нить управления связана с объектом процесса, который определяет распределение виртуального адресного пространства для нити и собирает результаты выполнения нитей. Несколько объектов Thread могут быть связаны с одиночным объектом процесса, который допускает параллельное выполнение нитей управления в одиночном адресном пространстве (возможно одновременное выполнение в многопроцессорной системе).
Timer	Используется для фиксирования временных интервалов и прерывания (по тайм-ауту) операций.

Таблица 2.3

## Управляющие объекты

Тип объекта	Описание
Asynchronous Procedures Call	Используется для прерывания выполнения специфицированной нити. Procedure Call управления и передачи управления вызываемой процедуре в определенном режиме процессора.
Interrupt	Используется для соединения источника прерывания и процедуры обслуживания прерывания через элемент таблицы управления прерыванием (IDT – Interrupt Dispatch Table). Каждый процессор имеет IDT, которая управляет прерываниями, происходящими в процессоре.
Process	Используется для представления пространства виртуальных адресов и управляющей информации, необходимой для выполнения набора объектов нити управления. Объект процесса содержит указатель на карту адресов, список готовых нитей управления, содержащих объекты нитей управления, пока процесс не находится в критическом состоянии, список принадлежащих объекту нитей управления, общее накопленное время для выполнения нитей управления процесса, базовый приоритет и свойства нити управления по умолчанию.
Profile	Используется для определения распределения времени выполнения внутри блока кода. Может применяться для пользовательского или системного кода.

Верхний уровень исполняющей системы называется системным сервисом (System Services). Показанный на рис. 2.2 системный сервис представляет собой интерфейс между подсистемами среды пользовательского режима и привилегированным режимом. Далее коротко описывается назначение каждой составляющей исполняющей системы (монитор безопасности описан позже).

Объектами являются отдельные элементы времени выполнения, имеющие объектный тип; управление этими элементами могут производить процессы операционной системы. Тип объекта включает определенный системой тип данных, список операций, которые могут выполняться над ним (например, wait, create или cancel), и набор

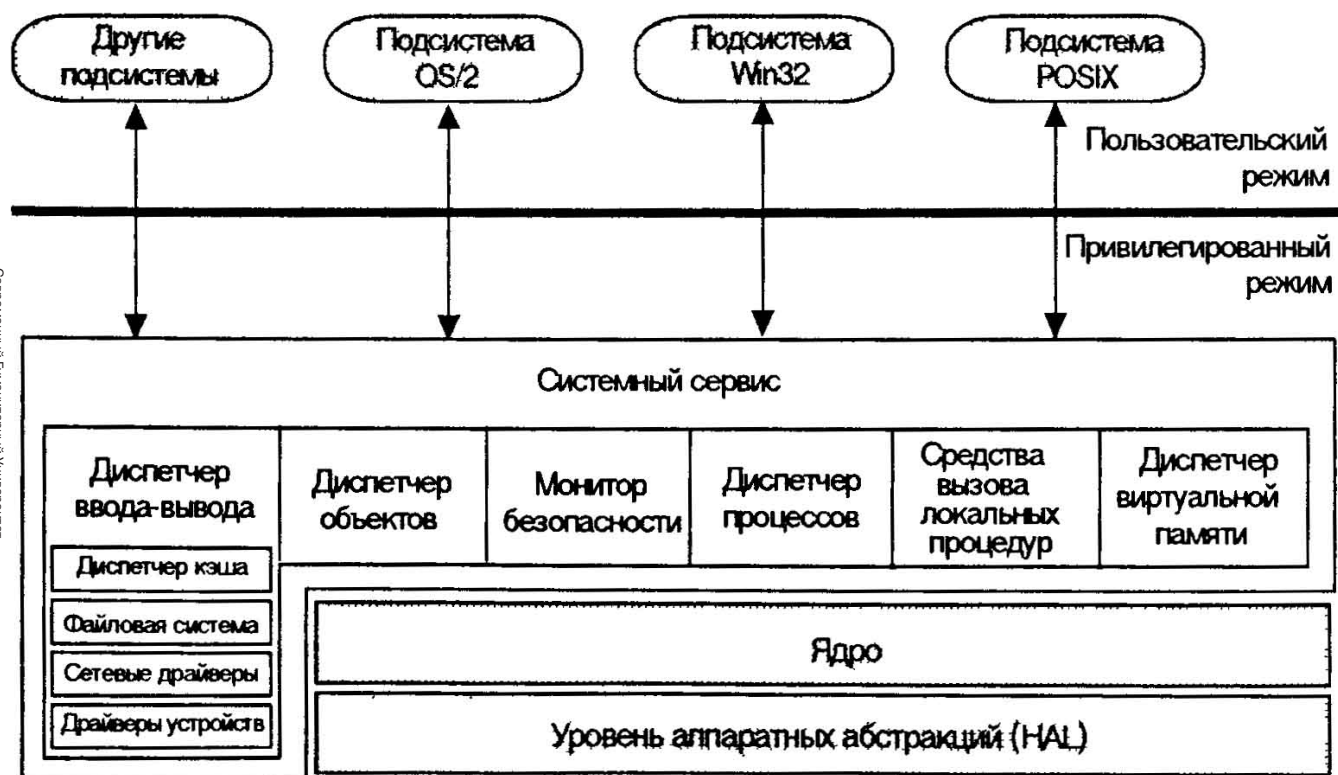


Рис. 2.2

атрибутов объекта. Диспетчер объектов обеспечивает унифицированные правила хранения, именования и безопасности объектов. Прежде чем процесс сможет управлять объектом Windows NT, он должен получить описатель объектов (object handle) через диспетчер объектов. Описатель объектов включает информацию управления доступом и непосредственно указатель на объект. Все описатели объектов создаются через диспетчер объектов. Кроме того, диспетчер объектов управляет глобальным пространством имен (namespace) для Windows NT и следит за созданием и использованием объектов любым процессом. Пространство адресов используется для доступа ко всем именованным объектам, которые содержатся в локальной компьютерной среде. Ниже представлен список объектов, которые могут иметь имена: объекты каталога (directory objects); объекты типа объекта (object type objects); символические объекты связи (symbolic link objects); объекты семафора и события (semaphore objects, event objects); объекты процесса и нитей управления (process objects, thread objects); объекты раздела и сегмента (section objects, segment objects); объекты порта (port objects); объекты файла (file objects).

Диспетчер процессов – компонент, который отслеживает два типа объектов: объекты процесса и объекты нитей управления. Процесс определяется как адресное пространство, набор доступных процессу объектов и совокупность выполняемых в контексте процесса нитей управления. Нить управления (thread) является основным управляемым элементом в системе. Она имеет собственный набор регистров, собственный стек ядра, блок среды нити и стек пользователя в адресном пространстве процесса.

Диспетчер процессов – компонент Windows NT, который управляет созданием и завершением процессов. Он обеспечивает набор стандартных услуг по созданию и использованию нитей управления и процессов в контексте специфической среды подсистемы. Кроме того, диспетчер процессов в некоторой степени диктует правила для нитей и процессов. Диспетчер процессов не налагает каких-либо требований по иерархии или группировке для процессов, а также не определяет отношений порожденности. Модель процессов Windows NT работает совместно с моделью безопасности и диспетчером виртуальной памяти для обеспечения безопасности процессов. Каждому процессу назначается маркер безопасного доступа (security access token), называемый первичным маркером процесса. Этот маркер используется процедурами проверки правильности доступа Windows NT, когда нити управления процесса ссылаются на защищенные объекты.

Архитектура памяти для Windows NT основана на использовании подкачиваемой по запросу виртуальной памяти системы и плоском, линейном адресном пространстве с 32-разрядным доступом. Виртуальная память (virtual memory) позволяет операционной системе управлять большим объемом памяти, чем тот объем, который компьютер

физически содержит. Каждый процесс размещается в уникальном виртуальном адресном пространстве, которое представляет собой набор адресов, доступных для использования нитями управления процесса. Это виртуальное адресное пространство разделяется на равные блоки, или страницы (pages) – см. рис. 2.3. Каждый процесс может использовать до 4 Гб собственного виртуального адресного пространства; из них 2 Гб зарезервированы для нужд программы, а оставшиеся 2 Гб – для системы. Windows NT может использовать до 4 Гб физической памяти, если аппаратные средства компьютера могут обеспечить подобный объем. Лишь некоторые операционные системы позволяют работать с памятью таких размеров.

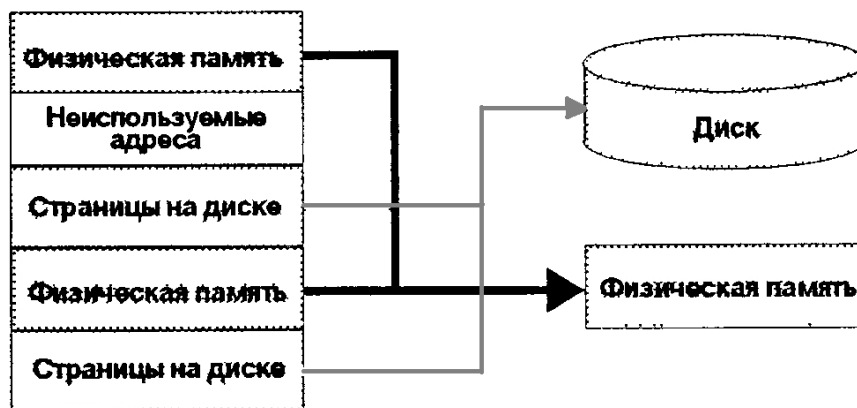


Рис. 2.3

Подкачка по запросу (demand paging) использует метод, посредством которого данные странично переносятся из физической памяти во временный страничный файл на диске. В случае необходимости использования этих данных для функционирования определенных процессов страничные данные переписываются обратно в физическую память. Диспетчер виртуальной памяти отображает виртуальные адреса в адресном пространстве процесса на физические страницы в памяти компьютера. При этом от нитей управления процесса скрывается физическая организация памяти. Это гарантирует, что нить управления может обращаться, в случае необходимости, к памяти своего процесса, не затрагивая память других процессов. Следовательно, просмотр виртуальной памяти процесса нитью управления намного упрощен по сравнению с реальным расположением страниц в физической памяти. Вследствие того что каждый процесс имеет отдельное адресное пространство, нити управления одного процесса не могут просматривать или изменять память другого процесса без соответствующего разрешения.

Приложения и подсистемы среды реализуют взаимоотношения типа «клиент-сервер». Это означает, что клиент (приложение) обращается к серверу среды (подсистеме) для удовлетворения запроса о предоставлении некоторого типа сервиса системы. Исполняющая система предоставляет средства прохождения сообщений, которые называются средствами вызова локальных процедур (LPC – Local Procedure Call). Они функционируют подобно вызовам удаленных процедур (RPC), используемым для работы в сетевой среде. Однако средства LPC оптимизированы для процессов, выполняющихся на одном компьютере. Прикладные программы взаимодействуют с подсистемами среды, передавая сообщения через средства LPC. Процесс прохождения сообщения скрыт от клиентского приложения функциональными заглушками (stubs); заглушки представляют собой невыполняемые фрагменты, которые используются при обращении к серверам среды. Заглушки реализованы в форме специальных динамически связываемых библиотек (DLL), как показано на рис. 2.4. Когда приложение производит обращение к интерфейсу прикладных программ (API – application program interface) подсистемы среды, заглушка клиентского процесса (приложения) упаковывает параметры для вызова и направляет их серверному процессу (подсистеме), который осуществляет выполнение. Средства LPC предусматривают, что после передачи данных серверу производится ожидание ответа.

Основное назначение диспетчера ввода-вывода – управление связью между драйверами. Диспетчер ввода-вывода поддерживает все драйверы файловой системы, драйверы аппаратных средств, сетевые драйверы и обеспечивает для них гетерогенную среду. Он предоставляет формальный интерфейс, доступный для вызовов всеми драйверами, что позволяет ему одинаково взаимодействовать со всеми драйверами, без какой-либо информации о фактическом управлении работой устройства.

Модель ввода-вывода Windows NT использует многоуровневую архитектуру, которая позволяет отдельным драйверам отвечать за логически законченный уровень обработки ввода-вывода. Например, драйверы самого низкого уровня управляют физическими устройствами компьютера (называются драйверами устройств – device drivers). Другие драйверы являются надстройкой к драйверам устройств, как показано на рис. 2.5. Драйверам более высокого уровня неизвестны любые подробности работы физических устройств. С помощью диспетчера ввода-вывода драйверы более высокого уровня просто передают запросы логического ввода-вывода драйверам устройств, которые и обращаются к обслуживаемым ими физическим устройствам. Устанавливаемые файловые системы Windows NT и сетевые редиректоры (redirectors) – примеры работающих таким образом драйверов высокого уровня. Использование подобной схемы обеспечивает легкую замену драйверов файловой системы и драйверов устройств. Кроме того, это

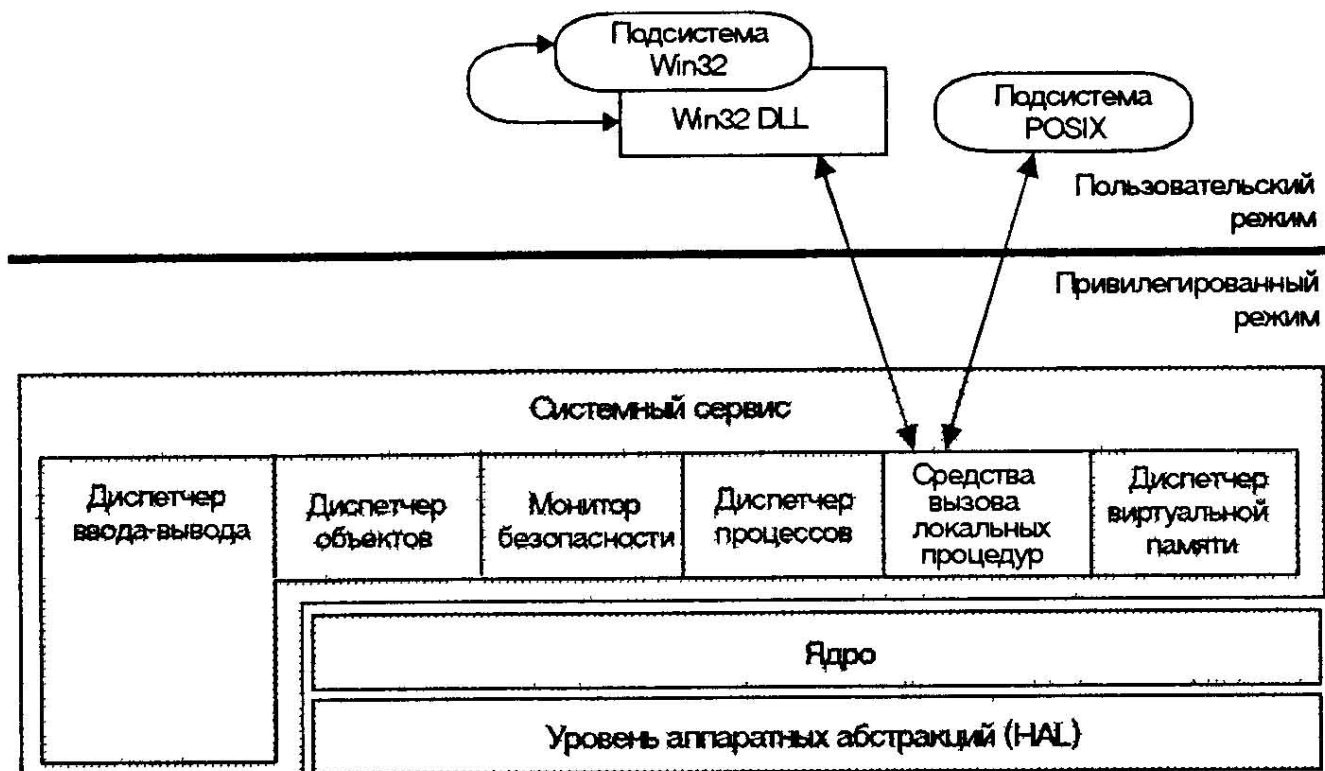


Рис. 2.4

позволяет быть активными одновременно нескольким файловым системам и устройствам, так как они адресуются через формальный интерфейс.

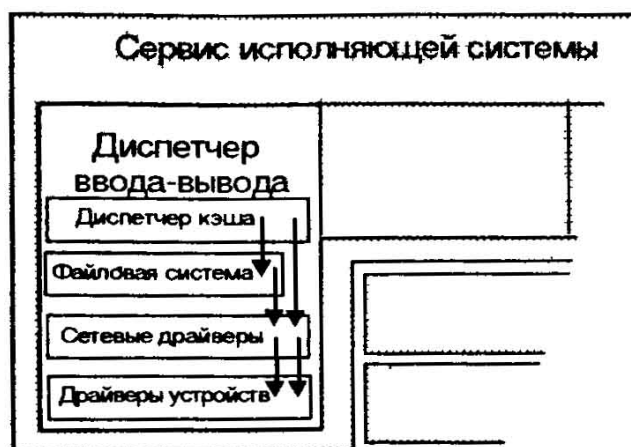


Рис. 2.5

Драйверы взаимодействуют друг с другом, используя структуры данных, называемые пакетами запроса ввода-вывода (I/O request packets). Самый простой способ выполнения операций ввода-вывода состоит в том, чтобы синхронизировать выполнение приложений с завершением запрашиваемых ими операций ввода-вывода (такой подход известен под названием синхронного ввода-вывода – synchronous I/O). Когда подобное приложение выполняет операцию ввода-вывода, функционирование собственно приложения блокировано. После завершения операции ввода-вывода приложению разрешается продолжение дальнейшего выполнения. Одним из способов оптимизации эффективности приложений является применение асинхронного ввода-вывода (asynchronous I/O); этот метод используется многими процессами в Windows NT. Когда приложение инициализирует операцию ввода-вывода, диспетчер ввода-вывода принимает запрос, но не блокирует работу приложения в процессе выполнения ввода-вывода. Вместо этого приложение продолжает свое функционирование. Большинство устройств ввода-вывода работает очень медленно в сравнении с процессором компьютера; таким образом, прикладная программа может выполнить множество операций в процессе ожидания завершения операции ввода-вывода. Когда подсистема среды выдает асинхронный запрос ввода-вывода, диспетчер ввода-вывода возвращается к подсистеме среды немедленно после помещения запроса в очередь, без ожидания завершения операции драйвером

устройства. В это время отдельная нить управления диспетчера ввода-вывода выполняет запросы из очереди наиболее эффективным образом (не обязательно в порядке поступления) (рис. 2.6).



Рис. 2.6

По завершении любого запроса ввода-вывода диспетчер ввода-вывода уведомляет об этом процесс, запросивший операцию. Так как применение асинхронного ввода-вывода разрешает приложению использовать процессор компьютера во время операций ввода-вывода, это затрудняет для приложения определение завершения операции ввода-вывода. Некоторые приложения применяют функцию повторного вызова (APC), которая вызывается после завершения асинхронной операции ввода-вывода. Другие приложения используют объекты синхронизации, типа событий или описателей файлов, которые система ввода-вывода приводит в соответствующее состояние после выполнения ввода-вывода.

Архитектура ввода-вывода содержит единственный диспетчер кэша (Cache Manager), который осуществляет кэширование для всей системы ввода-вывода. Кэширование (caching) – метод, используемый файловой системой для увеличения эффективности. Вместо непосредственной записи и считывания с диска, часто используемые файлы временно сохраняются в кэш-памяти; таким образом, работа с этими файлами

выполняется в памяти. Операции с данными, находящимися в памяти, производятся значительно быстрее операций с данными на диске. Диспетчер кэша использует модель отображения файла, которая интегрирована с диспетчером виртуальной памяти Windows NT. В зависимости от объема доступной оперативной памяти диспетчер кэша может динамически увеличивать или уменьшать размер кэша. Когда процесс открывает файл, который уже находится в кэше, диспетчер кэша просто копирует данные из кэша в виртуальное адресное пространство. Диспетчер кэша поддерживает службы типа ленивой записи (lazy write) и ленивой фиксации (lazy commit), которые могут значительно увеличить эффективность файловой системы. В процессе ленивой записи изменения регистрируются в кэше файловой структуры, обеспечивающем более быстрый доступ. Позднее, когда загрузка центрального процессора снижена, диспетчер кэша заносит изменения на диск. Ленивая фиксация подобна ленивой записи. Вместо немедленной маркировки транзакции как успешно завершившейся, переданная информация кэшируется и позднее в фоновом режиме записывается в журнал файловой системы.

Третьим типом драйверов, присутствующих в качестве компонента в архитектуре ввода-вывода, являются сетевые драйверы. Windows NT включает интегрированные возможности работы с сетями и поддержку для распределенных приложений. Как показано на рис. 2.7, работа с сетями обеспечивается рядом сетевых драйверов.

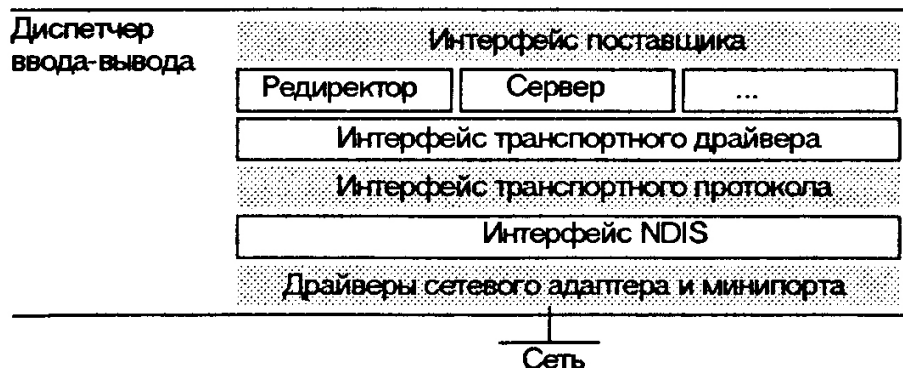


Рис. 2.7

Редиректоры и серверы функционируют как драйверы файловой системы и выполняются на уровне интерфейса поставщика или ниже, где находятся NetBIOS и Windows-сокеты. Драйверы транспортного протокола общаются с редиректорами и серверами через уровень, называемый интерфейсом транспортного драйвера (TDI – Transport Driver Interface). Windows NT включает следующие транспортные средства:

1. Протокол управления передачей/межсетевой протокол TCP/IP, который обеспечивает возможность работы с широким диапазоном существующих сетей.
2. NBF, потомок расширенного интерфейса пользователя NetBIOS (NetBEUI), который обеспечивает совместимость с существующими локальными вычислительными сетями на базе LAN Manager, LAN Server и MS-Net.
3. Управление передачей данных (DLC – Data Link Control), которое обеспечивает интерфейс для доступа к мейнфреймам и подключенным к сети принтерам.
4. NWLink, реализация IPX/SPX, обеспечивающая связь с Novell NetWare.

В нижней части сетевой архитектуры находится драйвер платы сетевого адаптера. Windows NT в настоящее время поддерживает драйверы устройств, выполненные в соответствии со спецификацией NDIS (Network Device Interface Specification) версии 3.0. NDIS предоставляет гибкую среду обмена данными между транспортными протоколами и сетевыми адаптерами. NDIS 3.0 позволяет отдельному компьютеру иметь несколько установленных в нем плат сетевого адаптера. В свою очередь, каждая плата сетевого адаптера может поддерживать несколько транспортных протоколов для доступа к различным типам сетевых станций.

#### **2.2.5. Подсистемы среды**

Система Windows NT была разработана таким образом, чтобы обеспечить бесперебойное выполнение множества различных типов приложений. Под Windows NT работают приложения, написанные для существующих операционных систем типа MS-DOS, OS/2 и Windows 3.x. Она также выполняет приложения, подготовленные для более нового API типа POSIX и Win32. Windows NT поддерживают различные приложения с помощью подсистем среды (environment subsystems), которые являются процессами Windows NT и эмулируют среду различных операционных систем. В настоящей главе рассмотрено, каким образом операционная система Windows NT обеспечивает общие услуги, которые могут быть вызваны подсистемами среды для выполнения базовых функций операционной системы. Подсистемы используют средства, предоставляемые исполняющей системой, для построения требуемой приложению специфической среды. На рис. 2.8 показано упрощенное представление подсистем среды в Windows NT.

Как показано на рис. 2.8, каждая подсистема выполняется в качестве отдельного процесса пользовательского режима. Выход из строя одной подсистемы не сказывается на работе других или операционной системы в целом. Таким образом, каждая подсистема защищена от ошибок в других подсистемах (исключение составляет

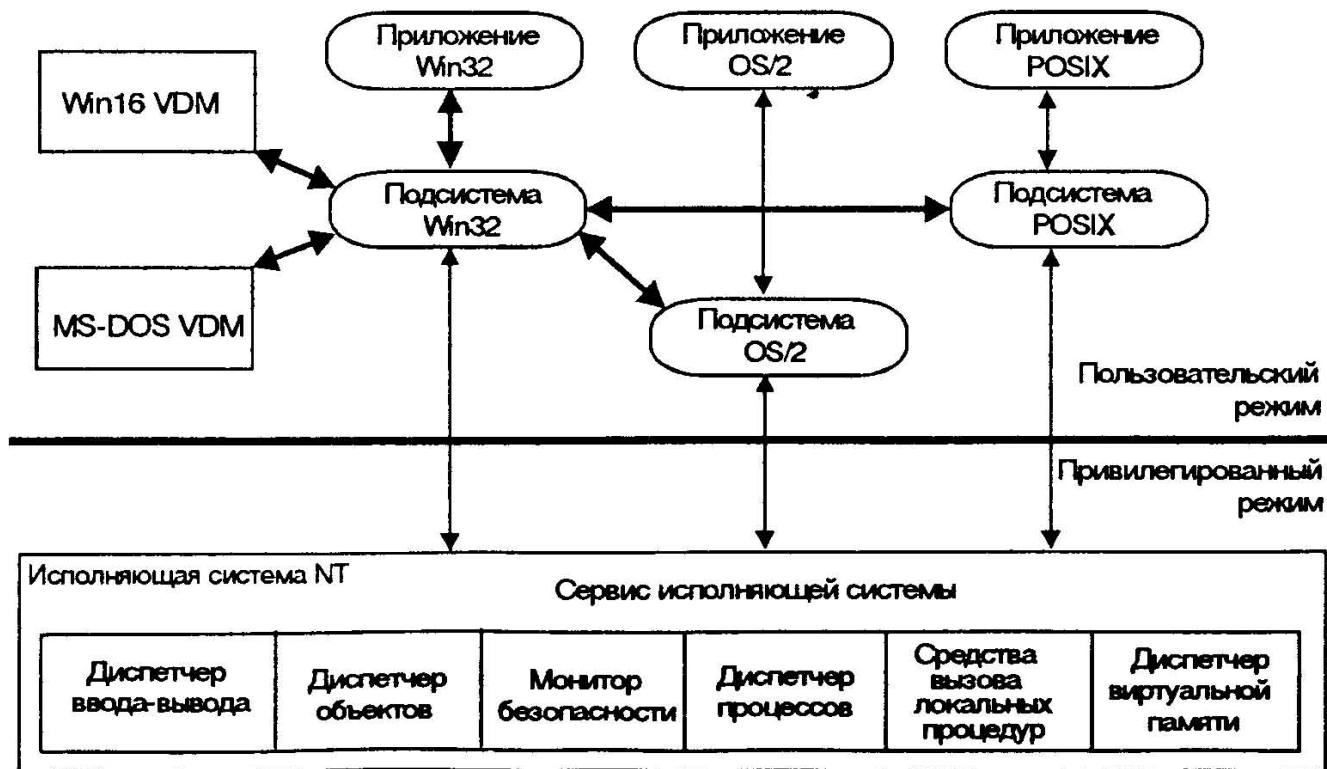


Рис. 2.8

лишь подсистема Win32, сбой в которой блокирует клавиатуру, мышь и работу с экраном для всех подсистем). Приложения являются процессами пользовательского режима, так что они не могут мешать подсистемам или исполняющей системе.

Windows NT обеспечивает следующие защищенные подсистемы и многочисленные виртуальные машины DOS (VDM – Virtual DOS Machine): MS-DOS VDM; Win 16 VDM; подсистему OS/2; подсистему POSIX; подсистему Win32. За исключением подсистемы Win32, каждая подсистема среды является дополнительной и загружается только в тот момент, когда ее услуги необходимы клиентскому приложению. Приложения, написанные для MS-DOS, при работе под Windows NT функционируют в контексте процесса, который именуется виртуальной машиной DOS. VDM является приложением Win32, которое обеспечивает полную виртуальную машину x86 (80386 или выше), работающую под управлением MS-DOS. Число одновременно задействованных VDM не ограничивается. Каждая виртуальная машина DOS функционирует в собственном адресном пространстве, что обеспечивает защищенность работающих приложений друг от друга, а также операционной системы от VDM (рис. 2.9).

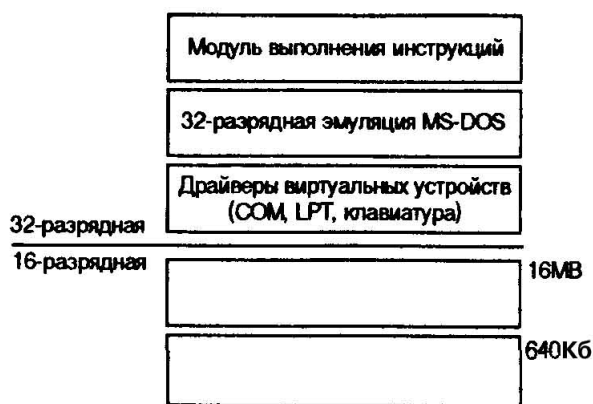


Рис. 2.9

При работе Windows NT на компьютерах с процессорами x86 доступен режим процессора, называемый виртуальным-x86. Этот режим позволяет напрямую выполнять большинство инструкций приложений DOS. Некоторые инструкции, такие как команды ввода-вывода, должны эмулироваться (для виртуализации аппаратных средств). В случае если работа Windows NT осуществляется на RISC-процессоре, аппаратная поддержка выполнения инструкций x86 недоступна. В этом случае дополнительно к аппаратной эмуляции требуется и эмуляция всех команд x86. Для нормальной работы приложений DOS VDM создает

виртуальный компьютер со следующими возможностями: поддержкой выполнения x86-инструкций, которая обеспечивается блоком выполнения инструкций (Instruction Execution Unit); поддержкой системы прерываний базовой системы ввода-вывода (BIOS), обеспечиваемой модулем эмуляции MS-DOS (MS-DOS Emulation Module); поддержкой системных функций MS-DOS (Int 21), которые доступны с помощью модуля эмуляции MS-DOS; виртуальными аппаратными средствами, такими как монитор и клавиатура, которые обеспечиваются драйверами виртуальных устройств VDD (Virtual Device Drivers). При использовании компьютеров на базе x86 выполнение текстовых приложений DOS возможно в оконном или полноэкранном режиме. Графические приложения функционируют только в полноэкранном режиме. В случае если работающее в окне приложение DOS в какой-либо момент осуществляет переключение в видеорежима, это приложение автоматически переходит на полноэкранное выполнение. На RISC-компьютерах все приложения DOS могут работать только в оконном режиме.

Windows NT использует одну многопоточную VDM, поддерживающую функционирование 16-разрядных приложений Windows (Win16). Основной целью поддержки Win16 является обеспечение полноценной работы приложений Win16 в среде Windows NT. Win16 VDM (иногда называется WOW – Windows on Windows) обеспечивает приоритетную многозадачность применительно к другим процессам, выполняющимся в системе. Однако любое приложение Win16 использует неприоритетную многозадачность по отношению к аналогичным приложениям. Таким образом, в каждый момент времени может выполняться только одно приложение Win16, в то время как другие приложения блокированы. После передачи управления Win16 VDM, при возврате из системы, всегда разблокируется приложение Win16, которое выполнялось перед переключением Win16 VDM (рис. 2.10).

Подсистема OS/2 поддерживает символично-ориентированные приложения OS/2 на x86-компьютерах. Эта подсистема не поддерживается на RISC-компьютерах; однако приложения реального режима OS/2 могут выполняться на RISC-компьютерах в среде MS-DOS. Связанные приложения, разработанные для OS/2 или MS-DOS, будут всегда выполняться в подсистеме OS/2, если она доступна.

Подсистема POSIX в Windows NT разработана для выполнения приложений POSIX и отвечает требованиям POSIX.1. POSIX (Portable Operating System Interface for Computing Environments – переносимый интерфейс операционной системы для вычислительных сред) представляет собой предварительный набор стандартов, подготовленных Институтом инженеров по электронике и радиотехнике (IEEE) для определения различных аспектов операционной системы, включая темы типа API, безопасности, работы с сетями и графического интерфейса. К настоящему моменту только один из этих стандартов, POSIX.1 (также называемый IEEE Standard 1003.1-1990), перешел от состояния проекта к оконча-

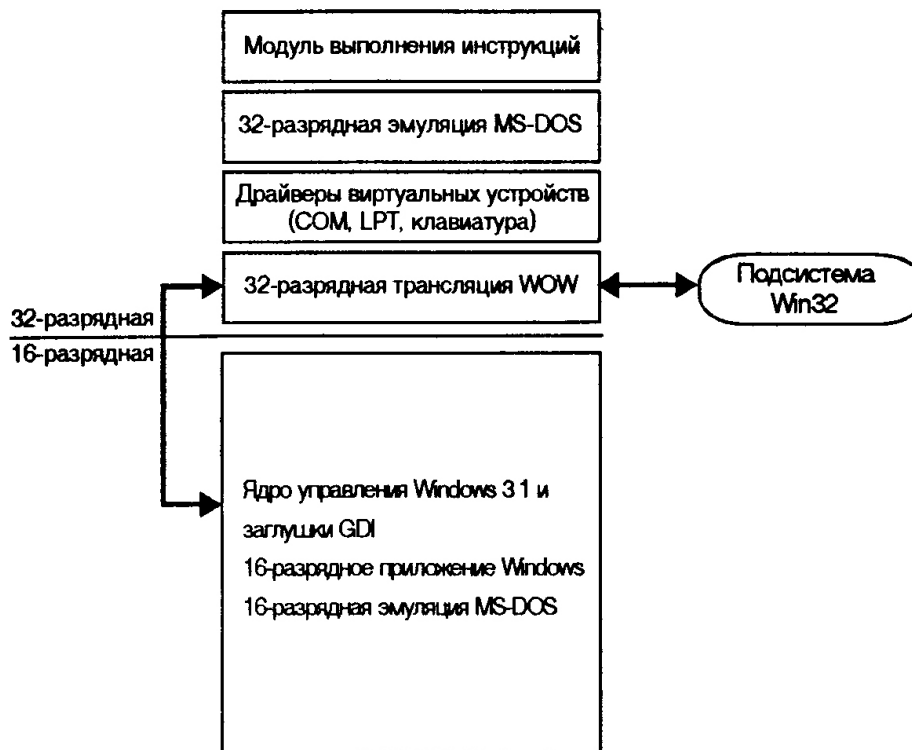


Рис. 2.10

тельной форме и получил признание пользователей. Приложения POSIX требуют от файловой системы определенных функциональных возможностей типа поддержки регистрозависимых имен файла и файлов с несколькими именами (или жестких связей). Новая файловая система NTFS удовлетворяет этим требованиям POSIX. Любое приложение POSIX, обращающееся к ресурсам файловой системы, должно иметь доступ к разделу NTFS. Приложения POSIX, которые не обращаются к ресурсам файловой системы, могут работать с любой из доступных файловых систем.

Win32 является основной подсистемой среды. Кроме способности выполнять приложения Win32, эта подсистема управляет клавиатурой, мышью и экранным выводом для всех подсистем. Подсистема Win32 отвечает за получение всей вводимой пользователем информации (или сообщений) и доставку этой информации соответствующим приложениям. Модель ввода Win32 оптимизирована для использования преимуществ приоритетной многозадачности Windows NT. На рис. 2.11

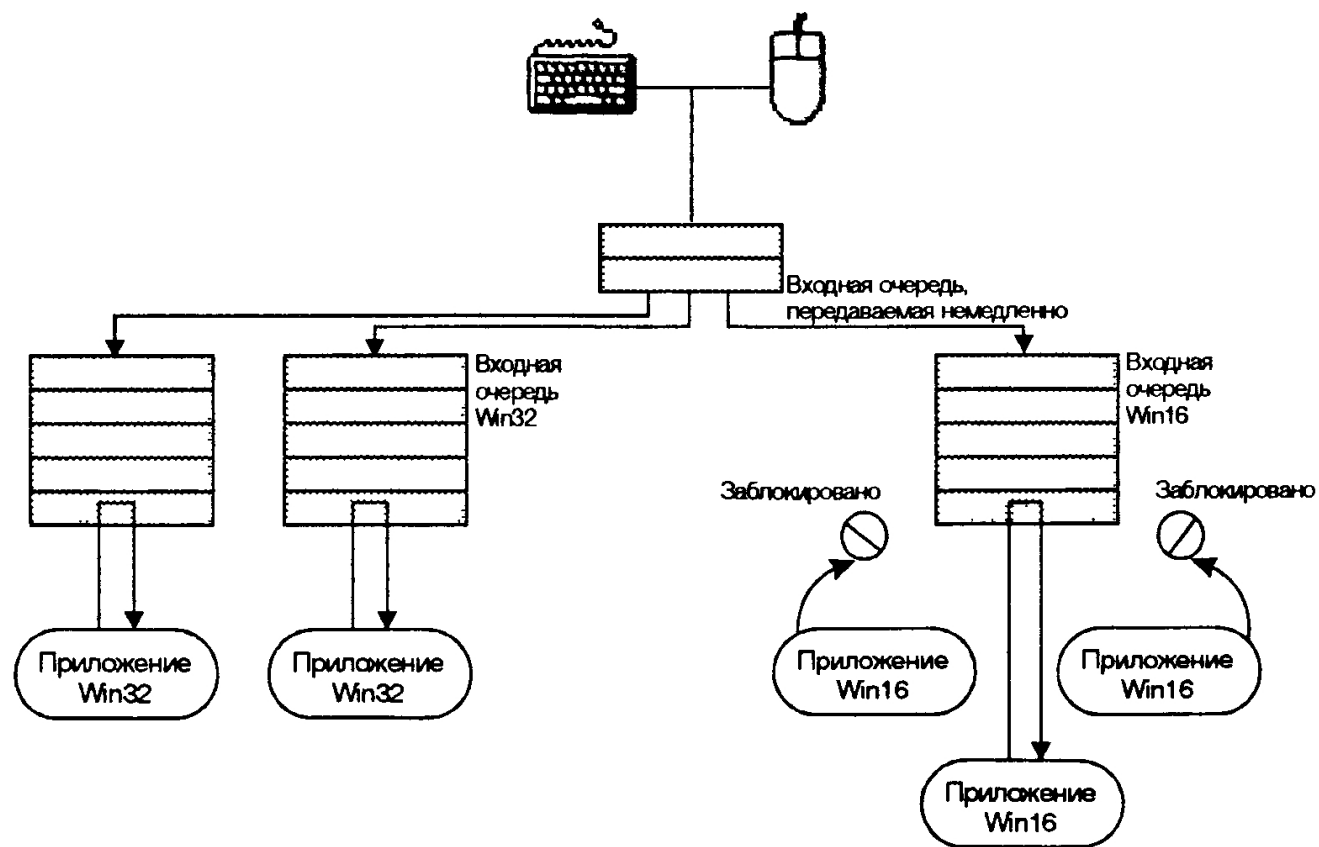


Рис. 2.11

показано, как подсистема Win32 обрабатывает ввод для приложений Win32 и Win16.

Win32 использует модель десинхронизированного (desynchronized) ввода для приложений Win32 и синхронизированного (synchronized) ввода для Win16. Например, когда подсистема Win32 получает сообщение для приложения Win32, она помещает сообщение в отдельную необработанную входную очередь. При первой возможности подсистема Win32 передает это сообщение нити управления входной очереди соответствующего приложения Win32. Если нить управления останавливает получение адресуемых ей сообщений, никакие другие приложения Win32 не могут воздействовать на них.

В противоположной ситуации все входные сообщения 16-разрядных приложений Windows заносятся в общую очередь. В любой момент времени все приложения, за исключением просматривающего входную очередь сообщений, блокированы. Однако, как и при использовании Windows 3.1, если выполняемое приложение имеет проблемы с получением сообщений из очереди или делает это очень медленно, остальные приложения остаются блокированными.

## **2.3. Файловая система**

Windows NT поддерживает различные файловые системы, включая существующие файловые системы FAT (File Allocation Table) и HPFS (High Performance File System). Она также включает новую файловую систему NTFS (NT File System), разработанную для использования преимуществ очень больших дисков и быстрых процессоров на существующих и перспективных компьютерах. Windows NT также поддерживает редиректоры и серверы в качестве файловых систем. Кроме того, Windows NT может использовать файловую систему CD для работы с дисководом CD-ROM. Также поддерживаются Named Pipes File System (NPFS) и Mailslot File System (MSFS), которые используются для связи между процессами.

### **2.3.1. Файловая система FAT**

Как было отмечено ранее, файловая система FAT названа в соответствии с наименованием метода организации данных – таблицы распределения файлов. Эта таблица обеспечивает связи одного распределяемого блока (одних или нескольких секторов) с другим. Через несколько лет после создания была произведена доработка для обеспечения функционирования с действительно большими дисками и мощными персональными компьютерами. Для MS-DOS версии 4.0 элементы FAT были увеличены с 12 до 16 бит, позволив, таким образом, работать с разделами объемом более 32 Мб. Корневой каталог имеет фиксированный размер и расположение на диске. Каталоги – специальные

файлы с 32-байтовыми элементами для каждого файла, содержащегося в этом каталоге. Элемент для каждого файла включает следующую информацию: имя файла (восемь плюс три символа); байт атрибута (8 бит); время модификации (16 бит); дату модификации (16 бит); первый размещаемый блок (16 бит); размер файла (32 бита).

Эта информация используется всеми операционными системами, которые поддерживают файловую систему FAT. Кроме того, Windows NT может сохранять дополнительные отметки времени на элементе каталога FAT. Эти элементы позволяют определить момент последнего доступа к файлу; применяются преимущественно приложениями POSIX.

Биты байта атрибута файла в элементе каталога указывают, имеет ли файл соответствующие атрибуты. Установленный первый бит идентифицирует, что файл является подкаталогом; второй отмечает файл в качестве метки тома. Обычно только операционная система может управлять назначениями этих битов. Кроме этого, файлы FAT имеют четыре специальных атрибута, которые могут применяться пользователем: архивный, системный, скрытый и только для чтения.

Windows NT версии 3.5 и выше использует эти биты атрибута для поддержки длинных имен файлов (до 255 символов) в разделах FAT; используемый для этого способ не мешает MS-DOS или OS/2 обращаться к подобному разделу. Всякий раз при создании пользователем файла с длинным именем (т.е. файла, имя которого превышает стандартное ограничение «восемь плюс три» файловых систем MS-DOS и OS/2 или содержит расширенные и смешанные символы) Windows NT создает стандартный элемент каталога для файла, обеспечивая имя «восемь плюс три» так же, как и на томе NTFS. Кроме этого стандартного элемента, Windows NT создает один или несколько вторичных элементов каталога для файла; каждый из вторичных элементов рассчитан на 13 символов в длинном имени файла. Эти вторичные элементы каталога сохраняют соответствующую часть длинного имени файла в Unicode. Windows NT устанавливает атрибуты (том, системный, скрытый, только для чтения) для вторичного элемента каталога, чтобы отметить его в качестве части длинного имени файла. MS-DOS и OS/2 игнорирует элементы каталога с таким набором атрибутов, так что эти элементы являются невидимыми для подобных операционных систем. Вместо этого MS-DOS и OS/2 обращаются к файлу, используя имя «восемь плюс три», которое содержится в стандартном элементе каталога для файла.

По умолчанию Windows NT поддерживает длинные имена файла для разделов FAT. Файловая система Windows NT FAT функционирует аналогично MS-DOS и Windows. Фактически, можно устанавливать Windows NT на существующем разделе FAT. Допускается безболезненный перенос или копирование файлов с тома FAT на NTFS. При выполнении обратной операции (от NTFS к FAT) будет потеряна информация о разрешениях и альтернативных потоках.

### 2.3.2. Файловая система HPFS

HPFS имеет особенности, которые способствуют ее эффективному управлению большими объемами жесткого диска. HPFS также поддерживает длинные имена файла (до 255 символов). Когда HPFS форматирует **том** (раздел или несколько разделов, отформатированных для использования файловой системой), первые 18 секторов резервируются для блока начальной загрузки (boot block), суперблока (super block) и запасного блока (spare block). Эти три структуры используются для загрузки операционной системы, поддержки файловой системы и восстановления при возможных ошибках.

HPFS также резервирует пространство под два битовых массива (bitmap) объемом 2 Кб для каждого дискового интервала в 16 Мб. Каждый битовый массив отводит по одному биту для каждого размещаемого блока (равного одному сектору) в полосе 8 Мб, показывая, какие размещаемые блоки находятся в использовании. Битовые массивы поочередно размещаются в конце и начале каждой полосы, обеспечивая, таким образом, максимальное количество непрерывного пространства для данных (16 Мб вместо 8 Мб). Кроме того, HPFS планирует запись новых файлов, оставляя участок памяти между новым и существующим файлом с тем, чтобы каждый файл имел участок памяти для расширения в непрерывном дисковом пространстве. Эта особенность помогает HPFS осуществлять быстрый поиск данных и минимизировать фрагментацию файлов.

Другая особенность, которая объясняет быстрый поиск в каталоге – использование HPFS-технологии B-tree (B-дерево). B-tree – древовидная структура с корнем и несколькими узлами. Она содержит организованные некоторым логическим способом данные; этот способ позволяет производить быстрый просмотр всей структуры. Корень содержит административную информацию, карту для остальной структуры и, возможно, некоторые данные. Для больших каталогов технология B-tree работает значительно эффективнее линейных списков, используемых файловой системой FAT.

HPFS использует B-tree для структуризации каждого каталога и каждого файла. Каждый каталог указывает на структуры Fnode для файлов, содержащихся в этом каталоге. Структура Fnode имеет размер 512 байт и содержит заголовок, имя файла (усеченное до 15 символов), длину файла, расширенные атрибуты (EA), список управления доступом (ACL) и расположение данных файла.

Благодаря расположению битовых массивов, размер одного фрагмента может быть почти 16 Мб. Обычно Fnode может включать до 8 указателей на фрагменты. Если файл настолько большой, что восьми фрагментов не хватает для размещения всей информации, Fnode может включать до 12 указателей к узлам распределения, которые имеют пространство для большего количества фрагментов файла. Если

расширенный атрибут и ACL не могут быть включены в Fnode, Fnode содержит указатель на эту информацию. Короче говоря, HPFS имеет мощные возможности и эффективно работает на дисках объемом до 2 Гб. Однако файловая система HPFS имеет и некоторые слабые стороны. Например, при повреждении первой части тома, которая содержит информацию начальной загрузки и указатель на корневой каталог, использование тома будет невозможно. Использование системой HPFS утилиты chkdsk при каждой начальной загрузке системы и восстановление диска после ошибок требуют больших временных затрат. Кроме того, HPFS предполагает использование 512-байтовых секторов, которые не очень подходят для больших томов.

Некоторые особенности HPFS используются в Windows NT иначе, чем в OS/2. Например, Windows NT не поддерживает информацию списка управления доступом HPFS или горячее фиксирование (эти возможности, однако, доступны с NTFS). Кроме того, управление кэшированием диска и ленивой записью производится диспетчером кэша Windows NT, а не файловой системой. При перемещении или копировании файла из NTFS в HPFS теряются разрешения и альтернативные потоки; имена файла преобразуются из Unicode в наборы символов OEM. Кроме того, имя файла записывается без учета регистра. Windows NT поддерживает HPFS прежде всего для обеспечения совместимости снизу вверх при выборочной загрузке OS/2 или Windows NT. NTFS обеспечивает все возможности HPFS, добавляя к ним дополнительные, такие как безопасность и надежность. Если том не предназначен для работы с OS/2, форматирование его под NTFS более предпочтительно, чем под HPFS.

### **2.3.3. Файловая система NTFS**

NTFS обеспечивает комбинацию эффективности, надежности и совместимости, отсутствующую в FAT или HPFS. Она разработана для быстрого выполнения стандартных файловых операций типа чтения, записи и поиска, а также улучшенных операций типа восстановления файловой системы на очень больших жестких дисках. NTFS также включает возможности безопасности, требуемые для файловых серверов и высококачественных персональных компьютеров в корпоративной среде. NTFS поддерживает управление доступом к данным и привилегии владельца, что является важным для целостности корпоративных данных. В то время как каталогам, разделяемым при помощи Windows NT Server, назначаются специфические разрешения, файлам и каталогам NTFS могут назначаться разрешения вне зависимости, разделены они или нет. NTFS – единственная файловая система в Windows NT, которая позволяет назначить разрешения для отдельных файлов.

NTFS является простой, но очень мощной разработкой. Для этой перспективной файловой системы вся информация на томе NTFS

является файлом или частью файла. Каждый распределенный на томе NTFS сектор принадлежит некоторому файлу. Даже метаданные (metadata) файловой системы (информация, которая описывает непосредственно файловую систему) являются частью файла.

Эта основанная на атрибутах файловая система поддерживает объектно-ориентированные приложения, обрабатывая все файлы как объекты, которые имеют определяемые пользователем и системой атрибуты.

Каждый файл на томе NTFS представлен записью в специальном файле, называемом главной файловой таблицей (MFT – master file table). NTFS резервирует первые 16 записей таблицы для специальной информации. Первая запись этой таблицы описывает непосредственно главную файловую таблицу; за ней следует зеркальная запись (mirror record) MFT. Если первая запись MFT разрушена, то NTFS читает вторую запись для отыскания зеркального файла MFT, первая запись которого идентична первой записи MFT. Местоположения сегментов данных MFT и зеркального файла MFT записаны в секторе начальной загрузки. Дубликат сектора начальной загрузки находится в логическом центре диска.

Третья запись MFT – файл регистрации (log file); используется для восстановления файлов. Файл регистрации подробно описан в настоящей главе ниже. Семнадцатая и последующие записи главной файловой таблицы используются собственно файлами и каталогами (также рассматриваются как файлы NTFS) на томе. На рис. 2.12 показана упрощенная структура MFT.

Главная файловая таблица отводит определенное количество пространства для каждой записи файла. Атрибуты файла записываются в распределенное пространство MFT. Небольшие файлы и каталоги (обычно до 1500 байт или меньше) могут полностью содержаться внутри записи главной файловой таблицы. Подобный подход обеспечивает очень быстрый доступ к файлам. Рассмотрим, например, файловую систему FAT, которая использует таблицу размещения файлов, в которой перечисляются имена и адрес каждого файла. Элементы каталога FAT содержат индекс в таблице размещения файла. В случае если необходимо просмотреть содержимое файла, FAT сначала читает таблицу размещения файлов и убеждается в существовании файла. Далее FAT восстанавливает файл, ища цепочку распределенных блоков, относящихся к этому файлу. В NTFS поиск файла производится только для непосредственного его использования. Записи каталога помещены внутри главной файловой таблицы так же, как записи файла. Вместо данных каталоги содержат индексную информацию. Небольшие записи каталогов находятся полностью внутри структуры MFT. Большие каталоги организованы в B-tree, имея записи с указателями на внешние кластеры, содержащие элементы каталога, которые не могли быть записаны внутри структуры MFT.

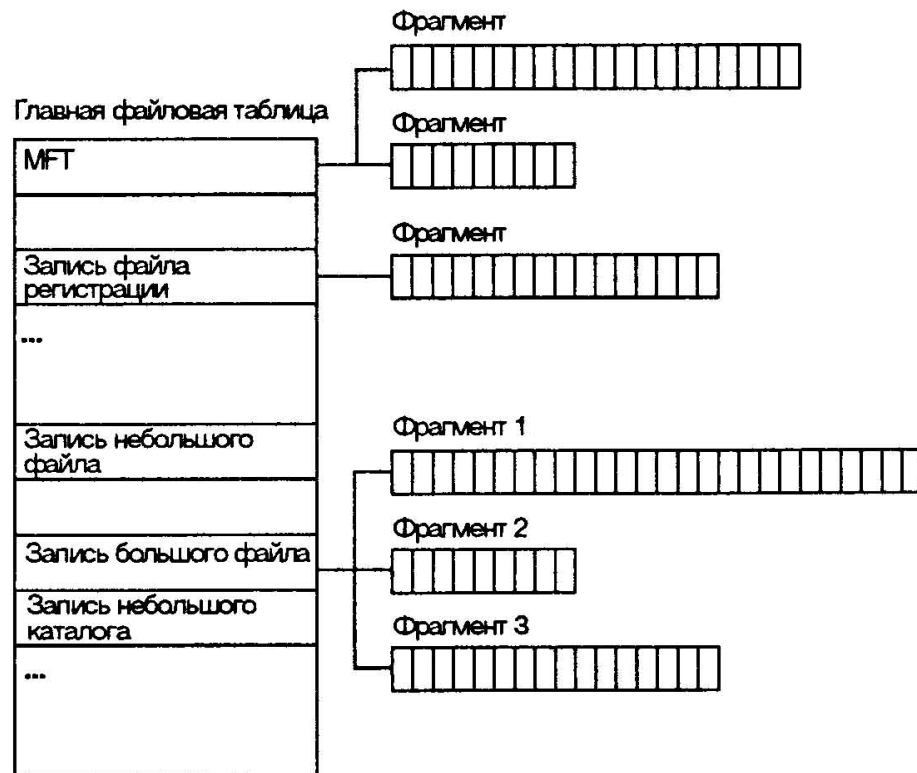


Рис. 2.12

NTFS просматривает каждый файл (или каталог) как набор атрибутов файла. Такие элементы, как имя файла, информация защиты и даже данные – все это атрибуты файла. Каждый атрибут идентифицирован кодом типа атрибута и, необязательно, именем атрибута. Если атрибуты файла могут находиться внутри записи файла MFT, они называются резидентными (resident) атрибутами. Например, информация типа имени файла и отметки времени всегда включается в запись файла MFT. Если файл слишком большой, чтобы содержать все атрибуты в записи файла MFT, часть атрибутов является нерезидентной (nonresident). Нерезидентные атрибуты занимают один или несколько пробегов (run) дискового пространства в другом месте тома (прогон дискового пространства – непрерывная линейная область на диске).

Вообще, все атрибуты могут быть вызваны как поток байтов независимо от того, являются ли они резидентными или нерезидентными.

Подобно HPFS, NTFS поддерживает имена файла до 255 символов. Имена файла NTFS используют набор символов Unicode с 16 битами; однако вопрос доступа из MS-DOS решен. NTFS автоматически генерирует поддерживаемое MS-DOS имя (восемь плюс три символа) для каждого файла. Таким образом, файлы NTFS могут использоваться через сеть операционными системами MS-DOS и OS/2. Это особенно важно для файловых серверов организации, которая использует персональные компьютеры с двумя или всеми тремя этими операционными системами.

Создавая имена файла «восемь плюс три», NTFS также позволяет приложениям MS-DOS и Windows 3.x работать с файлами, имеющими длинные имена NTFS. Кроме того, при сохранении файла приложениями MS-DOS или Windows 3.x на томе NTFS сохраняются и имя файла «восемь плюс три» и длинное имя NTFS. Поскольку NTFS использует набор символов Unicode для имен файлов, существует возможность задействования нескольких «запрещенных» символов, которые MS-DOS не может читать в имени файла. Для генерации короткого имени файла в стиле MS-DOS NTFS удаляет все эти символы и любые пробелы из длинного имени файла. Так как имя файла в MS-DOS может иметь только одну точку, NTFS также удаляет все дополнительные точки из имени файла. В случае необходимости NITS усекает имя файла до шести символов и добавляет тильду (~) и номер. Например, к каждому недублированному имени файла добавляется ~1. Повторяющиеся имена файлов заканчиваются символами ~2, ~3 и т. д. Расширение имени файла усекается до трех или меньшего количества символов. Наконец, при отображении имени файла в командной строке NTFS транслирует все символы в имени файла и расширении к верхнему регистру.

Windows NT не генерирует коротких имен для файлов, созданных приложениями POSIX в разделе NTFS. Это означает, что приложения MS-DOS и Windows 3.x не смогут работать с подобными именами, если эти имена не удовлетворяют условию «восемь плюс три». В случае необходимости работы из приложений MS-DOS или Windows с файлами, которые созданы приложениями POSIX, следует убедиться, что использованы стандартные имена MS-DOS.

NTFS поддерживает многочисленные потоки данных. Имя потока идентифицирует новый атрибут данных в файле. Потоки имеют отдельные блокировки opportunistic, блокировки файла, размеры размещения и размеры файла, но совместно используются как файл.

#### **2.3.4. Целостность и восстановимость данных в файловых системах**

До настоящего момента существовали два типа файловых систем – файловые системы с точной записью (careful write) и файловые системы с ленивой записью (lazy write). NTFS является третьим типом –

восстанавливаемой файловой системой.

Файловые системы с точной записью основываются на идее важности сохранения согласованной структуры тома. Примером файловой системы с точной записью является FAT в MS-DOS. Файловая система с точной записью работает следующим образом. Когда она осуществляет изменение структуры тома, то дается команда для записи на диск. Большинство модификаций тома производятся за один раз. Записи на диск для каждой модификации производятся таким образом, что сбой системы между двумя операциями дисковой записи оставляет том в распознаваемом состоянии с возможностью «ожидаемой» несогласованности. Диск остается пригодным для использования. Выполнение утилит типа `chkdsk` редко требуется для файловой системы с точной записью (в FAT, например, `chkdsk` необходима только для исправления последствий сбоя системы и обеспечивает быстрый способ восстановления согласованности файловой системы). Недостатком файловых систем с точной записью является медленное выполнение преобразованных в последовательность операций записи. Это происходит вследствие того, что первая запись на диск должна быть произведена и завершена прежде, чем сможет начаться вторая запись и т.д. Для мощных компьютеров подобный подход дает далеко не самое эффективное использование его возможностей.

Второй вид файловой системы, типа FAT в Windows NT и большинства файловых систем UNIX, называется системой с ленивой записью. Этот тип был разработан для ускорения дискового доступа. Так как вероятность возникновения дисковых сбоев достаточно низкая, файловая система с ленивой записью разрабатывалась с учетом использования интеллектуальной стратегии управления кэшем и обеспечения способа восстановления данных (типа утилиты `chkdsk`) в случае сбоя диска. Работа с данными производится через буфер системы ввода-вывода. В то время как пользователь просматривает каталоги или читает файлы, необходимые для записи на диск данные накапливаются в кэше. Таким образом, пользователь не должен ожидать окончания процесса выполнения записи. Кроме того, пользователь имеет возможность обращаться ко всем ресурсам файловой системы, которые, в противном случае, могли бы быть заняты выполнением операций записи. Запись данных на диск производится только в момент низкой загрузки ресурсов компьютера, а не в последовательном режиме.

Если одни и те же данные изменяются несколько раз, все модификации фиксируются в буфере системы ввода-вывода. Результатом является то, что при изменении данных система должна производить запись на диск только один раз. То есть файловая система открывает файл один раз и далее выполняет сразу все модификации, после чего файл закрывается. Недостатком файловой системы с ленивой записью является то, что в случае сбоя диска восстановление данных будет занимать значительно больше времени, чем при

использовании файловой системы с точной записью. Это происходит вследствие того, что утилита типа chkdsk должна при восстановлении просканировать весь том для проверки его фактического состояния.

NTFS является восстанавливаемой (recoverable) файловой системой. Она сочетает быстрое действие файловой системы с ленивой записью и практически мгновенное восстановление. NTFS гарантирует согласованность данных тома, используя стандартную регистрацию транзакций и методы восстановления. Она включает метод ленивой записи и систему восстановления тома, которая обычно занимает одну или две секунды после перезагрузки компьютера. Регистрация транзакции, позволяющая NTFS производить быстрое восстановление, требует значительно меньших затрат по сравнению с файловыми системами точной записи. При использовании раздела на одиночном устройстве, NTFS позволяет производить восстановление системы после сбоя, однако в результате ошибки ввода-вывода часть данных может быть потеряна. В сочетании с поддержкой зеркального отражения (mirroring) или контролем четности с чередованием (parity stripping), что выполняется отказоустойчивым драйвером, NTFS может выдержать любой одиночный сбой. Раздел NTFS остается доступным, хотя, возможно, не может использоваться для загрузки. То есть, даже если сектор начальной загрузки потерян и невозможна передача управления копии сектора начальной загрузки NTFS, компьютер можно загрузить из другого раздела или другого дисковода (при этом сбойный раздел NTFS будет оставаться доступным).

NTFS также поддерживает горячее фиксирование (hot-fixing), которое позволяет файловой системе при возникновении ошибки из-за плохого сектора переместить информацию в другой сектор и отметить первоначальный в качестве плохого. Этот подход прозрачен для любых приложений, выполняющих дисковые операции ввода-вывода. Горячее фиксирование позволяет устранить сообщения об ошибках типа «Abort, Retry, or Fail», которые происходят, когда файловая система типа FAT сталкивается с плохим сектором.

При использовании NTFS на отказоустойчивом устройстве и обнаружении ошибки в одной копии кластера данные могут быть восстановлены. Плохой кластер отмечается в файле плохих кластеров (Bad Cluster File) и заменяется другим кластером. Далее копия первоначальных данных записывается в новый кластер.

Каждая операция ввода-вывода, которая изменяет файл на томе NTFS, рассматривается файловой системой как транзакция и может выполняться как неделимый блок. При модификации файла пользователем сервис файла регистрации (Log File Service) фиксирует всю информацию по повторению (redo) или откату (undo) транзакции. Применительно к восстановлению, redo – информация, которая сообщает NTFS о путях повторения транзакции, undo – об отмене транзакции, которая не была завершена или имела ошибку.

Если транзакция завершена успешно, производится модификация файла. Если транзакция не завершена, NTFS заканчивает или производит откат транзакции, следуя инструкциям в информации отмены. Если NTFS обнаруживает ошибку в транзакции, транзакция также прокручивается обратно.

Восстановление файловой системы осуществляется NTFS очень просто. При сбое системы NTFS выполняет три прохода: проход анализа (analysis pass), повторный проход (redo pass) и проход отмены (undo pass). В течение прохода анализа, на основании информации файла регистрации, NTFS оценивает повреждение и точно определяет, какие кластеры должны быть модифицированы. Во время повторного прохода выполняются все этапы транзакции от последней контрольной точки (checkpoint). Проход отмены осуществляет возврат любых незавершенных транзакций.

Ленивая передача (lazy commit) – важная особенность NTFS. Она позволяет NTFS минимизировать затраты регистрации для поддержания высокой эффективности. Ленивая передача подобна ленивой записи. Вместо использования ресурсов для немедленной отметки транзакции как успешно завершившейся, эта информация заносится в кэш и записывается в файл регистрации как фоновый процесс. Если происходит сбой до того, как информация о транзакции была зарегистрирована, NTFS произведет повторную проверку транзакции для определения ее успешности. Если NTFS не может гарантировать, что транзакция была завершена успешно, производится откат транзакции. Никакие незавершенные модификации тома не позволяют.

Каждые несколько секунд NTFS проверяет кэш, чтобы определить состояние ленивой записи, и отмечает это состояние в качестве контрольной точки в файле регистрации (checkpoint). Если вслед за определением контрольной точки последует сбой, система имеет возможность приведения своего состояния к состоянию, зафиксированному контрольной точкой. Этот метод использует наиболее оптимальное время восстановления, сохраняя очередь событий, которая может потребоваться в процессе восстановления.

## **2.4. Служба каталогов**

Windows NT предлагает ряд методов управления диском, которые можно использовать для организации и сохранения данных на дисках. Пользователь может выбрать, сколько физических дисков и разделов логического диска будет иметь система: для организации данных в разделах может производиться объединение в том (Volume Set) или чередование дисков (Stripe Set); для гарантирования надежности данных в системе предлагаются несколько вариантов обеспечения отказоустойчивости; для предотвращения потери данных предоставляется возможность использования других способов резервного копирования и

восстановления, типа копирования на ленту и применения источников бесперебойного питания (UPS).

Физический диск может содержать один или несколько логических разделов. Каждый раздел или набор разделов форматируется для определенной файловой системы как том; назначается символ дисководов. Первичный раздел – часть физического диска, которая может быть использована операционной системой. Каждый диск может иметь до четырех разделов, один из которых может быть расширенным. Расширенные разделы могут быть разбиты на логические дисководы; разбиение первичных разделов невозможно. Свободное пространство в расширенном разделе может также использоваться для создания наборов томов или других видов томов в целях отказоустойчивости. В случае если диск не содержит раздела начальной загрузки, он может использоваться полностью как расширенный раздел.

*Объединение в том* – просто способ объединения многочисленных областей свободного пространства и форматирование его в одиночный логический диск с одиночным символом дисководов. Можно использовать утилиту Disk Administrator для создания и расширения объединенного тома. Каждый объединенный том может включать до 32 областей свободного пространства из одного или нескольких физических дисков или разделов. Объединенный том организован так, чтобы свободное пространство на одном диске было заполнено перед началом использования пространства на следующем диске. Использование объединенного тома не увеличивает дисковую эффективность. Только объединенный том, форматируемый под NTFS, может быть расширенным. Объединенный том не может содержать в своем составе зеркальных (mirrored) или чередующихся (striped) компонент.

*Чередование дисков* – способ увеличения дисковой эффективности (рис. 2.13). Создание чередующихся дисков производится с использованием утилиты Disk Administrator. Подобный метод увеличивает эффектив-

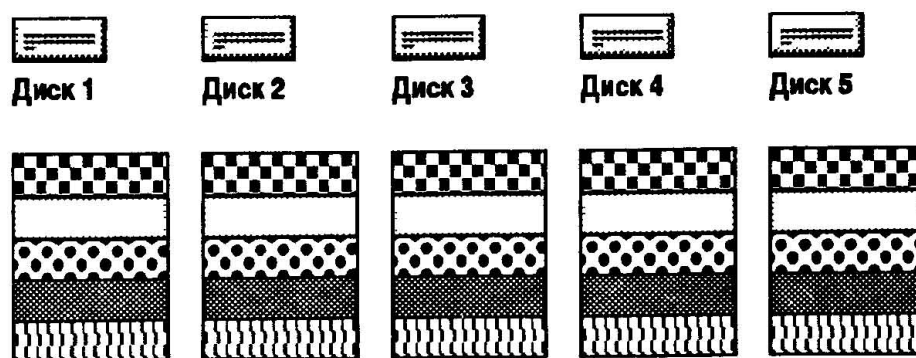


Рис. 2.13

ность и чтения и записи, так как одновременно могут выполняться несколько команд ввода-вывода. Метод чередования дисков позволяет объединять от 2 до 32 дисков. Если диски имеют различные объемы, то в качестве общего размера раздела используется самый маленький объем. Остаточное свободное пространство может быть использовано индивидуально или в составе объединенного тома.

Обратите внимание, что чередование дисков отличается от используемого Windows NT Server чередования дисков с контролем по четности. Чередование дисков в Windows NT не сопровождается чередованием контроля по четности. Из-за отсутствия чередования контроля по четности подобная организация критична к сбоям. Если информация потеряна, не существует способа ее восстановления. Некоторые дисковые драйверы обеспечивают возможности горячего фиксирования (hot-fix) для содействия гарантированию безопасности данных.

Каждый раз в процессе загрузки Windows NT выполняется процедура автоматической проверки. Если эта процедура обнаруживает «загрязнение» тома, она автоматически выполняет команду chkdsk для установления несогласованности или ошибки (обратим внимание, что для chkdsk является нормальным сообщать об ошибках противоречивости для дискового NTFS, содержащего страничный файл). Если не найдены никакие поврежденные файлы и не обнаружено других ошибок, том отмечается как чистый и потребность в выполнении chkdsk /f отсутствует. Если команда chkdsk размещает файлы или каталоги, которые потеряли указатели на родительский каталог, они получают имена FILE###.CHK или DIR###.CHK и записываются в каталог FOUND### на томе NTFS.

## **2.5. Безопасность и отказоустойчивость**

Безопасность для Windows NT является частью требований к ОС. Модель безопасности включает компоненты для управления доступом к объектам (например, к файлам или разделяемым принтерам), для определения того, кем осуществляются действия над объектом, и для назначения списка протоколируемых событий.

### **2.5.1. Модель безопасности Windows NT**

На рис. 2.14 представлена модель безопасности Windows NT, которая включает следующие компоненты:

1. *Процессы входа в систему* (Logon Processes), которые принимают запросы пользователей на вход в систему. Они включают начальный интерактивный вход в систему.
2. *Распорядитель локальной безопасности* (Local Security Authority), который гарантирует, что пользователь имеет разрешение на обращение к системе. Этот компонент является основой подсистемы

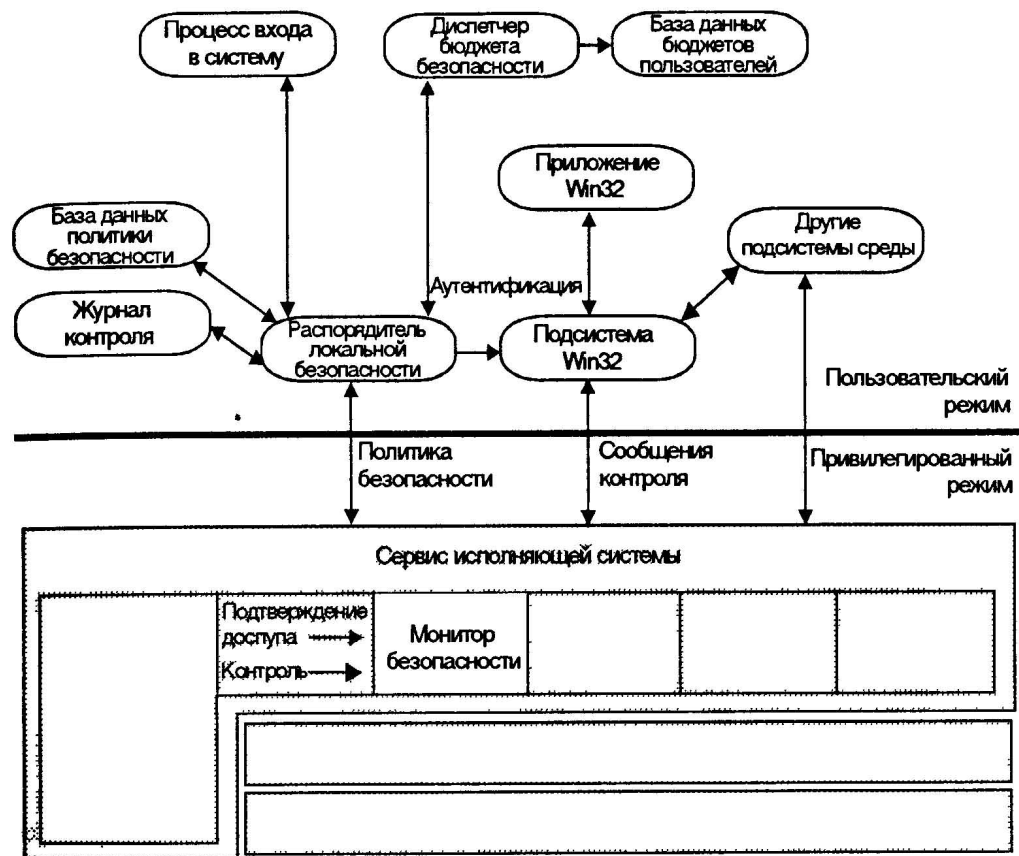


Рис. 2.14

темы безопасности Windows NT. Он генерирует маркеры доступа (описаны ниже), управляет политикой локальной безопасности и обеспечивает интерактивный сервис аутентификации пользователя. Распорядитель локальной безопасности также управляет политикой контроля и регистрирует контрольные сообщения, сгенерированные монитором безопасности.

3. *Диспетчер бюджета безопасности* (SAM – Security Account Manager), который поддерживает базу данных бюджетов пользователя. Эта база данных содержит информацию по бюджетам всех пользователей и групп. SAM обеспечивает аутентификацию пользователя, которая используется распорядителем локальной безопасности.
4. *Монитор безопасности* (Security Reference Monitor), который проверяет, имеет ли пользователь права доступа к объекту, и отслеживает любое предпринимаемое пользователем действие. Обеспечивает сервис и привилегированному и пользовательскому режимам, который гарантирует осуществление доступа к объектам только пользователям и процессам, имеющим необходимые разрешения.

Вместе эти компоненты известны как подсистема безопасности, которая является интегральной подсистемой, так как воздействует на операционную систему Windows NT в целом.

Модель безопасности Windows NT разработана в соответствии с уровнем C2, определенным Министерством обороны США. Наиболее важные требования уровня безопасности C2 перечислены ниже:

- владелец ресурса (например, файла) должен иметь возможность управлять доступом к ресурсу;
- операционная система должна защищать объекты от несанкционированного использования другими процессами. Например, система должна защищать память так, чтобы ее содержимое не могло читаться после освобождения процессом, и после удаления файла не допускать обращения к данным файла;
- перед получением доступа к системе каждый пользователь должен идентифицировать себя, вводя уникальное имя входа в систему и **пароль** (уникальная строка символов, вводимая для авторизации доступа и регистрации; важное средство защиты; пароль служит для ограничения входа в систему и доступа к компьютерным системам и ресурсам). Система должна быть способна использовать эту уникальную идентификацию для контроля действий пользователя;
- администратор системы должен иметь возможность контроля связанных с безопасностью событий (audit security-related events). Доступ к этим контрольным данным должен быть ограничен администратором.

- система должна защищать себя от внешнего вмешательства типа модификации выполняющейся системы или хранимых на диске системных файлов.

Основной целью модели безопасности Windows NT являются контроль и управление доступом к объектам. Модель безопасности хранит информацию для каждого пользователя, группы пользователей и объекта. Она может идентифицировать попытки доступа, которые сделаны непосредственно пользователем, или те, которые сделаны косвенно программой или другим процессом, выполняющимся в интересах пользователя. Windows NT также отслеживает и управляет доступом как к видимым пользователям объектам интерфейса (типа файлов и принтеров), так и к невидимым объектам (типа процессов и именованных каналов).

Кроме этого, модель безопасности определяет не только объект, к которому осуществляется обращение, но и способ доступа. Администратор может назначать разрешения (permissions) пользователям и **локальным группам** – группам, которым можно предоставлять права и привилегии исключительно для рабочих станций. **Привилегия** позволяет пользователю выполнять определённые действия в системе. Привилегии предоставляются системе в целом в отличие от прав доступа, применимых к определённым объектам. Привилегии могут включать в себя учётные записи пользователей и **глобальные группы** (группы, которые могут использоваться в собственном домене, серверах и рабочих станциях домена и доверяющих доменах. Во всех случаях глобальная группа имеет предоставленные права и привилегии и может становиться членом локальных групп) доверяемых доменов – предоставлять или запрещать доступ к отдельным объектам. Например, по отношению к определённому файлу пользователю могут быть назначены следующие разрешения: Read (чтение); Delete (удаление); Write (запись); Change Permission (разрешение модификации); Executive (выполнение); Take Ownership (монопольное использование); No access (отсутствие доступа).

Способность назначать разрешения по усмотрению владельца (или другого пользователя, имеющего на это разрешение) называется контролируемым управлением доступом (discretionary access control). Администраторы могут назначать разрешения индивидуальным пользователям или группам (в целях сопровождения желательно назначить разрешения группам). Например, администратор может управлять доступом к каталогу REPORTS, предоставляя GROUP1 доступ на чтение и GROUP2 доступ на чтение, запись и модификацию.

Возможности контроля (auditing) Windows NT позволяют записывать события с целью фиксирования попыток доступа пользователя к объектам, разновидности производимого доступа и его успешности или безуспешности. Для установки контроля на компьютере пользователь должен применять опции Auditing и Security в программах User Manager,

File Manager, Print Manager и других утилитах. Работая с этими инструментальными средствами, пользователь может определять типы контролируемых событий, которые необходимо включать в журнал безопасности. Дополнительную информацию по установке параметров контроля с помощью перечисленных программ можно найти в документации к Windows NT.

Пользователи идентифицируются системой с помощью **идентификатора безопасности (SID – Security ID)** – уникального имени, идентифицирующего зарегистрированного пользователя. Идентификатор безопасности может идентифицировать индивидуального пользователя или группу, уникален во времени и пространстве, то есть существование двух одинаковых идентификаторов безопасности невозможно. Когда пользователь осуществляет вход в систему, Windows NT создает соответствующий маркер доступа. Он включает идентификатор безопасности для пользователя, идентификаторы безопасности для групп, к которым пользователь принадлежит, а также другую информацию типа имен пользователя и групп. В дополнение к этому, каждый процесс, который выполняется от имени этого пользователя, будет иметь копию его маркера доступа.

Windows NT обращается к идентификатору безопасности внутри маркера доступа пользователя, когда пользователь делает попытку обратиться к объекту. Для определения того, имеет ли пользователь разрешение на этот доступ, идентификатор безопасности пользователя сопоставляется со списком разрешений на доступ к объекту.

Прежде чем пользователь сможет сделать что-либо в системе Windows NT, он должен произвести вход в систему с указанием имени и пароля. Рис. 2.15 иллюстрирует интерактивный процесс входа в систему для Windows NT.

Начальный процесс входа в систему для Windows NT интерактивен, т.е. пользователь должен ввести информацию с клавиатуры в ответ на диалоговое окно, отображенное операционной системой на экране. На основе представленной пользователем информации Windows NT разрешает или запрещает доступ.

Ниже представлен перечень шагов, задействованных в процессе интерактивного входа в систему:

- Пользователь нажимает комбинацию клавиш <Ctrl>+<Alt>+<Del> для уведомления Windows NT. Эта комбинация клавиш перед входом в систему защищает от программ типа троянского коня, которые пытаются выдать себя за операционную систему и нелегально зафиксировать имя пользователя и его пароль.
- После ввода пользователем имени и пароля процесс входа в систему вызывает распорядителя локальной безопасности.
- Распорядитель локальной безопасности запускает на выполнение соответствующий пакет аутентификации (authentication package).

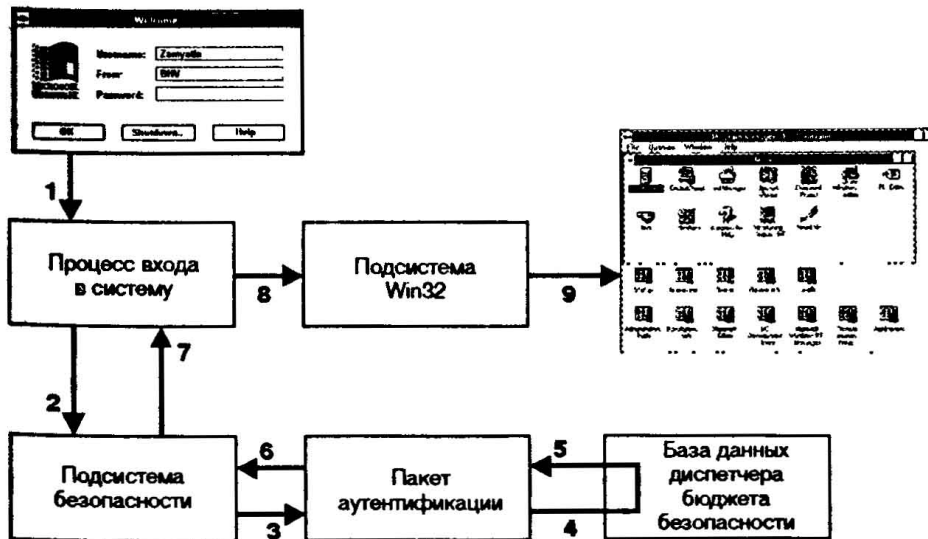


Рис. 2.15

- Аутентификационный пакет проверяет базу данных бюджетов пользователя для определения, является ли бюджет локальным. Если это так, имя и пароль сравниваются с содержащимися в базе данных бюджетов пользователя. Если нет, запрашиваемый вход в систему переадресовывается альтернативному аутентификационному пакету. Таким образом проводится **авторизация**, т.е. проверка регистрационной информации о пользователе. Регистрация выполняется на рабочей станции или на контроллере домена (при регистрации в домене).
- После того как бюджет подтвержден, SAM (который владеет базой данных бюджетов пользователя) возвращает идентификатор безопасности пользователя и идентификаторы безопасности всех глобальных групп, к которым принадлежит пользователь.
- Аутентификационный пакет создает сеанс входа в систему и далее передает этот сеанс и идентификаторы безопасности, связанные с пользователем, распорядителю локальной безопасности.
- В случае если вход в систему отклонен, сеанс входа в систему удаляется и процессу входа в систему возвращается ошибка. В противоположном случае создается маркер доступа, который содержит идентификатор безопасности пользователя и идентификаторы безопасности группы Everyone и других групп. Маркер доступа также содержит права пользователя, назначенные

совокупности идентификаторов безопасности. Этот маркер доступа возвращается процессу входа в систему со статусом успешного (Success).

- Сеанс входа в систему вызывает подсистему Win32, чтобы создать процесс и присоединить маркер доступа к процессу, создавая, таким образом, субъект для бюджета пользователя.

После корректного входа процессу оболочки пользователя передается маркер доступа. Информация этого маркера доступа отражается на всех действиях пользователя или любых выполняемых в интересах пользователя процессах. Обычно доступ к объекту определяется на основании сравнения информации о пользователе и группах, содержащейся в маркере доступа, с разрешениями, действующими для объекта. Однако некоторые операции, выполняемые пользователями, не связаны с отдельными объектами. Например, может потребоваться, чтобы определенные пользователи имели возможность регулярно создавать резервные копии информации сервера. Эти пользователи должны быть способны выполнять работу без учета разрешений, которые были установлены для архивируемых файлов. В подобных случаях администратор может назначать специальные права пользователя (user rights, иногда называемые привилегиями – privileges), которые дают пользователям или группам доступ к сервису. Пример установки прав пользователя показан на рис. 2.16.

Резервное копирование файлов и каталогов, закрытие системы, вход в систему не в интерактивном режиме, изменение системного

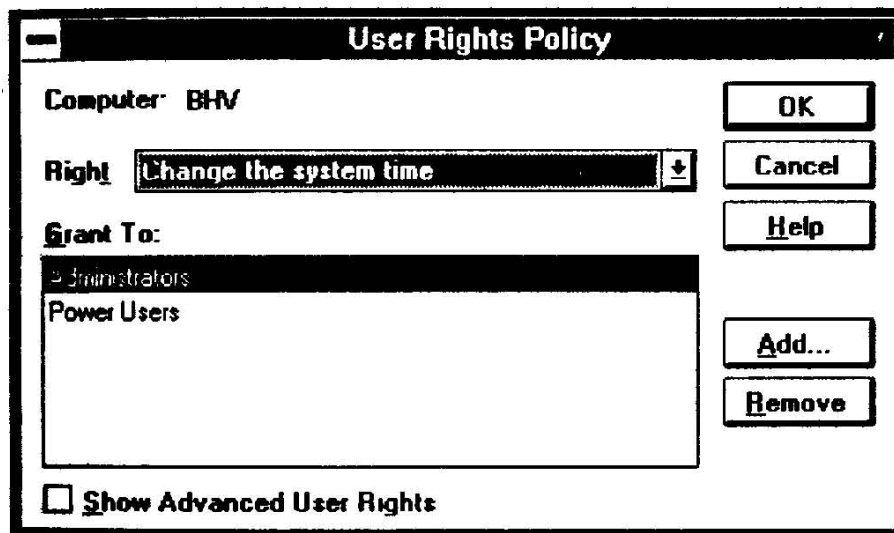


Рис. 2.16

времени – все это примеры прав пользователя, определенных Windows NT. Существует также понятие **тиражирования каталогов**, т.е. автоматического копирования главного набора каталогов с сервера, называемого сервером экспорта, на выбранные серверы или рабочие станции (называемые компьютерами импорта) в том же или другом домене. Тиражирование упрощает поддержание идентичности наборов каталогов на нескольких компьютерах, так как следить надо только за главным набором. Файлы тиражируются каждый раз при добавлении в каталог экспорта или при сохранении в них изменений.

*Субъект (subject)* – совокупность маркера доступа пользователя и приложения, действующего в интересах пользователя. Windows NT использует субъекты для отслеживания и управления разрешениями для всех выполняемых пользователем программ. Когда программа или процесс выполняются в интересах пользователя, считается, что выполнение производится в контексте безопасности (security context) этого пользователя. Контекст безопасности управляет доступом субъекта к объектам и сервису системы. Для реализации модели «клиент-сервер» Windows NT архитектура безопасности Windows NT включает два класса субъектов:

1. *Простой субъект (simple subject)* – процесс, которому был назначен контекст безопасности, когда соответствующий ему пользователь осуществил вход в систему. Не действует применительно к защищенным серверам, которые могут иметь других субъектов в качестве клиентов.
2. *Субъект сервера (server subject)* – процесс, выполняемый как защищенный сервер (типа подсистемы Win32) и имеющий другие субъекты в качестве клиентов. В этом случае субъект сервера обычно имеет контекст безопасности клиента, инициировавшего действия.

Вообще, при запросе сервера через защищенную подсистему маркер доступа субъекта используется внутри сервера, чтобы определить, кто произвел запрос, и решить, имеет ли запрашивающая сторона достаточные права доступа для выполнения запрашиваемого действия. Windows NT позволяют одному процессу заимствовать атрибуты безопасности другого через технику, называемую воплощением (impersonation). Например, процесс сервера обычно становится воплощением процесса клиента при выполнении задачи с задействованием объектов, к которым сервер в обычной ситуации не имеет доступа. В сценарии, показанном на рис. 2.17, клиент осуществляет доступ к объекту сервера Windows NT.

Первая нить в процессе – нить управления. Она ожидает получения RPC через именованный канал. Эта нить не является воплощением других процессов, поэтому любая проверка правильности доступа этой нити будет выполнена на основании первичного маркера процесса. Вторая нить процесса обрабатывает вызов от клиента. Эта нить

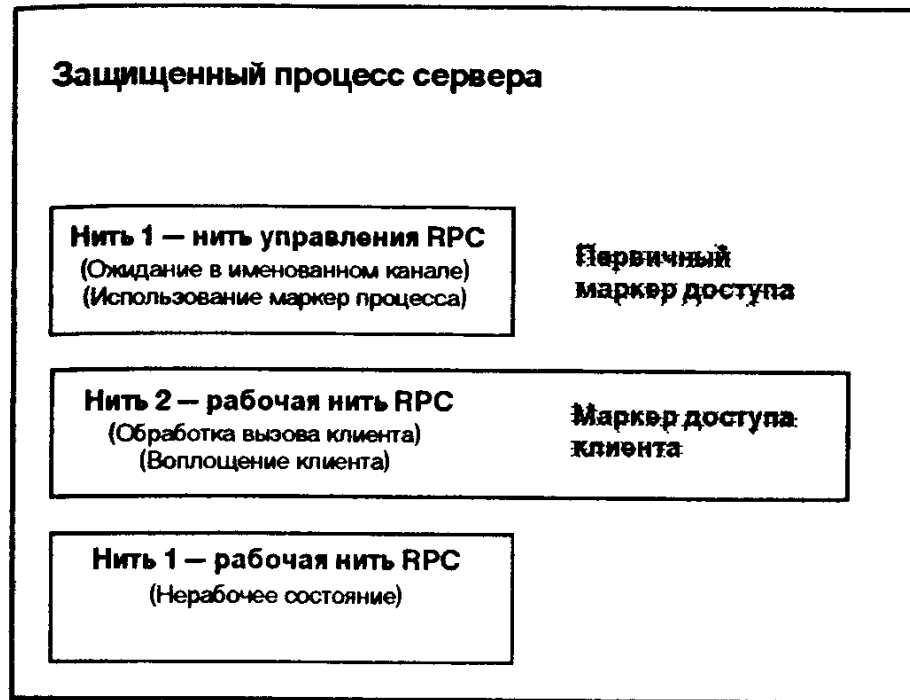


Рис. 2.17

производит обработку с временным использованием маркера доступа клиента с тем, чтобы выполняться с разрешениями доступа клиента (т.е. контекст безопасности клиента). В течение времени воплощения любая проверка правильности доступа этой нити будет выполнена в контексте безопасности клиента. Третья нить в этом сценарии – неактивная рабочая нить, не являющаяся воплощением других процессов.

Все именованные объекты в Windows NT, а также некоторые неименованные могут быть объектами безопасности. Атрибуты безопасности для объекта описываются дескриптором безопасности (security descriptor). Дескриптор безопасности объекта включает четыре части (рис. 2.18): 1) идентификатор безопасности владельца, который указывает пользователя или группу, являющихся владельцем объекта. Владелец объекта может изменять разрешения доступа для объекта; 2) идентификатор безопасности группы, который используется только подсистемой POSIX и игнорируется остальной частью Windows NT; 3) контролируемый список управления доступом (ACL – access control list), который идентифицирует, каким пользователям и группам дается или не дается разрешение доступа. Контролируемый ACL управляется владельцем

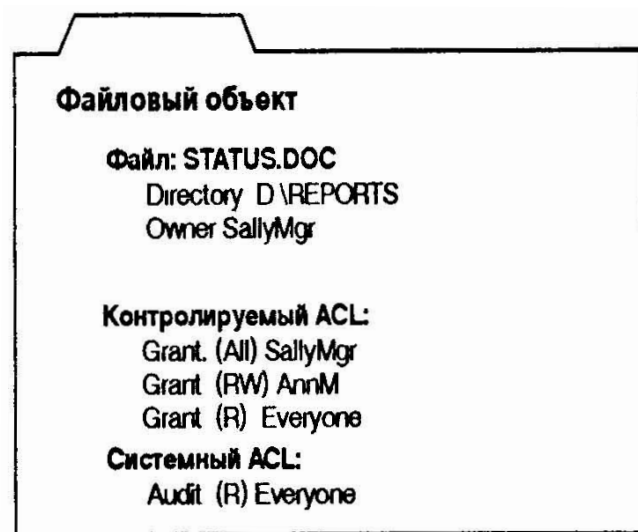


Рис. 2.18

объекта; 4) системный ACL, который управляет списком генерируемых системой сообщений контроля.

Системные ACL контролируются администраторами безопасности.

Каждый ACL состоит из элементов управления доступом (ACE – access control entries), которые специфицируют доступ или разрешения контроля применительно к объекту для отдельного пользователя или группы. Существуют три типа ACE: два для контролируемого управления доступом и один для безопасности системы. Контролируемыми ACE являются AccessAllowed и AccessDenied. Соответственно, они явно предоставляют и отвергают доступ для пользователя или группы пользователей. Существует важное отличие между пустым контролируемым ACL (не имеющим в своем составе ACE) и объектом без какого-либо контролируемого ACL. В случае пустого контролируемого ACL никакой доступ явно не предоставляется (т.е. доступ неявно отклонен). Для объекта без ACL не существует какой-либо защиты (т.е. предоставляется любой запрашиваемый доступ).

SystemAudit представляет собой ACE безопасности системы, используемый для хранения журнала безопасности (типа того, кто обращается к конкретным файлам), генерирования и регистрации сообщений контроля безопасности. Каждый ACE включает маску доступа (access mask), которая определяет возможные действия (рис. 2.19). Разрешения предоставляются или отклоняются на основании этой маски доступа. Одна из возможных аналогий для маски доступа –

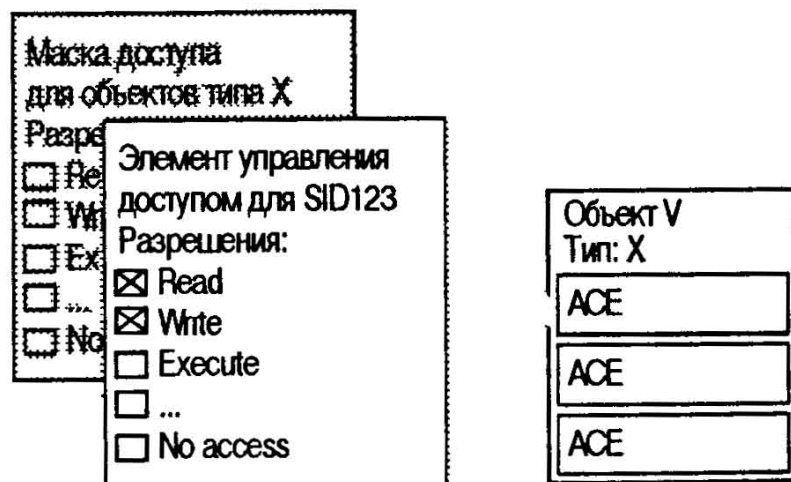


Рис. 2.19

разновидность меню, из которого выбраны предоставления и отклонения разрешений.

Специфические типы (specific types) включают параметры доступа, которые применяются специально к этому типу объекта. Каждый тип объекта может иметь до 16 специфических типов доступа. Совокупность специфических типов доступа для специфического типа объекта называется специфической маской доступа (specific access mask; определяется в момент определения типа объекта). Например, файлы Windows NT имеют следующие специфические типы доступа: ReadData (чтение данных); WriteData (запись данных); AppendData (присоединение данных); ReadEA (расширенное чтение); WriteEA (расширенная запись); Execute (выполнение); ReadAttributes (чтение атрибутов); WriteAttributes (запись атрибутов).

Стандартные типы относятся ко всем объектам и состоят из следующих разрешений доступа: SYNCHRONIZE – используется для синхронизации доступа и позволения процессу ожидания объекта для вывода сообщаемого состояния; WRITE\_OWNER – используется для назначения владельца записи; WRITE\_DAC – используется для предоставления или отклонения доступа на запись для контролируемого ACL; READ\_CONTROL – используется для предоставления или отклонения доступа на чтение для дескриптора безопасности и владельца; DELETE – используется для предоставления или отклонения доступа на удаление.

Объекты могут быть классифицированы как контейнерные или неконтейнерные. Контейнерные объекты (типа каталога) могут логически содержать другие объекты; неконтейнерные объекты (типа файла) других

объектов не содержат. По умолчанию, при создании новых объектов внутри контейнерного объекта новые объекты наследуют разрешения из родительского объекта (parent object). Для случая файлов и каталогов, когда пользователь изменяет разрешения каталога, эти изменения распространяются на каталог и его файлы (следует заметить, что это будет сделано только при выбранной опции Replace Permissions on Existing Files – перенос разрешений на существующие файлы), но не переносятся автоматически на существующие подкаталоги и их содержимое. Для применения измененных разрешений к существующим подкаталогам и их файлам необходимо выбирать опцию Replace Permissions on Subdirectories (перенос разрешений на подкаталоги).

Когда пользователь пытается обратиться к объекту, Windows NT сравнивает информацию безопасности в маркере доступа пользователя с информацией безопасности в дескрипторе защиты объекта (рис. 2.20).

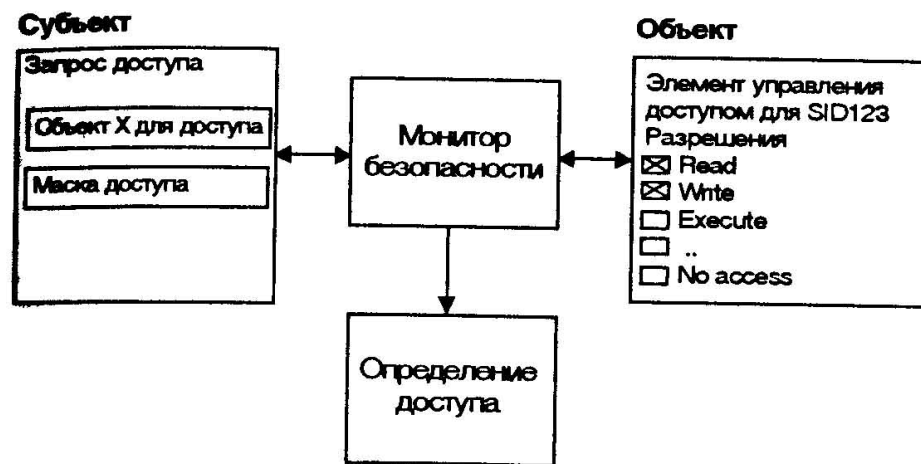


Рис. 2.20

Запрашиваемая маска доступа (desired access mask) создается для субъекта на основании типа доступа, который пользователь пытается получить. Эта запрашиваемая маска доступа, обычно создаваемая приложением пользователя, сравнивается с ACL объекта (все общие типы доступа в ACL отображаются как стандартные и специфические типы доступа).

Windows NT включает возможности контроля (auditing), который пользователь может применять для сбора информации относительно того, каким образом используется система. Эти возможности также позволяют контролировать события, связанные с безопасностью системы, идентифицировать любые нарушения безопасности и определять степень и местоположение любого повреждения. Уровень

контролируемых событий корректируем для достижения максимального удовлетворения потребностей любой организации. Некоторые организации нуждаются в небольшом количестве контрольной информации, в то время как другие готовы пожертвовать некоторой эффективностью и диском-пространством для получения детализированной информации, которую могут использовать для тщательного анализа системы. Отметим, что при осуществлении контроля возникают определенные затраты ресурсов системы для каждой осуществляемой проверки.

Windows NT может отслеживать события, связанные непосредственно с операционной системой или индивидуальным приложением. Каждое приложение может назначать собственные контролируемые события. Определения этих событий добавляются к реестру в момент установки приложения на компьютер с Windows NT.

Контролируемые события распознаются системой по имени вызвавшего событие модуля (который соответствует специфическому событию, записываемому в реестре) и идентификатору события. В дополнение к отображению событий по их идентификаторам, журнал безопасности в Event Viewer перечисляет их по категориям.

Внутри маркера доступа процесса Windows NT отслеживает следующую информацию: идентификатор безопасности бюджета пользователя, который был задействован при входе в систему; идентификатор безопасности групп и соответствующие атрибуты групп, в которых пользователь является членом; имена привилегий, назначенных и применяемых пользователем, а также соответствующие им атрибуты; идентификатор аутентификации, назначаемый в момент входа пользователя в систему.

### **2.5.2. Механизмы отказоустойчивости Windows NT**

Windows NT Server предлагает несколько механизмов отказоустойчивости: поддержку копирования на ленту (доступна и для Windows NT и Windows NT Server); использование источников бесперебойного питания (UPS); зеркальное копирование дисков (disk mirroring); дублирование дисков (disk duplexing); чередование дисков с контролем по четности (disk stripping with parity). Некоторые из этих механизмов могут использоваться совместно с любой файловой системой; некоторые – только с определенными файловыми системами. Ограничения рассматриваются в следующих разделах, подробно описывающих механизмы отказоустойчивости в Windows NT.

*Зеркальное копирование диска* – метод защиты против сбоев жесткого диска. Любая файловая система, включая FAT, HPFS и NTFS, может использовать зеркальное копирование дисков. Для зеркального копирования дисков используются два раздела на различных дисковых додах, управляемых одним контроллером диска. Все данные первичного раздела автоматически копируются во вторичный раздел.

Таким образом, при возникновении сбоя первичного диска никакие данные не теряются. Вместо сбойного раздела производится использование вторичного раздела диска. Для практического использования зеркальный раздел создается обычно с тем же размером, что и первичный. Зеркальный раздел не может иметь меньший размер. В случае если зеркальный раздел имеет больший размер, дополнительное пространство не задействуется.

*Дублирование дисков* – использование зеркального копирования дисков с применением дополнительного адаптера на вторичном дисковом. Это обеспечивает отказоустойчивость и при сбое контроллера и при сбое диска (использование нескольких адаптеров, соединяющихся с одним дисководом, не поддерживается). Кроме обеспечения отказоустойчивости, дублирование дисков может также повысить эффективность. Подобно зеркальному копированию, дублирование выполняется на уровне раздела. Для операционной системы Windows NT не существует различия между зеркальным копированием и дублированием. Это просто вопрос местонахождения другого раздела.

*Чередование дисков с контролем четности* – метод, при котором несколько разделов объединяются в отдельный логический привод (подобно описанному ранее чередованию дисков).

В рассматриваемом случае должно присутствовать от 3 до 32 дисков. Для каждого из дисков должен создаваться раздел примерно одинакового объема. Диски могут быть подключены к одному или различным контроллерам. Предпочтительно использование дисков SCSI, так как в процессе восстановления данных могут быть применены улучшенные возможности восстановления типа переотображения плохого блока. Данные чередуются через все используемые разделы. В дополнение с данными чередуется также информация контроля четности. Информация контроля четности – обычный байт контроля четности по строке или столбцу. Например, предположим, что чередование осуществляется для пяти дисков. Для уровня 0 присутствуют чередующиеся полосы 0 на диске 0, 1 на диске 1, 2 на диске 2, 3 на диске 3 и полоса контроля четности на диске 4. Размер полосы (обычно называемый фактором чередования – striping factor) принимается равным 64 Кб. Размер полосы контроля четности равен размеру полосы данных. Для следующей строки полоса контроля четности находится на диске 0. Данные содержатся на остальных дисках. Так как полосы контроля четности не сосредоточены на одном диске, никакой одиночный сбой не воздействует на работоспособность системы; кроме того, загрузка равномерно распределена.

При использовании любой из отказоустойчивых дисковых схем, Windows NT использует драйвер FTDISK.SYS, который получает команды и отвечает соответствующим образом на основании информации о типе используемого механизма отказоустойчивости. Таким образом, когда

файловая система генерирует запрос для чтения фрагмента файла, нормальная дисковая система получает запрос из файловой системы и передает его драйверу FTDISK.SYS.

Процессы обнаружения ошибок и восстановления очень похожи и для зеркального копирования, и для чередования с контролем четности. Конкретная реакция системы зависит от характера возникшей проблемы. Нарушенный набор (broken set) определяется, если в какой-то момент для одного или нескольких разделов, участвующих в зеркальном копировании или дублировании, не может быть произведена запись; это также распространяется на невозможность записи для чередования дисков. При обнаружении ошибки ввода-вывода в первый раз, система выполняет некоторые специальные процедуры с целью исправить ситуацию. Первым системным действием является переотображение (remap) сбойного сектора. Windows NT делает попытку переотображения, если диск оснащен контроллером SCSI. SCSI-устройства разработаны так, чтобы поддерживать концепцию переотображения. Благодаря этому SCSI-устройства хорошо работают в качестве отказоустойчивых устройств (обратите внимание, что некоторые фиксированные жесткие диски также поддерживают концепцию переотображения; но подобная поддержка не опирается на стандарт). Если диск не поддерживает отображение сектора или если другие попытки исправления ситуации терпят неудачу, в журнал регистрации событий заносится серьезная ошибка (high severity error). Сбойный раздел называется подвисшим (orphan). Важно обратить внимание, что фиксирование подвисания раздела происходит не в процессе чтения, а только в процессе записи. Это вызвано тем, что чтение не может воздействовать на дисковые данные; выполнение обработки подвисания является лишним.

В процессе инициализации системы, если система не может определить каждый раздел зеркального набора, в журнал регистрации событий заносится серьезная ошибка и используется оставшийся раздел зеркального набора. Если раздел является частью чередующихся дисков с контролем четности, то в журнал регистрации событий заносится серьезная ошибка и раздел отмечается как подвисший. Далее система продолжает функционирование с использованием свойственных ей отказоустойчивых возможностей. Если не определяются все разделы внутри набора и активизация привода не производится, то разделы не отмечаются как подвисшие. Это позволяет снизить время восстановления из-за простых неисправностей, типа отсоединения цепочки SCSI от компьютера. После того как раздел отмечен в качестве подвисшего, система продолжает функционирование, пока заменой диска или доступного раздела не будет разрешена проблема и гарантирована отказоустойчивость. Набор с подвисшим компонентом не является отказоустойчивым. Следующий отказ в наборе может стать (и с большой вероятностью станет) причиной потери данных.

## 2.6. Администрирование и мониторинг

### 2.6.1. Администрирование Windows NT

**УЧЕТНЫЕ ЗАПИСИ ПОЛЬЗОВАТЕЛЕЙ.** Учетные записи пользователей (user accounts) – основа безопасности Windows NT. Все происходящее в сети Windows NT можно проследить до учетной записи пользователя, выполнившего некоторое действие. Можно проследить и за несанкционированной деятельностью. Рассмотрим учетные записи подробнее. Так как в Windows NT реализована модель разграничения доступа на уровне пользователей, а не ресурсов, администратор обязан создать учетную запись для каждого пользователя, желающего получить доступ к управляемым ОС ресурсам. Когда пользователь регистрируется в системе, SAM сравнивает предоставленную информацию с имеющейся в базе учетных записей. Если аутентификация прошла успешно, LSA создает для пользователя уникальный *маркер доступа* (security token), который применяется для контроля привилегий и разрешений пользователя.

В зависимости от реализованной в сети модели доменов (существует также понятие **домена учётных записей** – домена, в котором хранятся учётные записи пользователей при отсутствии каких-либо ресурсов) пользователь может иметь несколько учетных записей с различным уровнем доступа к рабочим группам и доменам. Эта информация может находиться на компьютере пользователя (в случае учетной записи для рабочей группы), а также на сервере или контроллере домена Windows NT. Важно, не где она находится, а как используется.

Учетная запись пользователя определяет для него разрешения на доступ к ресурсам. Разрешения каждому конкретному пользователю дает администратор. Доступ к ресурсу возможен, только если у пользователя есть необходимые разрешения. Учетные записи также применяются в Windows NT для ведения **аудита** – audit trail (проверки элементов управления системой). Существуют также понятия **аудита каталогов** (отслеживания использования одного или нескольких каталогов) и **аудита приложений** (регистрации событий, связанных с попыткой открыть ключ реестра или извлечь из него информацию). Существует возможность проследить за большинством аспектов использования ресурсов; в журнале событий (Event Log) можно увидеть, в результате какого действия пользователя появилась запись о событии. Подобная идентификация оказывает неоценимую помощь в том, чтобы обнаружить и остановить потенциального злоумышленника.

**ТИПЫ УЧЕТНЫХ ЗАПИСЕЙ.** В Windows NT поддерживаются два основных вида учетных записей: глобальные (global) и локальные (local). **Глобальная учётная запись** – обычная учётная запись пользователя в домашнем домене. Если в сети существует несколько доменов, то

предпочтительно, чтобы каждый пользователь в сети имел одну учётную запись и только в одном домене; доступ пользователя к ресурсам других доменов должен осуществляться через систему доверительных отношений. **Доверительные отношения** – разновидность связи между доменами, предусматривающая выполнение аутентификации, когда пользователь, имея учётную запись только в одном домене, может обращаться ко всей сети. **Доверяющий домен** – домен, предоставляющий доступ к своим ресурсам пользователям других, доверяемых, доменов. **Доверяемый домен** – домен, пользователи которого могут осуществлять доступ к ресурсам других, доверяющих, доменов. Эти учётные записи располагаются в «домашнем» (home) домене пользователя и могут использоваться для доступа к ресурсам других доменов.

**Локальные учётные записи** назначаются тем пользователям домена, чьих глобальных учётных записей нет в *доверяемом* (trusted) домене. Такие учётные записи аутентифицируются только на основе информации на том компьютере, где происходит регистрация. Локальные учётные записи можно использовать только в том домене, где они были созданы; их нельзя применять для доступа к ресурсам других, доверяемых доменов. Как правило, локальные записи нужны для пользователей WfW и LAN Manager.

**АУТЕНТИФИКАЦИЯ.** Компьютеры с Windows NT, не участвующие в домене, используют системный сервис Netlogon, чтобы обработать запрос на регистрацию на локальной машине и, если необходимо, *передать регистрационные данные* (logon information) на сервер домена. Netlogon тиражирует любые изменения баз данных безопасности, включая базы данных SAM и LSA, на все контроллеры домена. Сервис Netlogon, работающий на сервере Windows NT, синхронизирует свою базу данных в следующих случаях: при первой установке контроллера домена; когда снова включается в работу отключенный контроллер домена, а его журнал регистрации изменений переполнен. Netlogon также обрабатывает запросы на регистрацию и извлекает информацию из базы данных SAM. При регистрации пользователя на компьютере, не являющемся контроллером домена Windows NT, система проводит *поиск контроллера домена* (discovery process). Если компьютер не участвует в домене, Netlogon прекращает обработку запроса на регистрацию.

Связь между системными компонентами двух сетевых компьютеров с Windows NT всегда осуществляется по защищенному каналу. Он создается при помощи комбинации сообщений «*вопрос-ответ*» (challenge-and-response), генерируемых и обрабатываемых сервисом Netlogon обоих компьютеров, и используется для передачи вызовов процедур API, имен пользователей и их зашифрованных паролей. Для обеспечения безопасности после установления защищенного канала Netlogon использует специальные встроенные учётные записи: *Workstation trust accounts* – позволяют находящейся в домене рабочей

станции проводить сквозную аутентификацию при обращении к серверу домена Windows NT; *Server trust accounts* – обеспечивают серверу Windows NT возможность получить от контроллера домена копию базы данных главного домена; *Interdomain trust accounts* – позволяют серверу Windows NT проводить сквозную аутентификацию при обработке запроса на регистрацию, поступившего из другого домена.

*Сквозная аутентификация* (pass-through security) применяется тогда, когда локальный компьютер оказывается не в состоянии аутентифицировать учетную запись. Имя пользователя и его пароль передаются серверу Windows NT, который сам будет пытаться провести аутентификацию. Для корректной работы этого механизма необходимо, чтобы на каждой рабочей станции домена функционировали сервисы Netlogon и Workstation. При этом локальный сервис Netlogon направляет запрос удаленному и ожидает от него ответа.

#### УПРАВЛЕНИЕ УЧЕТНЫМИ ЗАПИСЯМИ. РАЗРЕШЕНИЯ И ПРИВИЛЕГИИ.

Для управления учетными записями в Windows NT можно использовать User Manager или User Manager for Domains. Эти приложения имеют графический интерфейс и позволяют контролировать все параметры учетных записей. С помощью User Manager можно управлять аудитом, добавлять, изменять и удалять учетные записи. Следует обратить внимание на одно существенное различие между привилегиями и разрешениями: *привилегии* (rights) относятся к системным задачам, а *разрешения* (permissions) – к объектам.

Существует два вида привилегий – *обычные* (regular) (табл. 2.5) и *дополнительные* (advanced). Все они дают пользователю возможность выполнять некоторые общесистемные задачи, например, изменять настройки совместно используемого принтера, проводить резервное копирование и восстановление данных. При работе с программой User Manager или User Manager for Domains вы обязательно столкнетесь с такими привилегиями, смысл которых вначале будет непонятен.

Разрешения применяются для доступа к конкретным объектам, например, файлам, каталогам или принтерам. Типичное разрешение открывает доступ к какому-нибудь каталогу или принтеру. Назначением разрешений и управлением ими занимается *создатель* (creator) или *владелец* (owner) объекта. Только системный администратор может переопределить установленные разрешения и только после того, как он вступит во владение (takes ownership) объектом. По соображениям безопасности смена владельца объекта фиксируется в журнале регистрации.

Обычно привилегии пользователей имеют приоритет перед разрешениями на доступ. Покажем это на конкретном примере с группами пользователей, которые рассматриваются ниже. Пока же имейте в виду, что группы – это просто объединения пользователей с общими привилегиями. Пусть пользователь из группы *Backup Operators*

Таблица 2.5

## Обычные привилегии пользователей

Привилегия	Позволяет
Access this computer from network	Подсоединяться к компьютеру по сети
Change the system time	Устанавливать системное время
Back up files and directories	Выполнять резервное копирование
Force shutdown from a remote system	Удаленно выключать систему. Не реализовано
Log on locally	Входить в систему с подключенной к данной машине клавиатуры
Manage auditing and security log	Устанавливать параметры аудита, просматривать и очищать журнал безопасности (Security Log)
Restore/ies and directories	Выполнять восстановление с резервной копии
Shut down the system	Завершать работу операционной системы
Take ownership of files and objects	Вступать во владение объектами, которые, возможно, принадлежат другим пользователям

проводит резервное копирование системы. Если какой-нибудь другой пользователь создал каталог и запретил доступ к нему членам этой группы, любой из них все равно сможет его получить на основании привилегии на выполнение резервного копирования, тем самым переопределяя разрешения, назначенные владельцем или создателем каталога. Действительно позволяющая вступать во владение объектами привилегия *Take ownership of files and objects* по умолчанию имеется только у системного администратора. Однако, как и всякую другую, эту привилегию можно назначить любому пользователю.

**ГРУППЫ ПОЛЬЗОВАТЕЛЕЙ.** Понятие *группы* (group) в Windows NT позволяет назначать наборы разрешений и привилегий нескольким пользователям одновременно. Когда группе предоставляется разрешение или привилегия, они автоматически распространяются на всех входящих в нее пользователей. Для определения групп и управления ими служат программы User Manager или User Manager for Domains. В Windows NT используется три вида групп – *глобальные* (global), *локальные* (local) и *специальные* (special). Глобальные группы содержат

учетные записи пользователей только из того домена, где они созданы. Никакие группы не могут входить в глобальные. Глобальным группам могут предоставляться разрешения и привилегии, относящиеся к содержащим их доменам и тем доменам, которым они доверяют. Это удобное средство, чтобы экспортировать объединения пользователей в другие домены как единое целое.

В Windows NT три глобальные группы создаются по умолчанию: *Domain Admins*, *Domain Users* и *Domain Guests*. Первой предоставляется право изменять определенные характеристики домена и управлять ими. У второй прав меньше, что ограничивает возможное управление доменом. Последняя группа, *Domain Guests*, имеет еще меньшие права, так как предназначена для временных пользователей сети.

Локальные группы содержат учетные записи пользователей и глобальные группы других доменов. Локальной группе можно назначать права только по отношению к домашнему домену, и она может использоваться только на том компьютере, где создана. При установке Windows NT в роли сервера создается меньше локальных групп, чем при установке контроллера домена. В табл. 2.6 перечислены стандартные локальные группы и даны их описания.

При установке сервера Windows NT в роли контроллера домена автоматически создаются дополнительные группы, перечисленные в табл. 2.7.

В табл. 2.8 описаны специальные группы, с которыми вы можете столкнуться при назначении разрешений и привилегий. Под **специальной группой** понимается группа, члены которой не назначаются администратором. Пользователь становится членом такой группы при выполнении определенных действий в сети. Например, пользователь, выполнивший интерактивную регистрацию, принадлежит к специальной группе *Interactive*.

Использование всех перечисленных выше групп не является обязательным, за исключением группы *Administrators*, которая необходима для управления системой. Все они существуют для вашего удобства, и вы должны применять эти группы в соответствии с теми возможностями, которые они предоставляют пользователям. Решая, в какую группу добавить пользователя, обратитесь к приведенным выше описаниям или создайте новую группу и предоставьте ей необходимые привилегии. Ошибки при включении пользователей в группы или при определении их разрешений и привилегий могут нанести существенный урон безопасности системы.

### 2.6.2. Мониторинг Windows NT

Вместе с Windows NT поставляется несколько превосходных инструментов мониторинга, которые полезны при сборе и анализе информации о системе. Все они имеют свои функции и служат

Таблица 2.6

Группа	Описание
Administrators (Администраторы)	Группа, обладающая наибольшими возможностями, ее члены могут администрировать все аспекты домена или сервера NT. Необходимо быть осторожным при добавлении в нее пользователей. Эта группа – единственная, использование которой обязательно
Users (Пользователи)	Члены этой группы обладают минимальным набором привилегий на сервере NT. Они имеют возможность создавать локальные группы и управлять ими, если могут регистрироваться в системе и имеют доступ к программе User Manager
Guests (Гости)	Эта группа предназначена для нерегулярных и случайных пользователей. Как правило, ее членам предоставляются очень ограниченные привилегии
Backup Operators (Операторы резервирования)	Эти пользователи уполномочены проводить резервное копирование и восстановление файлов и каталогов. При необходимости они также могут завершить работу сервера
Replicator (оператор тиражирования)	Эту группу ОС использует для регистрации в системе сервиса тиражирования (Replicator). Она содержит единственную учетную запись пользователя домена, и других учетных записей в ней не должно быть
Everyone (Каждый пользователь)	Специальная группа, не отображаемая в User Manager. Однако вы можете использовать ее при определении разрешений на доступ к ресурсам. В нее входят все учетные записи домена, и ее наличие призвано упростить создание совместно используемых ресурсов с неограниченным доступом
Power Users (Подготовленные пользователи)	Входящие в эту группу пользователи могут создавать сетевые устройства и подключаться к ним, тем самым они способны нарушить установленную вами защиту

Таблица 2.7

Группа	Описание
Account Operators (Операторы счёта)	Члены этой группы могут использовать User Manager for Domains, но с некоторыми ограничениями. Они свободно могут создавать, изменять и удалять локальные и глобальные учетные записи и группы – кроме групп <i>Administrators</i> , <i>Domain Admins</i> , <i>Account Operators</i> , <i>Backup Operators</i> , <i>Print Operators</i> и <i>Server Operators</i> . Они не могут ничего делать с группой <i>Administrators</i> или изменять системные политики. Однако они могут при помощи Server Manager добавлять в домен компьютеры, входить в систему на сервере (с клавиатуры) и завершать ее работу
Print Operators (Операторы печати)	Эти пользователи могут создавать, изменять и удалять совместно используемые принтеры. Они также могут входить на серверы Windows NT и завершать их работу
Server Operators (Операторы сервера)	Данная группа обладает всеми привилегиями, необходимыми для управления серверами домена. Входящим в нее пользователям разрешено управлять принтерами, совместно используемыми сетевыми ресурсами, проводить резервное копирование и восстановление, форматировать диски, изменять системное время и дату, входить на сервер и завершать его работу

Таблица 2.8

Группа	Описание
Interactive	Все пользователи, которые зарегистрированы на компьютере локально
Network System	Все, кто подключен к компьютеру по сети. Сама операционная система
Creator/Owner	Создатель/владелец объекта (файла, каталога, задания на печать и т. д.)

конкретным целям. В этом разделе рассматриваются встроенные в Windows NT средства мониторинга, их функциональные особенности и возможности.

**ПРОГРАММА EVENT VIEWER.** На рис. 2.21 изображено окно программы Event Viewer, самого важного инструмента, имеющегося в вашем распоряжении. Event Viewer позволяет просматривать и анализировать три основных журнала регистрации: *системный журнал* (system log), *журнал безопасности* (security log) и *журнал приложений* (application log). В каждом из них фиксируются определенные типы событий: *системный журнал* включает события, относящиеся к деятельности самой ОС (загрузка и выгрузка драйверов, запуск и остановка сервисов и т.д.); *в журнале безопасности* фиксируются события, связанные с защитой, например, неудачные попытки зарегистрироваться в системе или получить доступ к ресурсам; *журнал приложений* включает записи о событиях, вызванных конкретными приложениями, например, выполняющим самодиагностику программным обеспечением источника бесперебойного питания или программой, завершившей резервное копирование.

Значение этой программы в общей схеме обеспечения безопасности невозможно переоценить. Необходимо регулярно просматривать все журналы регистрации, если надо надежно защитить свою систему (большинство вторжений случается в защищенных системах, за которыми неправильно или нерегулярно наблюдали).

**ПРОГРАММА SERVER MANAGER.** Это небольшая программа, которая позволяет в любой момент взглянуть на некоторые части Windows NT сервера. Она имеет два основных окна: отображения компьютеров – членов домена и отображения информации о конкретном компьютере домена.

В окне Server Manager отображается список всех компьютеров домена с указанием работающих на них ОС. Просматривая этот список, вы можете дважды щелкнуть мышью имя конкретного компьютера и увидеть его характеристики. У нижнего края диалогового окна Properties расположены несколько кнопок: Users, Shares, In Use, Replication и Alerts. С их помощью можно получить доступ к конкретным видам информации о компьютере. Рассмотрим их подробнее. При нажатии на кнопку появляется диалоговое окно:

- Users – перечисляются подключенные в данный момент к серверу пользователи; число открытых ими ресурсов; имена компьютеров, с которых выполнено подключение; время подсоединения; время простоя пользователя; список ресурсов, с которыми сейчас работает пользователь. Отключить пользователя от одного или всех ресурсов сразу можно, воспользовавшись соответствующими кнопками в нижней части окна;

Event Viewer - System Log on \\SERVER_PII						
Log View Options Help						
Date	Time	Source	Category	Event	User	Computer
31.01.00	11:12:49	Print	None	10	pastuhov	SERVER_PII
31.01.00	10:52:01	Rdr	None	3012	N/A	SERVER_PII
31.01.00	10:41:59	BROWSER	None	8015	N/A	SERVER_PII
31.01.00	10:41:59	BROWSER	None	8015	N/A	SERVER_PII
31.01.00	10:41:59	BROWSER	None	8015	N/A	SERVER_PII
31.01.00	10:41:59	BROWSER	None	8015	N/A	SERVER_PII
31.01.00	10:40:32	EventLog	None	6005	N/A	SERVER_PII
31.01.00	10:40:32	EventLog	None	6009	N/A	SERVER_PII
31.01.00	10:41:40	BROWSER	None	8021	N/A	SERVER_PII
31.01.00	10:38:32	EventLog	None	6006	N/A	SERVER_PII
31.01.00	10:38:31	BROWSER	None	8033	N/A	SERVER_PII
31.01.00	10:38:29	BROWSER	None	8033	N/A	SERVER_PII
31.01.00	10:38:29	BROWSER	None	8033	N/A	SERVER_PII
31.01.00	10:38:29	BROWSER	None	8033	N/A	SERVER_PII
31.01.00	9:33:14	BROWSER	None	8015	N/A	SERVER_PII
31.01.00	9:33:14	BROWSER	None	8015	N/A	SERVER_PII
31.01.00	9:33:14	BROWSER	None	8015	N/A	SERVER_PII
31.01.00	9:33:13	BROWSER	None	8015	N/A	SERVER_PII
31.01.00	9:33:04	BROWSER	None	8021	N/A	SERVER_PII
31.01.00	9:31:40	EventLog	None	6005	N/A	SERVER_PII
31.01.00	9:31:40	EventLog	None	6009	N/A	SERVER_PII
31.01.00	9:32:51	BROWSER	None	8021	N/A	SERVER_PII
28.01.00	17:09:18	EventLog	None	6006	N/A	SERVER_PII
28.01.00	17:09:17	BROWSER	None	8033	N/A	SERVER_PII
28.01.00	17:09:16	Rdr	None	8005	N/A	SERVER_PII
28.01.00	17:09:15	BROWSER	None	8033	N/A	SERVER_PII
28.01.00	17:09:15	BROWSER	None	8033	N/A	SERVER_PII
28.01.00	17:09:15	BROWSER	None	8033	N/A	SERVER_PII
28.01.00	16:33:19	Rdr	None	3012	N/A	SERVER_PII

Рис. 2.21

- Shares – в нем находится информация об объектах, являющихся совместно используемыми ресурсами сети: их имена, число подключений, фактическое местоположение, имена подключенных пользователей и время их подсоединения. Вы увидите также, используется ли ресурс в данный момент;
- In Use – в нем содержится информация о ресурсах системы, выделенных в данный момент для совместного использования: общее число таких ресурсов: число установленных в данный момент блокировок файлов (file locks), имена обратившихся к ресурсам пользователей; вид доступа к отдельным ресурсам; число блокировок для каждого ресурса, путь к фактическому местоположению ресурса;
- Replication – в нем перечислены каталоги, выделенные для экспорта и импорта данных в заданной системе. В этом окне можно устанавливать и изменять их параметры;
- Alerts – в нем перечислены получатели административных сигналов тревоги (administrative alerts), отправляемых данной системой. Вы можете добавлять в этот список пользователей и компьютеры и удалять их из него.

Таким образом, Server Manager объединяет несколько диалоговых окон, облегчающих управление сервером. Кроме описанных выше, Server Manager имеет и другие полезные возможности: отображение списка всех совместно используемых ресурсов для указанного сервера; создание новых совместно используемых ресурсов; изменение разрешений на доступ к совместно используемым ресурсам; создание списка всех сервисов, работающих на удаленном компьютере с Windows NT; остановка и запуск отдельного сервиса на компьютере с Windows NT; изменение параметров запуска сервиса; отправка сообщения всем пользователям компьютера с Windows NT; синхронизация всех контроллеров одного домена; добавление компьютеров в домен и их удаление; перечисление всех серверов домена; перечисление всех рабочих станций домена; перечисление всех компьютеров домена;

**ПРОГРАММА NETWORK MONITOR.** Эта утилита, предназначенная для сбора и анализа информации о сетевых пакетах, часто остается незамеченной в арсенале системных средств Windows NT. Network Monitor (рис. 2.22) с интерфейсом администратора устанавливается в виде сервиса Windows NT 4.0 и позволяет перехватывать сетевой трафик на том компьютере, где работает. Для перехвата пакетов с других компьютеров нужно использовать версию Network Monitor, поставляемую Microsoft с Systems Management Server.

Network Monitor позволяет задавать фильтры для перехвата конкретных типов пакетов, ограничивая тем самым объем перехваченных данных. Вы можете просматривать различную информацию о перехваченных пакетах: адреса сетевых адаптеров и IP-адреса

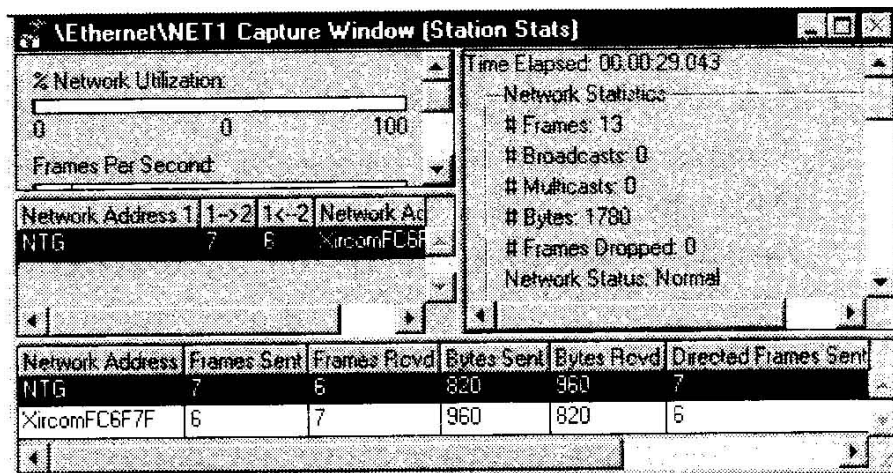


Рис. 2.22

отправителя и получателя, тип протокола, описание содержащихся в пакете данных, а также другие сведения, в зависимости от типа пакета и используемого протокола. На рис. 2.23 показано несколько перехваченных пакетов. Первый из них отправлен узлом с именем NTG, запрашивающим IP-адрес узла NS2.Necropolis.net у сервера DNS по адресу 128.9.0.107.

Перехваченные пакеты могут быть бесценным источником информации при диагностике сетевых неисправностей. Средства наблюдения

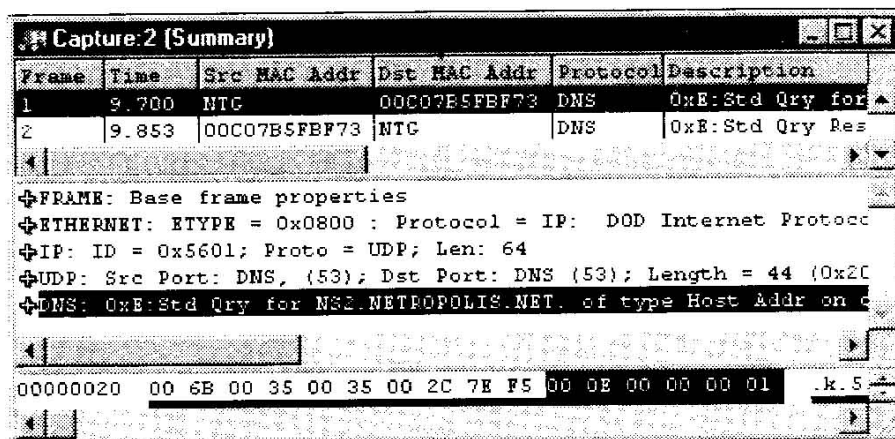


Рис. 2.23

за сетевой активностью могут быть незаменимыми помощниками при обнаружении неисправностей и особенно при отслеживании потенциальных взломщиков.

**ПРОГРАММА PERFORMANCE MONITOR.** Встроенная в Windows NT утилита Performance Monitor – прекрасный инструмент анализа. Как следует из названия, она предназначена для изучения характеристик производительности системных сервисов и процессов. Интерфейс Performance Monitor показан на рис. 2.24. Эта программа позволяет получать информацию как о компьютере, на котором работает, так и о других машинах сети.

С помощью Performance Monitor можно наблюдать за поведением таких системных объектов, как процессоры, память, кэш, потоки (threads), процессы и сетевой трафик. Каждый наблюдаемый объект характеризуется набором счетчиков (counters), указывающих, например, длину очередей, интенсивность использования устройства, задержки, производительность и внутреннюю загруженность. Имеющиеся в Performance Monitor средства построения диаграмм, оповещения и создания отчетов отражают не только текущую, но и прошедшую активность. Вы можете открывать файлы регистрации, просматривать их и создавать графики так, как будто имеете дело с текущим состоянием системы. Кроме того, Performance Monitor позволяет:

- просматривать данные, собранные одновременно с нескольких компьютеров;
- просматривать и изменять графики текущей активности;
- наблюдать показания счетчиков, снимаемые через заданные интервалы времени;
- экспортировать данные графиков, файлов регистрации, оповещений и отчетов в электронные таблицы и базы данных для дальнейшей обработки, просмотра и печати;
- добавлять системные оповещения (system alerts), при которых будет происходить регистрация событий в журнале Alert Log при этом возможна регистрация события в журнале приложений программы Event Viewer, отправка уведомления по сети или переключение программы в режим просмотра оповещений (Alert view);
- выполнять определенную программу, когда счетчик превышает заданное значение – всегда или только в первый раз;
- создавать файлы регистрации, содержащие данные об объектах на разных компьютерах;
- присоединять выделенные разделы файлов регистрации к единому файлу, создавая долгосрочный архив;
- просматривать отчеты о текущей активности или создавать их на основе данных из файлов регистрации;
- сохранять индивидуальные настройки для графиков, оповещений,

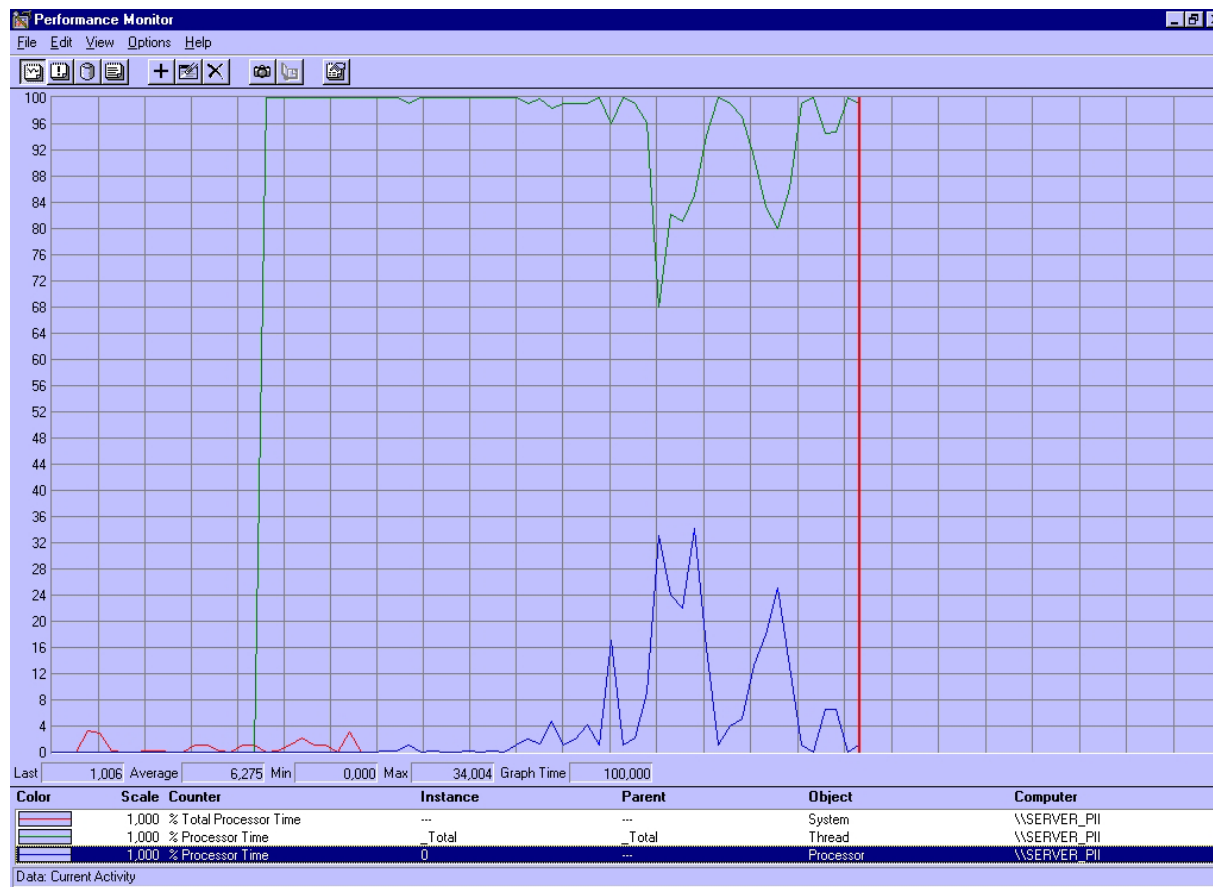


Рис. 2.24

файлов регистрации и отчетов или сохранять все параметры рабочего пространства.

При наблюдении за системой с помощью этого инструмента вы в действительности видите поведение объектов. Помните, что объект – это всего лишь стандартный способ идентификации и использования системного ресурса. Объекты представляют отдельные процессы, участки совместно используемой памяти, физические устройства. Зная это, вы можете применять Performance Monitor для выявления скрытых проблем системы, которые могут на самом деле оказаться ранними признаками вторжения. Например, если вы обнаружите, что сервис удаленного вызова процедур (Remote Procedure Call, RPC) использует 90% времени центрального процессора или более, у вас могут возникнуть подозрения, что кто-то пытается осуществить атаку типа «отказ в обслуживании». Короче говоря, данная программа представляет собой базовые средства оценки производительности системы, которые можно использовать для обнаружения отклонений от нормы.

**ПРОГРАММА TASK MANAGER.** Чтобы запустить Task Manager, просто нажмите Ctrl-Alt-Del и выберите соответствующий пункт меню. Эта программа предоставляет доступ к целому кладезю информации, кое в чем повторяя Performance Monitor. Она позволяет легко и быстро просматривать сведения о работающих процессах, распределении памяти, времени использования процессора и других жизненно важных данных. Давайте рассмотрим каждую вкладку основного окна Task Manager. Первая из них, Applications, изображена на рис. 2.25. В ней в колонке Task перечислены все работающие в данный момент приложения и их состояния. Используя это диалоговое окно, Вы также можете запустить новое приложение. На второй вкладке, Processes (рис. 2.26), в колонке Image Name указаны все исполняющиеся в системе процессы и приведены другие жизненно важные данные: идентификаторы процессов, время использования и степень загруженности процессора, объем занимаемой памяти. На рис. 2.27 показана третья вкладка, Performance, содержащая информацию об использовании процессора и памяти, графики этих параметров, общее количество *описателей* (handles), процессов и потоков, данные об использовании физической памяти и памяти ядра, а также *выделенной памяти* (commit charge usage). Многие ресурсы, информацию о которых отображает Task Manager, можно подробнее анализировать при помощи Performance Monitor. Тем не менее, вкладка Processes чрезвычайно полезна – на ней сразу видно, какие процессы расходуют больше всего процессорного времени и памяти.

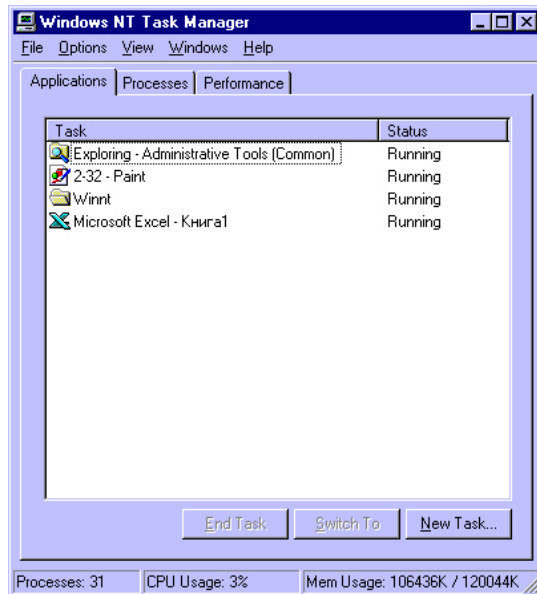


Рис. 2.25

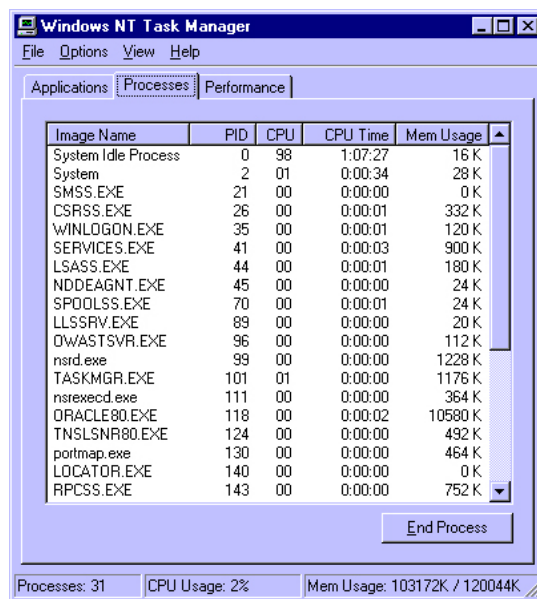


Рис. 2.26

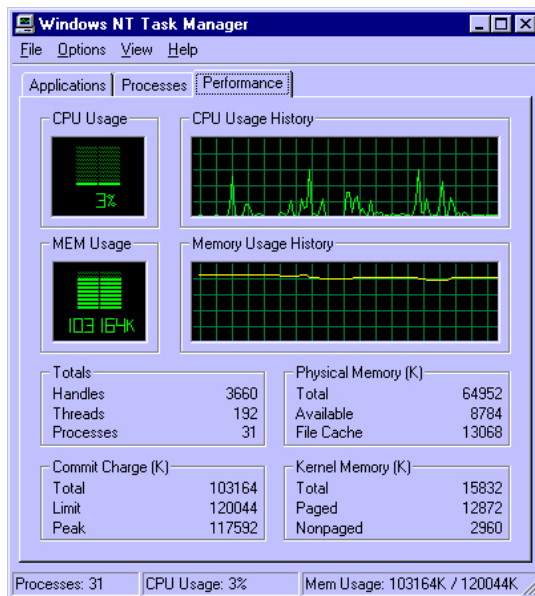


Рис. 2.27

## 2.7. Работа в глобальных сетях

### 2.7.1. Сетевые средства

Средства сетевого взаимодействия Windows NT направлены на реализацию взаимодействия с существующими типами сетей, обеспечение возможности загрузки и выгрузки сетевого программного обеспечения, а также на поддержку распределенных приложений. Windows NT с точки зрения реализации сетевых средств имеет следующие особенности: *встроенность на уровне драйверов*. Это свойство обеспечивает быстрое действие; *открытость* – обуславливается легкостью динамической загрузки-выгрузки, мультиплексируемостью протоколов; *наличие RPC*, именованных конвейеров и почтовых ящиков для поддержки распределенных приложений; *наличие дополнительных сетевых средств*, позволяющих строить сети в масштабах корпорации: дополнительные средства безопасности – централизованное администрирование, отказоустойчивость (UPS, зеркальные диски).

Windows NT унаследовала от своих предшественников редиректор и сервер, протокол верхнего уровня SMB и транспортный протокол NetBIOS (правда, с новым «наполнением» – NetBEUI). Как и в сети MS-NET

редиректор перенаправляет локальные запросы ввода-вывода на удаленный сервер, а сервер принимает и обрабатывает эти запросы.

Сетевой редиректор обеспечивает средства, необходимые одному компьютеру Windows NT для доступа к файлам и принтерам другого компьютера. Так как он поддерживает SMB-протокол, то он работает с существующими серверами MS-NET и LAN Manager, обеспечивая доступ к системам MS-DOS, Windows и OS/2 из Windows NT. Механизмы безопасности обеспечивают защиту данных Windows NT, разделяемых по сети, от несанкционированного доступа.

Редиректор имеет одну основную задачу: поддержку распределенной файловой системы, которая ведет себя подобно локальной файловой системе, хотя и работает через ненадежную среду (сеть). Когда связь отказывает, редиректор ответственен за восстановление соединения, если это возможно, или же за возврат кода ошибки, чтобы приложение смогло повторить операцию. Подобно другим драйверам файловой системы, редиректор должен поддерживать асинхронные операции ввода-вывода, если они вызываются. Когда пользовательский запрос является асинхронным, то редиректор должен вернуть управление немедленно, независимо от того, завершилась ли удаленная операция ввода-вывода или нет. При этом редиректор выполняется в контексте этой нити. Вызывающая нить должна продолжить свою работу, а редиректор должен ждать завершения запущенной операции. Есть два варианта решения этой проблемы: или редиректор сам создает новую нить, которая будет ждать, или он может передать эту работу уже готовой нити, существующей в системе. В Windows NT реализован второй вариант.

Редиректор отправляет и получает блоки SMB для выполнения своей работы. Протокол SMB является протоколом прикладного уровня, включающим сетевой уровень и уровень представления. SMB реализует: установление сессии; файловый сервис; сервис печати; сервис сообщений.

Интерфейс, в соответствии с которым редиректор посылает блоки SMB, называется *интерфейсом транспортных драйверов* (transport driver interface – TDI). Редиректор вызывает функции TDI для передачи блоков SMB различным транспортным драйверам, загруженным в Windows NT. Для вызова функций TDI редиректор должен открыть канал, называемый виртуальной связью (virtual circuit), к машине назначения, а затем послать SMB-сообщение через эту виртуальную связь. Редиректор создает только одну виртуальную связь для каждого сервера, с которым соединена система Windows NT, и мультиплексирует через нее запросы к этому серверу. Транспортный уровень определяет, каким образом реализовать виртуальную связь, и пересылает данные через сеть. Как и редиректор, сервер Windows NT на 100% совместим с существующими SMB-протоколами MS-NET и LAN Manager. Эта полная совместимость позволяет серверу обрабатывать запросы, исходящие не только от

систем Windows NT, но и от других систем, работающих с программным обеспечением LAN Manager. Как и редиректор, сервер выполнен в виде драйвера файловой системы.

Открытая архитектура сетевых средств Windows NT обеспечивает работу своих рабочих станций (и серверов) в гетерогенных сетях не только путем предоставления возможности динамически загружать и выгружать сетевые средства, но и путем непосредственного переключения с программных сетевых средств, ориентированных на взаимодействие с одним типом сетей, на программные средства для другого типа сетей в ходе работы системы. Windows NT поддерживает переключение программных средств на трех уровнях: 1) на уровне редиректоров – каждый редиректор предназначен для своего протокола (SMP, NCP, NFS, VINES); 2) на уровне драйверов транспортных протоколов, предоставляя для них и для редиректоров стандартный интерфейс TDI; 3) на уровне драйверов сетевых адаптеров – со стандартным интерфейсом NDIS 3.0.

Для доступа к другим типам сетей в Windows NT, помимо встроенного, могут загружаться дополнительные редиректоры. Специальные компоненты Windows NT решают, какой редиректор должен быть вызван для обслуживания запроса на удаленный ввод-вывод. За последние десятилетия получили распространение различные протоколы передачи информации по сети. И хотя Windows NT поддерживает не все эти протоколы, она, по крайней мере, разрешает включать их поддержку. После того, как сетевой запрос достигает редиректора, он должен быть передан в сеть. В традиционной системе каждый редиректор жестко связан с определенным транспортным протоколом. В Windows NT поставлена задача гибкого подключения того или иного транспортного протокола, в зависимости от типа транспорта, используемого в другой сети. Для этого во всех редиректорах нижний уровень должен быть написан в соответствии с определенными соглашениями, которые и определяют единый программный интерфейс, называемый *интерфейсом транспортных драйверов (TDI)*.

TDI позволяет редиректорам оставаться независимым от транспорта. Таким образом, одна версия редиректора может пользоваться любым транспортным механизмом. TDI обеспечивает набор функций, которые редиректоры могут использовать для пересылки любых типов данных с помощью транспортного уровня. TDI поддерживает как связи с установлением соединения (виртуальные связи), так и связи без установления соединения (датаграммные связи). Хотя LAN Manager использует связи с установлением соединений, Novell IPX является примером сети, которая использует связь без установления соединения. Microsoft изначально обеспечивает транспорты – NetBEUI (NetBIOS Extended User Interface), TCP/IP, IPX/SPX, DECnet и AppleTalk.

### 2.7.2. Средства BackOffice

Компания Microsoft объединяет под названием BackOffice набор своих серверов: сервер Windows NT Server, составляющий основу для построения остальных специализированных серверов: сервера баз данных Microsoft SQL Server, почтового сервера Microsoft Mail Server и сервера интегрированной службы обработки сообщений Microsoft Exchange, шлюз к SNA-сетям Microsoft SNA Server и сервер управления вычислительной системой Microsoft System Management Server. Все эти продукты хорошо работают вместе, образуя интегрированную и управляемую систему специализированных серверов офиса. В этой интегрированной среде можно построить приложения модели “клиент-сервер” практически любого масштаба, используя серверные возможности базы данных, почты, средств организации групповой работы.

**СЕРВЕР БАЗ ДАННЫХ SQL SERVER.** Этот сервер представляет собой систему управления реляционными базами данных высшего класса, построенную в архитектуре “клиент-сервер”. SQL Server достаточно легко интегрируется со всеми существующими на сей день клиентами – настольными компьютерами и системами на базе хостов. В перечень поддерживаемых клиентов входят: Windows 3.1, Windows for Workgroups 3.11, Windows NT Workstation, MS-DOS, OS/2 и Apple Macintosh. Для поддержки клиентов, работающих на UNIX и VMS, можно воспользоваться программным обеспечением Open Client Software компании Sybase. Сетевая поддержка (под **встроенной сетевой поддержкой понимают** функции совместного использования файлов, устройств и объектов) включает Microsoft Windows NT Server, Microsoft LAN Manager, Novell NetWare, сети с протоколами стека TCP/IP, IBM LAN Server, Banyan VINES, DEC PATHWORKS и AppleTalk. Все сети поддерживаются с помощью их родных протоколов.

Конечные пользователи могут получить доступ к данным, хранящимся на сервере, и оперативно составлять отчеты и проводить анализ данных с помощью таких средств, как Microsoft Access и Microsoft Excel. Для перехода от баз данных других форматов к SQL Server компания Microsoft разработала ряд полезных утилит миграции: Access Upsizing Tool для перехода от архитектуры приложений модели файл-сервер, Transfer Manager для переноса данных из баз данных Sybase или SQL Server, работающих в среде UNIX или OS/2, на платформу Windows NT.

Компания Microsoft предусмотрела также ряд средств для того, чтобы предлагаемое ей решение было открытым. Среди них технология ODBC, интерфейс DB-Library, шлюз ODS и язык ANSI SQL.

- *Технология ODBC (Open Database Connectivity)* – это открытый и независимый от производителя прикладной программный интерфейс (API) между клиентами и сервером. Технологию ODBC

поддерживают свыше 130 независимых производителей приложений, драйверов и сервисов баз данных, среди которых IBM, Lotus Development Corporation, Novell и Word Perfect.

- *DB-Library* – это родной интерфейс для Microsoft SQL Server, который поддерживается большим количеством коммерческих утилит и приложений. SQL Server также поддерживает и интерфейс Open Client компании Sybase.
- *ODS (Open Data Services)* – это API для разработки шлюзов, работающих на базе сервера Windows NT, для предоставления клиентам доступа к любым источникам информации.
- *Transact-SQL* – язык, который разработан специально для Microsoft SQL Server. Эта реализация SQL полностью совместима со стандартом SQL 1989 и дополнена возможностями для создания таких компонент базы данных, как триггеры, правила, хранимые процедуры, и некоторых других.

Надежность системы SQL Server определяется надежностью операционной системы Windows NT, а также собственными средствами – механизмом транзакций, системой автоматического восстановления после сбоев и отказов, компонентами целостности данных (правилами, хранимыми процедурами и триггерами). Производительность и масштабируемость обеспечиваются за счет полного использования широких возможностей в этих аспектах сервера Windows NT Server. SQL Server использует средства создания и диспетчирования нитей, управления их приоритетами, средства безопасности, управления событиями и их мониторинга Windows NT, исключая тем самым ненужное дублирование кода в операционной системе и СУБД. SQL Server не создает для каждого пользователя отдельного процесса, а работает как единый процесс, создающий для каждого пользовательского соединения отдельную нить. На SMP платформах каждая нить назначается на свободный или малозагруженный центральный процессор, обеспечивая динамическую балансировку загрузки.

Администрирование SQL Server обеспечивается за счет поставляемых утилит с графическим интерфейсом, предназначенных для работы под управлением Windows 3.x, Windows 9x или Windows NT. Эти утилиты поставляются в 32-битном и 16-битном вариантах. С помощью утилит администрирования можно управлять несколькими SQL серверами в сети. SQL Server поддерживает опцию интегрированной безопасности, которая обеспечивает один логический вход как в сеть, так и в сервер баз данных. При этом доступ к SQL серверу управляется привилегиями, которые устанавливаются для пользователей и групп пользователей в Windows NT. Специальная компонента, называемая SQL Monitor, позволяет составить и отработать расписание автоматического копирования данных из базы на устройства резервного копирования, такие как стриммеры.

Перспективы развития компания Microsoft связывает с версией SQL Server 6.0. Эта версия предназначена для крупных распределенных корпоративных систем и отличается следующими особенностями:

- Распределенные данные и приложения будут поддерживаться за счет встроенной системы репликации с удобными графическими средствами управления.
- Трехуровневая архитектура администрирования, основанная на технологии OLE, будет обеспечивать централизованное администрирование распределенными серверами.
- Объектная ориентация на основе технологии OLE предназначена для превращения SQL сервера из пассивного внешнего источника данных в активного участника пользовательских приложений. SQL Server 6.0 будет поддерживать богатый интерфейс OLE Automation для связей с настольными приложениями с помощью языка Visual Basic for Applications и интерфейса MAPI. Например, используя OLE, он может отослать пользователям результаты запроса по почте как встроенные объекты электронной таблицы Microsoft Excel.
- Повышение производительности до 20 Гбайт в час за счет новой методики параллельного архивирования и технологии компрессирования данных.

**ШЛЮЗ SNA SERVER.** В традиционной среде мейнфреймов вся обработка данных осуществляется на хост-машинах, причем мейнфреймы IBM традиционно считаются надежными средствами централизованного управления и администрирования такого рода обработки. Однако терминалы для доступа к мейнфреймам обычно не имеют средств пользовательского графического интерфейса, что усложняет работу пользователей, а также не могут производить обработку данных, как это делают персональные компьютеры. Кроме того, мейнфреймам недостает средств, необходимых для быстрой разработки приложений.

С другой стороны, настольные приложения, такие как Microsoft Access и Microsoft Excel, обеспечивают простоту использования данных, но не обладают достаточной мощностью для поддержки ответственных корпоративных данных. Чтобы получить преимущества от использования всех корпоративных данных, необходимо объединить данные, используемые в настольных компьютерах, с данными, хранящимися на серверах сетей и на мейнфреймах. Для западных пользователей эта проблема особенно актуальна, так как там около 80% всех компьютерных данных хранятся на мейнфреймах и других хостах.

Одним из средств объединения локальных сетей с мейнфреймами является Microsoft SNA Server for Windows NT, который обеспечивает для пользователей локальных сетей доступ к мейнфреймам и мини-компьютерам фирмы IBM. По сравнению с прямым подключением

персональных компьютеров к мейнфрейму, использование шлюза SNA Server экономит производительность как мейнфрейма, так и персоналок, обеспечивает централизованное управление взаимодействием, защищает корпоративные данные на уровне безопасности C2, обеспечивает высокую готовность доступа к мейнфрейму за счет средств отказоустойчивости и архивирования данных. SNA Server обеспечивает:

- Соединение со всеми популярными мейнфреймами архитектуры SNA (например, 3090 или ES/9000) и SNA-мини-компьютеров (семейства AS/400).
- Использование всех типов SNA-каналов: SDLC, X.25, 802.2.
- Взаимодействие с серверами локальной сети, работающими под управлением операционных систем: Microsoft, NetWare, Banyan VINES, AppleTalk.
- Взаимодействие с клиентами сети, работающими под управлением: MS-DOS, Windows 3.x, Windows NT Workstation, Windows NT Server и Macintosh (UNIX через дополнительный шлюз TN3270).
- Взаимодействие с клиентами через мосты, маршрутизаторы или сервер удаленного доступа RAS для Windows NT.

Шлюз SNA Server может использоваться в различных конфигурациях – один шлюз для доступа к одному хосту, один шлюз для доступа к нескольким хостам, много шлюзов для доступа к одному хосту, много шлюзов для доступа к нескольким хостам. Шлюз поддерживает до 250 одновременных соединений в любых комбинациях восходящих соединений (с хостом), равноправных или нисходящих соединений. Шлюз поддерживает до 2 000 пользователей, а в одном домене Windows NT может быть до 50 шлюзов SNA Server. Шлюз использует ту учетную пользовательскую информацию, которая хранится на сервере Windows NT, а для доступа к хостам пользователям шлюза должны быть предоставлены специальные права. Шлюз можно администрировать и с хоста с помощью системы NetView.

**ПОЧТОВЫЕ СИСТЕМЫ MICROSOFT MAIL И СИСТЕМА КОЛЛЕКТИВНОЙ РАБОТЫ MICROSOFT EXCHANGE.** Компания Microsoft предлагает в настоящее время три системы обработки сообщений – Microsoft Mail Server 3.2, Microsoft Mail Server 3.5 и Microsoft Exchange. Почтовая система Microsoft Mail 3.2 выпускается достаточно давно и состоит из почтового агента передачи сообщений (Message Transfer Agent, MTA), работающего на выделенном персональном компьютере под управлением MS-DOS или OS/2, и почтовых клиентов, которые могут работать в локальной сети под управлением DOS, Windows, OS/2 или Macintosh. Кроме этого, на любом невыделенном компьютере располагается база данных почтовой системы – почтовое отделение (Post Office, PO). В почтовую систему входят также шлюзы к другим типам почтовых систем, в том числе к системам, основанным на стандарте

X.400, системам обмена сообщениями мейнфреймов PROFS и SNADS, почтовой системе SMTP сетей TCP/IP, системе обмена сообщениями MHS фирмы Novell и некоторым другим. Все эти шлюзы работают на выделенных компьютерах под управлением DOS.

Недавно компания Microsoft объявила о выпуске новой версии почтовой системы Microsoft Mail Server 3.5, включающей новую версию многозадачной программы-агента передачи сообщений (Multitasking Message Transfer Agent, MMTA), которая работает в среде ОС Windows NT Server. Таким образом, старый вариант MMTA переводится из OS/2 в родную среду Windows NT. Кроме этого, новый пакет включает новые административные утилиты, позволяющие усовершенствовать управление почтовыми ящиками пользователей, личными адресными книгами и глобальными списками адресов. Интеграция Microsoft Mail с Windows NT Server является частью стратегии Microsoft по обеспечению возможности функционирования в рамках одной организации двух систем – Microsoft Mail и Microsoft Exchange Server. Exchange Server предназначен не только для поддержки почтового обмена сообщениями, но и содержит надстройки для организации работы в группе. Система Microsoft Mail Server 3.5 полностью совместима с клиентским ПО Microsoft Exchange, которое вошло в комплект поставки Windows 95. Для использования этого ПО совместно с почтовыми отделениями Microsoft Mail не требуется дополнительных лицензий.

Система Microsoft Exchange интегрирует электронную почту, планирование работы пользователей, электронные формы, средства разделение документов и содержит некоторые приложения, например, отслеживание активности покупателей. По сути, это объединение электронной почты и системы разделения информации на основе технологии клиент-сервер. Система состоит из сервера Microsoft Exchange Server и клиентов для различных популярных операционных систем. Система организует для своих клиентов доступ к различным источникам информации, например, к серверам баз данных, на основе механизма обмена сообщениями в технологии store-and-forward, присущей почтовым системам. Сервер поддерживает не только локальные, но и глобальные связи, а также обеспечивает шифрацию передаваемых сообщений по алгоритму LSA.

## **ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ**

- 1. Составьте логическую схему базы знаний по теме юниты.*

## ТРЕНИНГ УМЕНИЙ

### Пример выполнения упражнения на умение № 1

#### Задание

Выбрать поля number и dolgnost из БД otdel, dolgnost и oklad из БД vedomost со значением oklad, большим 1000 руб., при условии совпадения поля dolgnost в обеих таблицах.

#### Решение

№ п/п	Алгоритм	Конкретное соответствие данной ситуации предложенному алгоритму
1	Выбрать БД	Из исходных данных следует, что необходимо выбрать БД otdel и vedomost. SQL оператор выглядит так: FROM otdel, vedomost
2	Определить поля для выборки данных	Из исходных данных следует, что необходимо выбрать поля otdel.number, otdel.dolgnost, vedomost.dolgnost и vedomost.oklad. SQL оператор выглядит так: SELECT otdel.number, otdel.dolgnost, vedomost.dolgnost, vedomost.oklad
3	Определить условия выборки данных	Из исходных данных следуют условия выборки данных: vedomost.oklad >= 1000 и otdel.dolgnost = vedomost.dolgnost. SQL оператор выглядит так: WHERE (vedomost.oklad >= 1000) AND (otdel.dolgnost = vedomost.dolgnost)
4	Полностью написать SQL запрос.	SELECT otdel.number, otdel.dolgnost, vedomost.dolgnost, vedomost.oklad FROM otdel, vedomost WHERE (vedomost.oklad >= 1000) AND (otdel.dolgnost = vedomost.dolgnost)

*Решите самостоятельно следующие задания:*

#### Задание 1

Выбрать поля number и dolgnost из БД otdel, dolgnost и oklad из БД vedomost со значением oklad, лежащим в пределах 1000-2000 руб., при условии совпадения поля dolgnost в обеих таблицах.

### *Задание 2*

Выбрать из БД `otdel` и `vedomost` все поля со значением `vedomost.oklad`, меньшим 1000 руб., при условии совпадения поля `dolgnost` в обеих таблицах.

### *Задание 3*

Выбрать все поля из БД `otdel1`, `otdel2` и `otdel3` при условии совпадения поля `dolgnost` во всех таблицах.

## Пример выполнения упражнения на умение № 2

### Задание

Из браузера считан полный адрес узла – <http://polyn.net.kiae.su/polyn/mamfest.html>. Определите имя домена.

### Решение

№ п/п	Алгоритм	Конкретное соответствие данной ситуации предложенному алгоритму
1	Считать из браузера полный адрес узла	Из исходных данных известен полный адрес узла – <a href="http://polyn.net.kiae.su/polyn/mamfest.html">http://polyn.net.kiae.su/polyn/mamfest.html</a>
2	Определить используемый протокол, имя домена и адрес узла в домене	После анализа полного адреса получаем: используемый протокол – http имя домена - polyn.net.kiae.su адрес узла в домене – polyn/mamfest.html

*Решите самостоятельно следующие задания:*

### Задание 1

По адресу <http://144.206.130.137:8080/altai/index.html> определите адрес узла в домене и используемый протокол.

### Задание 2

По адресу <http://144.206.130.137:8080/altai/data/index.html/#tip> определите адрес узла в домене и имя домена.

### Задание 3

По адресу gopher://gopher.kiae.su:70:7/kuku определите имя домена и используемый протокол.

### Задание 4

По адресу ftp://polyn.net.kiae.su/pub/0index.txt определите адрес узла в домене, имя домена и используемый протокол.

## Пример выполнения упражнения на умение № 3

### Задание

Установить запрет на запись в каталогах User, Private и Data диска C: для группы пользователей "Гость".

### Решение

№ п/п	Алгоритм	Конкретное соответствие данной ситуации предложенному алгоритму
1	Выбрать (выделить) требуемые диски, тома, каталоги и/или файлы	Выделяем каталоги User, Private и Data диска C:
2	Открыть окно «Свойства» выбранных объектов	Нажав правую кнопку мыши, выберем из всплывающего меню пункт «свойства», при этом открывается диалоговое окно «Свойства».
3	Выбрать закладку «Безопасность»	Наведя указатель мыши на закладку «Безопасность» и нажав на неё, открываем эту закладку

№ п/п	Алгоритм	Конкретное соответствие данной ситуации предложенному алгоритму
4	В панели «Разрешения» нажать кнопку «Разрешения»	В панели «Разрешения» нажать кнопку «Разрешения», при этом открывается новое диалоговое окно – «Разрешения: Каталог»
5	Установить права доступа для нужных групп пользователей	Если в разделе «Имя» отсутствует группа «Гость», то добавить её, нажав кнопку «Добавить». Выделить группу. В разделе «Тип доступа» установить значение «Чтение» или «Просмотр». Последовательно нажимать кнопку «ОК» до закрытия всех открытых вами диалоговых окон

*Решите самостоятельно следующие задания:*

#### *Задание 1*

Установить запрет доступа к каталогам User, Private и Data диска C: для группы пользователей «Гость» и «Репликаторы».

#### *Задание 2*

Снять запрет доступа к каталогам User, Private и Data диска C: для группы пользователей «Репликаторы».

### **Задание 3**

Запретить внесение изменений в файлы с расширением \*.doc диска C: для группы пользователей "Гость".

### **Пример выполнения упражнения на умение № 4**

#### **Задание**

Включить запись доступа (аудит) для пользователей, входящих в группу "Гость".

#### **Решение**

№ п/п	Алгоритм	Конкретное соответствие данной ситуации предложенному алгоритму
1	Выбрать (выделить) требуемые диски, тома, каталоги и/или файлы	Выделяем каталоги User, Private и Data диска C:
2	Открыть окно «Свойства» выбранных объектов	Нажав правую кнопку мыши, выберем из всплывающего меню пункт «свойства»; при этом открывается диалоговое окно «Свойства»

№ п/п	Алгоритм	Конкретное соответствие данной ситуации предложенному алгоритму
3	Выбрать за- кладку «Без о- пасность»	Наведя указатель мыши на закладку «Без о- пасность» и нажав на неё, открываем эту з а- кладку
4	В панели «А у- дит» нажать кнопку «Аудит»	В панели «Аудит» нажать кнопку «Аудит», при этом открывается новое диалоговое окно – «Аудит: Каталог»
5	Установить па- раметры аудита для нужных групп пользо- вателей	Если в разделе «Имя» отсутствует группа «Гость», то добавить её, нажав кнопку «Д о- бавить». Выделить гру ппу. В разделе «Аудит событий» установить г а- лочки для всех типов аудита. Последов а- тельно нажимать кнопку «ОК» до закрытия всех открытых вами диалог овых окон

*Решите самостоятельно следующие задания:*

#### *Задание 1*

Установить аудит за всеми событиями в каталогах User, Private и Data диска C: для группы пользователей “Гость”.

#### *Задание 2*

Снять полный аудит и установить аудит на запись к каталогам User, Private и Data диска C: для групп пользователей “Гость” и “Репликаторы” (соответственно).

### *Задание 3*

Снять аудит на запись и установить аудит на чтение к каталогам User, Private и Data диска C: для группы пользователей “Репликаторы”.

## **СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ**

ЮНИТА 3

### **СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ СЕТЕЙ ЭВМ**

Редактор Л.С. Лебедева

Оператор компьютерной верстки Д.В. Федотов

---

Изд. лиц. ЛР № 071765 от 07.12.1998

Сдано в печать

НОУ “Современный Гуманитарный Институт”

Тираж

Заказ

---

Современный Гуманитарный Университет