
ПРОФЕССИОНАЛЬНОЕ
ОБРАЗОВАНИЕ

ФОРУМ

И.И. ПОПОВ
Т.Л. ПАРТЫКА

# ОПЕРАЦИОННЫЕ СИСТЕМЫ, СРЕДЫ И ОБОЛОЧКИ

Т. Л. Партыка, И. И. Попов

# ОПЕРАЦИОННЫЕ СИСТЕМЫ, СРЕДЫ И ОБОЛОЧКИ

*Допущено Министерством образования Российской Федерации в качестве учебного пособия для студентов учреждений среднего профессионального образования, обучающихся по специальностям информатики и вычислительной техники*

Москва
ФОРУМ - ИНФРА-М
2003

УДК 002.56(075.32)
ББК 32.973я723
П57

*Рецензенты:*
доцент кафедры Проектирование автоматизированных информационных систем РЭА им. Г. В. Плеханова, к.ф.-м.н. *Б. В. Евтеев',*
директор Института компьютерных технологий МЭСИ, зав. кафедрой Общая теория систем и системного анализа, д.э.н., профессор *А. А. Емельянов*;
председатели предметных (цикловых) комиссий Математического колледжа *В. П. Агальцов, В. А. Макунин*

**Партыка Т. Л., Попов И. И.**
П57 Операционные системы, среды и оболочки: Учебное пособие. - М.: ФОРУМ: ИНФРА-М, 2003. - 400 с.: ил. - (Серия «Профессиональное образование»).

ISBN 5-8199-0072-3 (ФОРУМ)
ISBN 5-16-001355-5 (ИНФРА-М)

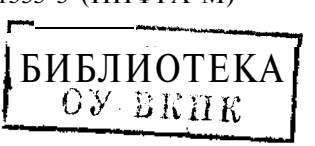
В учебном пособии рассматриваются общие принципы организации, состав, структура операционных систем и их оболочек, а также ряд конкретных систем. Значительное внимание уделяется проблемам управления информацией, процессами в ЭВМ и связи с оператором в рамках различных интерфейсов. В качестве примеров конкретных систем рассматриваются как ОС персональных компьютеров — MS-DOS, Windows 3.x, 95/98/ME, NT/2000, так и ОС для многопользовательских ЭВМ — 08 360/370/375, RSX, Unix, LINUX. Рассмотрен ряд оболочек, расширяющих свойства ОС ЭВМ, как с текстовым, так и с графическим интерфейсом.

Предназначено для учащихся техникумов, колледжей, а также студентов вузов.

УДК 002.56(075.32)
ББК 32.973я723

ISBN 5-8199-0072-3 (ФОРУМ)
ISBN 5-16-001355-5 (ИНФРА-М)





# Предисловие

Современный компьютер — это универсальное, многофункциональное, электронное автоматическое устройство для работы с информацией. Компьютеры в современном обществе взяли на себя значительную часть работ, связанных с обработкой данных. По историческим меркам компьютерные технологии обработки информации еще очень молоды и находятся в самом начале своего развития, они сегодня преобразуют или вытесняют старые традиционные технологии обработки информации.

Программное обеспечение (ПО) компьютера можно разделить на общесистемное и прикладное программное обеспечение.

*Операционная система (ОС),* являясь основой общесистемного ПО, обеспечивает функционирование и взаимосвязь всех компонентов компьютера и предоставляет пользователю доступ к его аппаратным возможностям.

*Прикладное программное обеспечение* можно в свою очередь разделить на две группы программ: средства разработки и приложения.

*Средства разработки* — это инструменты программиста. Традиционными средствами разработки являются системы (среды) программирования (СП), использующие алгоритмические языки программирования (ЯП). Основой систем программирования являются трансляторы, т. е. программы, обеспечивающие перевод исходного текста программы (на ЯП) на машинный язык (объектный код), которые бывают двух типов — интерпретаторы и компиляторы.

*Приложения* — это программные продукты, предназначенные Для решения задач в какой-либо конкретной предметной области. Многообразие приложений соответствует спектру задач, которые могут быть решены алгортимически.

Ранние ЭВМ не предусматривали ОС, поэтому процессы запуска/остановки программы, присоединения внешних носителей Управлялись вручную или из прикладной программы. В середине 60-х годов ряд ведущих фирм-производителей ЭВМ — IBM (International Business Machine — США), ICL (International Computer Limited — Великобритания), СП (Compagne Internationale роиг Informatique — Франция) практически одновременно приступили к выпуску моделей машин (соответственно — IBM 360, System 4, Iris 80), оснащенных операционными системами (operating system).

Наиболее совершенной и конкурентноспособной оказалась система OS/360 (IBM), в которой были заложены практически все основные черты ОС, позволяющие превратить ЭВМ в «автоматизированную фабрику» обработки информации при минимальном участии человека. OS/360 и другие современные ей системы были ориентированы на обработку потока заданий (или пакетную обработку — batch processing), при которой пользователь не может вмешаться в ход выполняемой задачи, оперативно просмотреть промежуточные данные, т. е. оторван от машины.

Появление и широкое распространение видеотерминалов привели к возможности предоставить пользователю интерактивный диалоговый доступ к вычислительному процессу, которым он занимается. В О5/360 фирмой и пользователями были внесены дополнения — появились системы TSO (Time Sharing Option), CICS (Сизlотег Information Control System). Известен ряд удачных отечественных разработок — PRIMUS, FOCUS. Появившиеся в последующий период ОС ориентировались исключительно на интерактивную работу пользователей — RSX, VMS и пр.

В настоящее время наиболее распространенной является интерактивная ОС UNIX, версии которой разработаны практически для всех моделей ЭВМ. Для IBM PC-совместимых ЭВМ (ПЭВМ) в свое время была разработана UNIX-подобная система MS/DOS (фирма MicroSoft). Следует согласиться с остроумным замечанием Питера Нортона о том, что «MS-DOS — это UNIX для дошкольников, UNIX — это MS-DOS для лиц с высшим образованием».

ОС является первичной программной оболочкой для всякой ЭВМ; без ОС ЭВМ становится неодушевленным предметом. При включении электропитания ЭВМ автоматически осуществляется считывание с магнитного носителя, запись в оперативную память и запуск резидентных программ ОС, или загрузка ОС (loading). В некоторых системах процесс загрузки прерывается для запроса у оператора адреса (номера), внешнего устройства, на котором размещены программы ОС (резидентного устройства). При включении ПЭВМ поиск устройства с ОС осуществляется автоматически.

Резидентное устройство (точнее — НМД) часто называют *bootable*, а процесс загрузки — *boot* («обувать»), что хорошо иллюстрирует, во-первых, «голый» статус компьютера без ОС, во-вторых, возможность «одеть» компьютер в разные ОС, при этом «образ машины» может измениться до неузнаваемости. Это давно и хорошо известно опытным пользователям больших компьютеров, а в последнее время стало «достоянием широких масс» в связи с тем, что последние моде-

ли ПЭВМ уже в состоянии работать с несколькими различными ОС - M5 DOS, ОЗ/2, Windows 95/98/ME/NT/2000UNIX и пр.

ОС предназначены для выполнения следующих основных (тесно взаимосвязанных) функций:

- управлениеданными;
- управление задачами;
- связь с внешней средой.

В различных ОС эти функции реализуются в различных масштабах и с помощью разных технических, программных, информационных методов.

Структурно операционная система представляет собой совокупность программ, управляющих ходом работы вычислительной машины, идентифицирующих прикладные программы и данные и осуществляющих связь между машиной и оператором. Операционная система повышает производительность вычислительного комплекса за счет гибкой организации прохождения потока задач через машину, равномерной загрузки оборудования, оптимального использования всех ресурсов ЭВМ, стандартной организации хранения в машине больших массивов данных при наличии разнообразных способов доступа к ним.

В настоящем учебном пособии рассматриваются общие принципы организации, состав, структура операционных систем и их оболочек, а также ряд конкретных систем.

**В первой главе** рассматриваются основные принципы организации и функционирования операционных систем их состав и структура. Рассматриваются функции управления данными, включая планирование размещения данных и оперативное управление их прохождением через систему в процессе решения задач. С данной функцией тесно связано управление заданиями (процессами, задачами), дисциплины обслуживания процессов и подпроцессов, управление очередями, оптимизация использования памяти для размещения задач. Функция связи с оператором реализует совокупность интерфейсов как предназначенных для текущего управления вычислительным процессом (оператор ЭВМ), так и для конфигурирования и установки ОС и ее компонент (системный администратор) и для передачи данных в прикладную программу и их вывода из нее (прикладной пользователь).

**Во второй главе** рассматриваются операционные системы персональных компьютеров (однопользовательские, однопрограммные), как получившие наибольшее распространение (сотни миллионов экземпляров). Это прежде всего операционная система MS-DOS (на

примере версии 6.22), затем графические программные оболочки Windows 3.x, операционные системы Windows 95/98/ME, системы Windows NT/2000. Возможно, сюда следовало бы отнести также ОС Linux и версии UNIXдля ПЭВМ (AIX, XENIX), а также оболочку X Window, однако мы поместили эти вопросы в следующую главу. Это связано, скорее, с историческими и генеалогическими соображениями — рассматриваемые во второй главе системы, во-первых, являются продукцией Microsoft, а, во-вторых, их возможности росли и расширялись вместе с аппаратной платформой, на которую они разрабатывались, — *Intel-8086-80286-386-486-Pentium* и т. д. И хотя Windows NT/2000, как уверяют специалисты, ненамного уступает по сетевым и многопользовательским свойствам таким ОС, как UNIX/LINUX, надо помнить, что UNIX/LINUX пришли на ПЭВМ с больших вычислительных систем, а не наоборот.

**В третьей главе** рассматриваются многопользовательские многозадачные операционные системы в той исторической последовательности, в которой они были разработаны. Прежде всего, это ряд систем OS/360/370/375, являющихся классическим прототипом всех последующих разработок, затем операционные системы RSX (ОС РВ) и наиболее популярные сегодня среди системных администраторов мощных машин системы UNIX и LINUX. Основной чертой данных ОС является обеспечение работы систем в одном из следующих режимов:

- *системы с разделением времени,* в которых каждый участник как бы монопольно пользуется ресурсами ЭВМ, и основной задачей администраторов и разработчиков является защита данных от несанкционированного доступа и *взаимная изоляция* участников;
- *системы обеспечения групповых решении (СОГР)* — (Computer Supported Cooperative Work), groupware — ориентированные на прямо противоположную задачу — обеспечить взаимодействие пользователей в процессе принятия решений. СОГР сочетают коммуникационную, вычислительную технологии и технологию принятия решений для облегчения формулирования и решения неструктурированных проблем группой лиц.

**В четвертой главе** рассматриваются среды и оболочки операционных систем. Прежде всего, дается краткое описание программ расширения возможностей пакетных ОС (О5/360/370/375) — диалоговые мониторы ЕС ЭВМ, затем в исторической последовательности появления на сцене — монитор PCTOOLS для ПЭВМ, оболочка Norton Commander (NC), Norton Commander-подобные оболочки

для Windows (в том числе NC для Windows, Windows Commander, Far Manager), программная оболочка Dosshell. В целом, если строго придерживаться классификации оболочек на текстовые и графические, то средства NC для Windows и Windows Commander следовало бы рассматривать в ряду с Windows 3.x как графические, однако они помещены в 4-ю главу как прямые потомки NC, функционально сходные с Far Manager, который является принципиально текстовой оболочкой.

Учебное пособие базируется на материалах, накопленных авторами в процессе практической, исследовательской, а также преподавательской (МИФИ, МИСИ, МГУ, РГГУ, РЭА им. Г. В. Плеханова) деятельности. Авторы выражают благодарность коллегам, принявшим участие в обсуждении материала: А. Г. Романенко (РГГУ), К. И. Курбакову (РЭА им. Г. В. Плеханова), П. Б. Храмцову (РНИЦ «Курчатовский институт»), рецензентам, а также студентам РГГУ и РЭА им. Г. В. Плеханова за предоставленные иллюстративные материалы.

# Глава 1. Операционные системы ЭВМ. Основные принципы и понятия

Функционирование современных ЭВМ обеспечивается на паритетных началах аппаратными и программными средствами. *Программное обеспечение* выполняет функцию посредника между пользователями и ЭВМ, расширяет возможности аппаратуры вычислительной машины, являясь логическим ее продолжением. Использование развитого программного обеспечения позволяет увеличить производительность вычислительных систем, автоматизировать многочисленные рутинные информационные процессы в различных областях человеческой деятельности, повысить производительность труда разработчиков различных систем автоматизированной переработки информации, сократить общие сроки разработок и т. д.

Программное обеспечение можно разделить на системное и прикладное. *Системное программное обеспечение* представляет собой комплекс управляющих и обрабатывающих программ, описаний и инструкций, обеспечивающих функционирование вычислительной системы, а также разработку и исполнение программ пользователей. Состав системного программного обеспечения почти не зависит от характера решаемых задач пользователей.

Программы системного программного обеспечения различаются по функциональному назначению и характеру исполнения. Они делятся на испытательные программы, системы программирования (СП) и операционные системы (ОС).

На базе операционных систем строятся программные средства, расширяющие функции ОС, и пакеты общего назначения для решения различных научных, технических, экономических и других задач. Такие пакеты не входят в ОС и поставляются отдельно.

*Прикладное программное обеспечение* представляет собой совокупность программ решения конкретных задач из различных сфер применения ЭВМ. Специализированные комплексы программ решения конкретных задач вместе с сопровождающей документацией называют *пакетами прикладных программ* (ППП) или приложениями [1, 3].

Объем программного обеспечения современных вычислительных систем непрерывно возрастает, несмотря на то, что его стоимость остается довольно высокой даже при использовании промышленных методов разработки.

Особенно велика роль системного программного обеспечения, так как на его базе разрабатывается специальное программное обеспечение. Нередко доля стоимости системного программного обеспечения от общей стоимости вычислительной системы достигает 70 % и выше.

В исторической последовательности развития программных средств первыми появились узко ориентированные приложения *(«программа, предназначенная для вычисления числа* я *с точностью до 200-го знака», «программа, предназначенная для расчета и печати платежной ведомости»* и пр.), затем СП (ранние их версии назывались *системами автоматизации программирования)* и ОС.

## 1.1. Функции и состав операционных систем

***Функции ОС.*** *Операционная система* — это набор программ, обеспечивающий организацию вычислительного процесса на ЭВМ. Основные задачи ОС следующие:

- увеличение пропускной способности ЭВМ (за счет организации непрерывной обработки потока задач с автоматическим переходом от одной задачи к другой и эффективного распределения ресурсов ЭВМ по нескольким задачам);
- уменьшение времени реакции системы на запросы пользователей пользователями ответов от ЭВМ;
- упрощение работы разработчиков программных средств и сотрудников обслуживающего персонала ЭВМ (за счет предоставления им значительного количества языков программирования и разнообразных сервисных программ).

Операционные системы могут классифицироваться по следующим показателям:

- *количество пользователей:* однопользовательские ОС (M5-DOS, Windows) и многопользовательские ОС (VM, UNIX);
- *доступ:* пакетные (О5 360), интерактивные (Windows, UNIX), системы реального времени (QNX, Neutrino, RSX);
- *количество решаемых задач:* однозадачные ОС (MS-DOS) и многозадачные ОС (Windows, UNIX).

Операционная система предназначена для выполнения следующих основных (тесно взаимосвязанных) функций:

- управление данными;
- управление задачами (заданиями, процессами);
- связь с человеком-оператором.

В различных ОС эти функции реализуются в различных масштабах и с помощью разных технических, программных, информационных методов и средств.

Структурно операционная система представляет собой совокупность программ, управляющих ходом работы вычислительной машины, идентифицирующих прикладные программы и данные и осуществляющих связь между машиной и оператором. Операционная система повышает производительность вычислительного комплекса за счет гибкой организации прохождения потока задач через машину, равномерной загрузки оборудования, оптимального использования всех ресурсов ЭВМ, стандартной организации хранения в машине больших массивов данных при наличии разнообразных способов доступа к ним.

В состав системного программного обеспечения входят также сервисные программы, которые предназначены для проверки исправности блоков ЭВМ, обнаружения и локализации отказов устройств и устранения их влияния на работу системы в целом.

Системное программное обеспечение ЭВМ предназначено для осуществления адаптируемости программ пользователей к изменениям состава ресурсов ЭВМ. Высокая производительность вычисли-


Рис. 1.1. Основные функции операционных систем (ОС)

тельной системы обеспечивается операционной системой благодаря применению режимов пакетной обработки и мультипрограммного и наличию специальных программных средств для выполнения трудоемких операций ввода-вывода информации. Высокая производительность труда программиста достигается за счет предоставления ему большого числа языков программирования; специальных библиотек программ; удобных средств ввода-вывода, средств отладки программ и оформления заданий.

К числу наиболее известных первых управляющих программ относятся комплексы SAGE, SABRE, MERCURY, реализованные на ЭВМ второго поколения. Для ЭВМ IBM/360 были разработаны операционные системы, обеспечивающие пакетную технологию обработки данных и работу в реальном масштабе времени, а также реатазацию многомашинных и мультипроцессорных комплексов.

Первая функционально полная ОС — О5/360 была предложена фирмой IBM в качестве оболочки ЭВМ IBM/360. Разработка и внедрение ОС позволили разграничить функции операторов, администраторов, программистов, пользователей, а также существенно (в десятки и сотни раз) повысить производительность ЭВМ и степень загрузки технических средств. Версии OS/360/370/375 — MPT (мультипрограммирование с фиксированным количеством задач), MVT (с переменным количеством задач), SVS (система с виртуальной памятью), SVM (система виртуальных машин) — последовательно сменяли друг друга и во многом определили современные представления о роли ОС в общей иерархии систем управления данными и задачами при обработке данных на ЭВМ.

Ранние версии OS/360 были ориентированы на пакетную (batch processing) обработку информации — входной поток заданий (на МЛ, МД или перфокартах) подготавливался заранее и поступал на обработку в непрерывном режиме. В дальнейшем возникли расширения О3/360/375, допускающие диалоговую обработку данных с терминалов пользователя, последняя из версий (OS SVM) фактически предоставляла в распоряжение пользователя «виртуальную персональную ЭВМ» с полной мощностью вычислительной установки IBM/360/375. Операционные системы других семейств (поколений), например RSX (для PDP/11 DEC) или Unix с самого возникновения ориентировались на интерактивное взаимодействие с пользователями.

Одно из основных требований к разработке программного обеспечения ЭВМ — *модульность.* Модульная структура программ и программных комплексов облегчает организацию работы больших коллективов программистов по созданию программного обеспече-

ния. Другое важное требование к программному обеспечению — *возможность развития* программной системы. Выполнению этого требования способствует модульная организация программ. Существенным является требование *простоты* освоения, поддержания, эксплуатации и совершенствования возможностей программного обеспечения. Это позволяет обходиться небольшим числом специалистов, обслуживающих принятое к эксплуатации программное обеспечение.

Система программного обеспечения предназначена для эксплуатации многочисленными группами пользователей в различных организациях и предприятиях, поэтому она должна обладать свойствами *гибкости, адаптируемости*. Эти требования обеспечиваются применением принципов открытости, машинной независимости обрабатывающих программ, унификации использования периферийного оборудования и т. д. По возможности должна достигаться *совместимость* программного обеспечения различных ЭВМ и систем обработки данных. Как правило, совместимость программ обеспечивается в рамках ЭВМ одной аппаратной платформы. Программная совместимость для различных семейств ЭВМ достигается на уровне языков программирования.

Требование *минимальности вмешательства человека* в процесс обработки информации на ЭВМ удовлетворяется путем автоматизации различных этапов вычислительного процесса. В частности, автоматическое распределение ресурсов повышает эффективность использования вычислительной системы. Программное обеспечение должно удовлетворять также требованиям *параметрической универсальности, функциональной избыточности* (наличия в системе нескольких программ, реализующих одну и ту же функцию), *функциональной избирательности* (возможность конфигурирования программного обеспечения в соответствии с потребностями и возможностями конкретного пользователя).

Техническая документация на программные средства, являющаяся одним из важнейших элементов программного обеспечения, должна оформляться по единым стандартам. К технической документации относятся графические и текстовые документы, определяющие назначение, состав и структуру созданного программного средства. В них должны содержаться сведения, необходимые для тестирования, приемки, обучения пользователей, эксплуатации и наращивания возможностей программ. Выпуск документации является трудоемким процессом, поэтому желательно его автоматизировать. Документация на программное обеспечение должна удовлетворять требованиям *единства* терминологии, номенклатуры и наименова-

ния документов, единой системы обозначений в документах, идентичности документов независимо от места их разработки. Кроме того, должны соблюдаться единые правила внесения изменений, учета и хранения документации. Детальность описания отдельных модулей программного обеспечения должна соответствовать уровню подготовки потенциальных пользователей (системного программиста, программиста-пользователя, оператора и т. д.).

***Программы ОС.*** Программы ОС постоянно (резидентно) занимают в оперативной памяти объем, установленный при конфигурировании системы. Остальные части операционной системы по мере необходимости вызываются из внешней памяти на МД.

Операционная система обеспечивает осуществление в вычислительной системе следующих процессов:

- обработки задач;
- работы системы в режиме диалога и квантования времени;
- работы системы в реальном масштабе времени в составе многопроцессорных и многомашинных комплексов;
- связи оператора с системой;
- протоколирования хода выполнения вычислительных работ;
- обработки данных, поступающих по каналам связи;
- функционирования устройств ввода-вывода;
- использования широкого набора средств отладки и тестирования программ;
- планирования прохождения задач в соответствии с их приоритетами;
- ведения учета и контроля за использованием данных, программ и ресурсов ЭВМ.

Основные компоненты операционных систем — управляющие и обрабатывающие программы. Управляющие программы управляют работой вычислительной системы, обеспечивая в первую очередь автоматическую смену заданий для поддержания непрерывного режима работы ЭВМ при переходе от одной программы к другой без вмешательства оператора.

*Управляющая программа* определяет порядок выполнения обрабатывающих программ и обеспечивает необходимым набором услуг Для их выполнения. Основные функции управляющей программы: последовательное или приоритетное выполнение каждой работы (Управление задачами); хранение, поиск и обслуживание данных независимо от их организации и способа хранения (управление данными).

*Программы управления задачами* считывают входные потоки задач, обрабатывают их в зависимости от приоритета, инициирует од-

новременное выполнение нескольких заданий; вызывают процедуры; ведут системный журнал.

*Программы управления данными* обеспечивают способы организации, идентификации, хранения, каталогизации и выборки обрабатываемых данных. Эти программы управляют вводом-выводом данных с различной организацией, объединением записей в блоки и разделением блоков на записи, обработкой меток томов и наборов данных.

*Программы управления восстановлением* после сбоя обрабатывают прерывания от систем контроля, регистрируют сбои в процессоре и внешних устройствах, формируют записи о сбое в журнале, анализируют возможность завершения затронутой сбоем задачи и переводят систему в состояние ожидания, если завершение задачи невозможно.

*Конфигурация системы.* Прикладная программа в операционных системах может получить от ОС в процессе своей работы характеристики конкретной реализации системы, в среде которой она функционирует: имя, версию и редакцию операционной системы, тип и технические характеристики компьютера. В ОС обычно имеются средства локализации, позволяющие настроить систему на конкретное национальное (местное) представление данных: представление десятичных дробей, денежных величин, даты и времени.

Операционные системы предоставляют программе пользователя возможность узнать текущие дату и время. За начало отсчета, например, в MS-DOS принята дата 1 января 1980 г. О часов 0 минут О секунд по Гринвичу, в UNIX 1 января 1970 г. Системы предоставляют возможность измерения временных интервалов короче 1 секунды с помощью специальных системных вызовов. ОС может переводить дату и время из внутреннего числового представления в символьное (пригодное к выводу, например, на терминал); местное время во время по Гринвичу и наоборот; предоставлять информацию о часовом поясе, летнем и зимнем времени.

## 1.2. Управление данными в операционных системах

Управление данными включает следующие компоненты:

- долговременное планирование — организацию размещения данных на внешних носителях, их выборку и предоставление пользовательским программам;

- оперативное управление — распределение оперативной памяти под программы и данные, реализацию обмена данными между оперативной и внешней памятью;
- управление внешними устройствами ввода-вывода и размещения данных.

***Внешние устройства ЭВМ.*** Несколько слов о тех устройствах, которые отвечают за размещение и ввод-вывод данных. Здесь мы ограничимся рассмотрением спектра устройств IBM PC-совместимых ПЭВМ. Прежде всего, необходимо отметить, что типовая конфигурация внешних устройств (ВУ) в данном случае включает: *терминал/консоль (экран и клавиатура), накопители на магнитных дисках (НМД) и принтер.* Эти устройства будут подробно рассмотрены ниже. Пока же вкратце охарактеризуем принципы функционирования ВУ и их перечень в целом. Прежде всего, контроллеры ВУ ПЭВМ представляют собой стандартного размера электронные платы (*интерфейсные карты, адаптеры* и пр.), которые практически полностью взаимозаменяемы, что позволяет укомплектовать экземпляр ПЭВМ любым желаемым набором устройств (но не более 4—8, в зависимости от класса машины).

*Контроллеры типовых устройств,* как правило, являются несъемными и размещаются на системной плате (motherboard) ПЭВМ. И далее, практически все устройства требуют для своей работы программной поддержки (как минимум — запуска и непрерывного функционирования специальных программ — *драйверов устройств,* или более сложных *прикладных программ).* В более мощных системах (UNIX, Windows NT) подобные программы входят в состав *операционной системы* и обязательно присутствуют в машине. В случае, например, MS-DOS — это необязательные компоненты, которые необходимо приобретать и устанавливать дополнительно.

***Накопители на магнитных носителях, файлы, циклы обработки.*** Накопители данного типа являются основной средой хранения информации в ЭВМ и разделяются на накопители на магнитных лентах (НМЛ) и магнитных дисках (НМД). Появлявшиеся в различное время магнитные барабаны и магнитные карты особого распространения не получили. В различные временные периоды НМД и НМЛ по очереди доминировали в качестве основного вида накопителя. В настоящее время устоялось следующее представление: НМД используются для оперативного (во время решения задач) хранения информации, НМЛ — для резервного (архивного) хранения (стримеры).

В состав внешних запоминающих устройств ЭВМ входят накопители на магнитных лентах (НМЛ), магнитных дисках (НМД),

кассетных магнитных лентах (НКМЛ), дискетах или гибких магнитных дисках (НГМД) и т. д. Наибольшее распространение получили разработанные ранее других накопители на магнитных лентах и магнитныхдисках.

*Файл* (набор данных на внешнем носителе) рассматривается как совокупность записей одинаковой структуры (обычно, хотя и необязательно — фиксированной длины), каждая из которых представляет собой набор (агрегат) разнородных данных (в языках программирования — PL/1, Pascal, Си за подобными объектами так и закрепилось название *структура* — *structure*).

Понятие «файл» появилось впервые в операционной системе О5/360 фирмы IBM, причем в ранних версиях системы «настоящим файлом» считался только перфокарточный массив (file=картотека), данные на МД и МЛ обозначались как DS (Data Set — набор данных). В последующих ОС (RSX, UNIX, MS-DOS) файлами становятся именованные организованные наборы данных на любых носителях и устройствах, за сохранность и обновляемость которых (а также передачу в прикладные программы /из прикладных программ) и несет ответственность ОС ЭВМ.

*Цикл обработки файла* (например, внесение изменений в счета клиентов) включает следующие операции (рис. 1.2):

- открытие файла — занятие устройства, на котором файл размещен (например, МД), создание в оперативной памяти (ОП) *управляющего блока,* в котором записывается справка о состоянии файла и *буфера* (или набора буферов — буферного пула) для хранения текущей, обрабатываемой записи файла;
- организацию цикла, управляемого файлом (заканчивается по исчерпании записей файла — наступлении состояния EOP — end-of-file), после чего выполняется некоторый оператор (обычно освобождение устройства). Цикл должен содержать команду типа READ, CET (ввод записи) или PUT, WRITE (вывод записи) либо REWRITE (обновить запись). Команда READ может являться функциональным аналогом заголовка цикла.
- закрытие файла — выполнение операций по внесению всех окончательных изменений в файл и его реквизиты, освобождение памяти, отведенной под файл, и устройства, на котором он размещался.

Таким образом, траектория данных, обрабатываемых в компьютере, выглядит следующим образом:

- считывание (ввод) порции *(блок)* данных с *накопителя* (ВУ) и помещение его в область ОП *(буфер);*

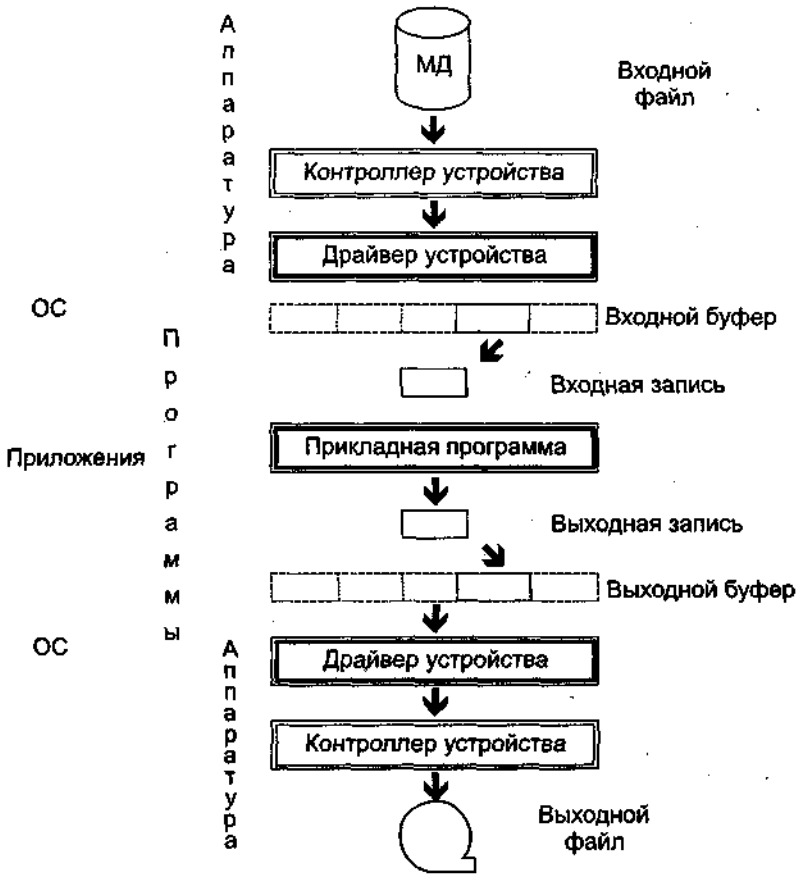



Рис. 1.2. Типичная траектория данных при обработке в ЭВМ

- извлечение данных из буфера, их обработка и помещение обратно или в другой (выходной) буфер;
- после окончания обработки — вывод (запись) результатов на выходной накопитель также в форме одного или нескольких блоков.

Т. е. всякая правильно выполненная и завершенная операция обработки данных начинается считыванием информации с ВУ и Должна заканчиваться записью результата на ВУ. Всякое Прерывание данной Последовательности неминуемо приводит к разрушению информации и потере данных.

***Адресация, имена, спецификация данных в ОС.*** Понятие «управление данными» является характерным не только для. ОС, но и для СУБД (систем управления базами данных). В чем же заключается различие?

На уровне ОС осуществляется связь между *адресом* данных и *именем* (файла). В эпоху до появления ОС и систем программирования (СП) программист должен был писать программы в непосредственных адресах ЭВМ. Элементом такой программы является команда в абсолютных адресах, например, как это было в очень популярной в свое время двухадресной машине Минск 22/32:

10 00 1234 7653

(«сложить содержимое адреса $1234_8$ с содержимым адреса $7653_8$ и записать по адресу $7653_8$»).

При этом управление данными на внешних носителях состояло в написании команд вида:

45 00 1200 0000

47 00 0002 1234

[«на устройстве накопления данных на МЛ перемотать ленту на $12_8$ зон (блоков) затем прочитать 2 зоны в оперативную память, размещая данные с адреса $1234_8$»].

Операционные системы избавляют программиста от таких забот, предоставляя возможность оперировать файлами и их именами.

При этом в различных ОС приняты различные принципы именования данных.

1. Для ОС/360 — прабабушки современных ОС — обозначение файла могло бы выглядеть следующим образом:

```
UNIT=SYSDA, VOL=SER=MASTER, DSN=SYS2.PGMLIB(COPIER3),
```

что означает набор данных (файл), который размещен на устройстве прямого доступа (НМД — SYSDA), причем пакет дисков имеет имя (метку) MASTER, имя файла состоит из группового обозначения (SYS2) и индивидуального (PGMLIB), причем последнее соответствует партиционному (состоящему из разделов) файлу, раздел которого COPIER3 и является основным фигурантом данного описания.

2. Для ОС RSX 11/20 (предшественница UNIX) обозначения выглядят так:

```
устройство:[g,n] имя.расширение;версия
```

где:

— устройство — идентификатор устройства;

— [g,n] — каталог (UIC — User Identification Code), идентификатор пользователя, состоящий из имени (номера) группы (g) и имени пользователя в группе (п);

— имя — выбираемое пользователем наименование НД (не более 8 символов);

— расширение — идентификатор файла (не более 3 символов), используемый для группирования файлов в типы.

Некоторые стандартные типы файлов, используемые в ОС и в пользовательских программах:

`.ftn` — текст программы на Фортране;

`.bas` — текст программы на ЯП Бейсик;

`.cmd` — командный файл;

`.tsk` — исполнительный модуль;

`.txt` — текстовый файл.

Пример: `DP0:[1,7]ADABAS.TSK;1` — это программный модуль с именем ADABAS, размещенный в директории [1,7] на МД с адресом DP0:.

Сокращенное наименование файла может состоять только из имени. При этом устройство и [g,n] берутся из системных умолчаний или пользовательских назначений; расширение — задается стандартным типом файла; версия — максимальная из существующих.

*Спецификация файлов* — соглашения о кратком групповом обозначении некоторой совокупности обрабатываемых, переименовываемых, удаляемых, копируемых и пр. файлов.

В спецификации файлов могут использоваться символы маскирования «*» и «?», вносимые в компоненты обозначения файла, причем «*» соответствует допустимой строке символов, а «?» — одному допустимому символу.

Примеры:

`[*,*]*.TSK;2` — все файлы задач во всех директориях, 2-й версии;

`[1,5]ADA*.SYS` — файлы директории [1,5] с именем, начинающимся с ADA, расширением SYS, 1-й версии;

`[5,5]SYSTEM.??` — файлы с именем SYSTEM, имеющие 2-символьные расширения.

3. UNIX-спецификация файла может иметь вид:

```
ROOT/USR/CPP/EXPERT2.C
```

Здесь описан полный путь для поиска файла (EXPERT2.C — текст программы на ЯП Си), включающий каталоги и подкаталоги ROOT, USR, CPP. Методы групповой спецификации файлов в ОС UNIX подробно описаны ниже в соответствующем разделе.

4. MS DOS-спецификация файла в типовом случае выглядит так:

```
C:\WIN98_SE\PROGRAMMS\COMMAND.COM.
```

Методы групповой спецификации аналогичны указанным выше для RSX.

Вначале для большинства ОС были установлены ограничения на длину и состав имени файла, во многом аналогичные ограничениям на идентификаторы переменных, принятых в то время в языках программирования:

- имя может содержать только символы заглавной латиницы, цифры и подчеркивание;
- имя должно начинаться с буквы;
- длина имени файла не более 8 символов, длина расширения (типа) не более 3.

В дальнейшем, по мере развития и распространения ОС, эти ограничения во многом стали сниматься:

- появилось понятие «длинного имени файла», включающего ранее запрещенные символы (пробелы и пр.);
- были разрешены национальные символы в наименованиях файлов (кириллица и пр.).

Уровни доступа к данным, реализуемые ОС (либо ОС совместно с прикладными программами), приведены на рис. 1.3, При этом участок «Том — Каталог — Файл» реализуется во всех ОС. Участок «Блок — Строка — Слово — Символ» может поддерживаться как функциями ОС, так и в рамках прикладных программ. Иногда это распределение функций устанавливается пользователем (программистом) путем указания типа файла — например *записе-ориентированный* или *потоко-ориентированный* (см. ниже).

Вернемся к проблеме соответствия ОС и СУБД (с точки зрения управления данными). Надо отметить, что СУБД оперируют данными на содержательном уровне, хотя физические структуры, используемые для этих целей, могут быть аналогичны структурам, создаваемым ОС (см. ниже файловые системы ОС).

Коренное отличие СУБД от файловых систем ОС состоит в том, что СУБД устанавливает связь между *содержанием и адресом,* а ОС — между *именем и адресом* данных. В то же время эта грань постоянно подвергается «атакам» с обеих сторон.

Известен ряд ОС, перешедших эту грань, (например, ОС-360 с «индексным доступом к данным», IN-PICK, включающая язык по-

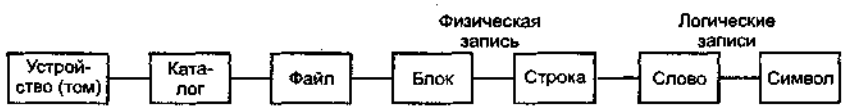



Рис. 1.3. Уровни доступа к данным в абстрактной ОС

иска записей файлов по содержанию, UNIX, включающая команды сортировки, коррекции или объединения содержимого текстовых файлов, наподобие того, как это осуществляется с таблицами данных в СУБД).

Тем не менее следует признать это скорее исключением, чем правилом и в компетенцию ОС надо относить только связь «имя — адрес», оставляя другие зависимости на ответственность прикладных программ и оболочек СУБД и АИПС (автоматизированные информационно-поисковые системы).

***Накопители на магнитных лентах.*** Эти накопители относятся к классу внешних запоминающих устройств последовательного доступа. В них доступ к требуемому набору данных происходит только после завершения перемотки всей предшествующей части магнитной ленты (МЛ). Такие накопители благодаря низкой стоимости, простоте эксплуатации и хранения, компактности и долговременности использования обладают несомненными преимуществами в тех случаях, когда порции данных обрабатываются последовательно друг за другом.

Магнитные ленты для цифровой записи данных размещаются на бобинах или кассетах (подобно лентам для бытовой аудио- или видеозаписи). Однако принципы размещения информации на МЛ в данном случае существенно другие — рис. 1.4:

- информация размещается на носителе в виде блоков (массивов данных фиксированной или переменной длины);
- информационные блоки разделены пустыми промежутками (*gap*), позволяющими считывающему устройству распознать начало (окончание) блока. Размер промежутка между записями выбирается достаточным для разгона ленты до установленной скорости и остановки ее точно на следующем промежутке. Недостаток использования промежутков между записями — уменьшение полезного объема МЛ, так как области, отведенные под промежутки, нельзя использовать для хране-

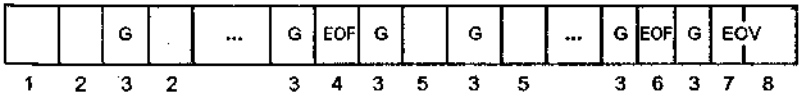



Рис. 1.4. Структура данных на магнитных лентах: 1 — физическое начало ленты (начальный ракорд); 2 — информационные блоки (ИБ) 1-го файла; 3 — CAP, промежуток между блоками; 4 — ЁОР — end-of-file, служебный блок, задающий конец 1-го файла; 5 — информационные блоки (ИБ) 2-го файла; 6 — конец 2-го файла; 7 — EOV — end-of-volume, служебный блок, задающий логический конец ленты; 8 — физический конец ленты (ракорд)

ния данных. Частично указанный недостаток устраняет *процесс блокирования,* суть которого состоит в объединении нескольких записей в блоки;

- блоки разделяются на *информационные* (ИБ — распознаются программами) и *служебные* (распознаются устройством — конец файла и конец тома);
- физическое начало и физический конец ленты обычно определяются оптическим или механическим образом (независимо от содержания ленты).

В ЭВМ обычно применяется девятидорожечная магнитная лента. Информация записывается одновременно девятью магнитными головками. Из девяти одновременно записываемых битов информации восемь являются информационными (один байт) и один — контрольным битом четности. Начало области магнитной ленты, в которую записывается информация, называется точкой загрузки и помечается специальным физическим маркером. Физический маркер представляет собой кусочек алюминиевой фольги, наклеиваемый на расстоянии от начала магнитной ленты. Конец информационной области МЛ помечается таким же физическим маркером, наклеиваемым на расстоянии от конца МЛ. Наличие указанных специальных маркеров, распознавание которых производится фотоэлектронным способом, позволяет осуществить перемотку МЛ к началу информационной области и автоматический останов по достижении ее конца.

Максимальное ограничение на размер блока зависит от размера доступной оперативной памяти (возможность размещения буфера считывания файла). Блокирование увеличивает полезный объем магнитной ленты за счет сокращения числа промежутков между записями. Кроме того, уменьшается количество операций ввода-вывода, так как за одну операцию производится пересылка не одной записи, а сразу нескольких. Преимущества блокирования, заключающиеся в увеличении полезного объема МЛ и уменьшении общего времени работы программы на ввод-вывод данных, значительно превосходят возникающие при этом недостатки, связанные с увеличением объемов данных в программе пользователя и необходимостью выполнения процедур по формированию блоков и разделению принятых блоков на записи.

Устройство записи-считывания информации с МЛ («магнитофон») позволяет осуществлять следующие операции:

- пропустить (вперед или назад) $N$ ИБ;
- пропустить (вперед или назад) $N$ файлов;
- прочитать (записать) блок;

. прочитать (записать) файл;
. позиционировать на конец файла (для записи дополнительных ИБ в этот файл; очевидно, что данные последующих файлов будут затерты);
• позиционировать на начало ленты;
• позиционировать на конец ленты (для записи дополнительного файла).

Очевидно, если блок EOV будет записан в начале ленты, то все файлы становятся недоступными и лента приобретает статус *инициализированной.*

Значение контрольного бита четности выбирается в зависимости от значений восьми информационных битов. Если число единиц в восьми информационных битах нечетное, то в контрольный бит четности заносится единица, а если четное — нуль. Таким образом, общее число единиц в девяти записываемых битах всегда должно быть четным, это контролируется в процессе чтения данных.

Индикатором возникшей ошибки является нечетное число битов в считываемом с МЛ символе. Причинами ошибок часто бывают дефекты покрытия МЛ и налипание на ее поверхности пыли.

Предусмотрена возможность пропуска выявленных дефектных участков на МЛ. Помимо посимвольного контроля, производимого при помощи контрольного бита четности, существует поблочный контроль данных. Его суть заключается в том, что в конце каждого блока записывается контрольная комбинация. В случае возникновения в блоке единичной ошибки посимвольный и поблочный контроль позволяет определить ее местонахождение и выполнить автоматическое исправление. Для этого перед байтом поблочного контроля записывается байт циклического контроля. После обнаружения ошибки делается предположение о наличии временного дефекта МЛ и осуществляется повторная попытка записи информации на то же место. Если и последующие попытки заканчиваются неудачей, то Дефектный участок пропускается. В целях контроля правильности выполнения операций записи-чтения помимо основного набора магнитных головок используется дополнительный.

С помощью дополнительного набора магнитных головок считываются только что записанные на МЛ биты информации, в случае их несовпадения идентифицируется состояние ошибки. Оба набора магнитных головок считывают данные с МЛ и при несовпадении какой-либо пары битов также будет выработан сигнал об ошибке.

Магнитные ленты в силу ряда своих положительных достоинств играют важную роль при организации больших информационных архивов и фондов пакетов программ.

Размещение информации на МЛ связано со следующими проблемами. Для уверенного распознавания промежутка (вар) он должен иметь значительную длину (особенно при высоких скоростях перемотки / чтения). При скорости движения ленты 2—3 м/с длина промежутка должна составлять не менее 1—2 см. Очевидно, что для того чтобы эффективность использования МЛ была достаточно высокой, длина ИБ должна, как минимум, в 2—3 раза превышать длину промежутка (при этом коэффициент полезного использования МЛ будет составлять 60—75 %). При этом также увеличивается скорость обмена между ОП и ВУ, т. к. за одно обращение к МЛ считывается как минимум один ИБ. Однако увеличение длины ИБ требует увеличения объема ОП для размещения буфера, связанного с данным файлом (буфер выделяется операционной системой при открытии файла), в связи с чем одновременное открытие большого числа файлов может оказаться невозможным при ограниченном размере ОП.

В накопителях на магнитной ленте кассетного типа (НКМЛ) носителем информации является стандартная компакт-кассета с магнитной лентой шириной 3,81 мм и длиной 90 м. По сравнению с бобинными магнитными лентами преимущества НКМЛ состоят в компактности и менее высокой стоимости. Их недостатки заключаются в меньшей скорости обмена данными и в десятки раз меньшей информационной емкости.

Основной недостаток внешних запоминающих устройств на магнитных лентах — значительное время ожидания на помещение требуемой области магнитной ленты в зону магнитных головок для выполнения операции записи (считывания). Это занимает в большинстве случаев несколько десятков секунд, что существенно замедляет процесс обработки данных. Прогресс в этой области был достигнут путем разработки запоминающих устройств прямого доступа, включающих в свой состав накопители на магнитных барабанах и дисках (НМД), на гибком магнитном диске (НГМД) и кассетном магнитном диске (НКМД).

***Накопители на магнитных дисках.*** Накопители на магнитных дисках получили наибольшее распространение. В них каждая запись данных имеет свой собственный уникальный адрес, обеспечивающий непосредственный (минуя все остальные записи) доступ к ней. В НМД предусмотрена аналогичная НМЛ возможность последовательного доступа к информации. Накопитель на магнитных дисках сочетает в себе несколько устройств последовательного доступа, причем сокращение времени поиска данных обеспечивается за счет независимости доступа к записи от ее располо-

жения относительно других записей. Конструкция НМД сложнее, чем У НМЛ, а следовательно, выше их стоимость. В НМД в качестве носителей данных используется *пакет магнитных дисков*, закрепленных на одном стержне, вокруг которого они вращаются с постоянной скоростью. Поверхность магнитного диска, покрытая ферромагнитным слоем, называется *рабочей.*

Каждый магнитный диск пакета, кроме верхнего и нижнего, имеет две рабочие поверхности. Верхний и нижний магнитные диски обладают по одной *рабочей поверхности,* расположенной соответственно на нижней и верхней частях указанных дисков. Каждая рабочая поверхность магнитного диска разбита на $N$ окружностей *(дорожек),* пронумерованных от 0 до $N$-1 от края к центру. На каждой из дорожек начало области данных механически идентифицировано при помощи маркера начала оборота. Дорожки, расположенные одна под другой на разных магнитных дисках, образуют соответственно $N$ *цилиндров.*

Из $N$ цилиндров $M$ являются резервными и $N$-$M$ — основными. Дорожки резервных цилиндров пользователям недоступны. Системные средства обеспечивают замену дорожки основного цилиндра, ставшей дефектной, на дорожку запасного цилиндра. Запись и считывание информации в НМД производит механизм доступа, состоящий из держателей магнитных головок (блок магнитных головок).

Количество магнитных головок равно числу рабочих поверхностей на одном пакете дисков. Если пакет состоит из И дисков, то механизм доступа состоит из 10 держателей с двумя магнитными головками на каждом из них. Держатели магнитных головок объединены в единый блок таким образом, чтобы обеспечить их синхронное перемещение вдоль всех цилиндров. Фиксируя блок механизма доступа на каком-либо из цилиндров с помощью только электронного переключения головок, можно сделать переход с одной дорожки на другую данного цилиндра. При фиксированном положении блока механизма доступа возможно обращение к любой из записей, находящихся на дорожках текущего цилиндра. Дорожки в цилиндре нумеруются начиная с верхних. Как правило, обращение к дорожкам происходит с нулевой по последнюю одного цилиндра, потом с нулевой дорожки следующего цилиндра и т. д.

Любая операция чтения (записи) информации с (на) магнитного Диска состоит из трех этапов. На первом этапе происходит механический подвод магнитной головки к дорожке, содержащей требуемые данные. На втором этапе обеспечивается ожидание момента, пока требуемая запись не окажется в зоне магнитной головки. На третьем этапе осуществляется собственно процесс обмена информа-

цией между вычислительной машиной и магнитным диском. Таким образом, общее время, затрачиваемое на операцию записи-считывания, состоит из суммы времен поиска соответствующей дорожки, ожидания подвода записи (так называемое время ротационного запаздывания) и обмена с ЭВМ. Максимальное значение времени ротационного запаздывания равно времени, за которое совершается полный оборот магнитного диска.

В идейном плане размещение информации на МД аналогично МЛ (дорожка МД эквивалентна отрезку МЛ). Адрес информационного блока на МД состоит из номера дорожки и номера блока на дорожке. Начало и конец блока распознаются по промежуткам между блоками, начало и конец дорожки — оптическим (для сменных МД) или электромагнитным (для постоянных МД) датчиком угла поворота оси пакета МД.

Размер блока, очевидно, не может быть больше длины дорожки МД. Считывающее устройство в данном случае ориентировано на выполнение единственной операции — прочитать (или записать) информационный блок, который задан своим адресом. За считывание файла несет ответственность операционная система, поддерживающая файловые структуры на МД.

Соображения по поводу длины блоков, отмеченные выше по поводу МЛ, сохраняют свою силу и для МД, однако здесь возникают и некоторые дополнительные сложности. Использование блоков фиксированной длины на МЛ не дает никаких преимуществ, в то время как для НМД использование блоков фиксированной длины позволяет использовать датчик угла поворота как дополнительный идентификатор конца блока, что приводит к увеличению КПД использования поверхности диска.

При создании нового файла операционная система выделяет под его размещение по меньшей мере один блок, и если в среднем длина файла оказывается намного меньше размера блока, коэффициент использования МД оказывается низким. Тем самым, если предполагается обрабатывать большое число файлов малого объема, то целесообразно задать небольшую стандартную длину блока. Таким образом, выбор длины блока данных на МД определяется противоречивыми факторами как ***за*** увеличение длины, так и ***против*** этого.

***Особенности и характеристики НМД для персональных компьютеров.*** Различают магнитные диски: жесткие (НЖМД, HDD, «винчестер») и гибкие (НГМД, FDD, «флоппи»). HDD являются более скоростными устройствами, чем FDD.

*Винчестер (HDD)* — накопитель на несъемном пакете магнитных дисков был создан в 1973 г. Все магнитные диски (объединенные в *пакет дисков)* *герметически* «упакованы» в общий кожух. Магнитные диски не могут изыматься из HDD и заменяться на аналогичные.

*флоппи (FDD)* (разработка фирмы IBM) — накопитель на съемном гибком магнитном диске (флоппи). флоппи-диск имеет пластиковую основу и находится в пластиковом кожухе. Флоппи-диск вставляется в FDD вместе с кожухом и вращается внутри кожуха со скоростью 300 об/мин.

В персональных компьютерах используются 2 типоразмера FDD: 5.25" (дискета заключена в гибкий пластиковый кожух) и 3.5" (дискета 3.5" заключена в жесткий пластиковый кожух).

Магнитная поверхность разбивается на дорожки (концентрические окружности, см. рис. 1.5). Дорожки нумеруются начиная с 0-й (максимальный радиус). Магнитная поверхность «разбита» также на секторы. Секторы нумеруются начиная с 1-го. Размер каждого сектора обычно равен 512 байт (для MS-DOS). Физический адрес сектора составляется как сумма (точнее — конкатенация) соответствующих номеров: № поверхности, № дорожки, № сектора.

Таким образом, информационный объем дискеты равен:

$$V = P \cdot D \cdot S \cdot 512 \text{ (байт)},$$

где $V$ — информационный объем дискеты (байт); $P$ — количество поверхностей дискеты (одна или две); $D$ — количество дорожек на поверхности; $S$ — количество секторов на дорожке.

Если дискета является системной, то ядро MS-DOS размещается начиная с 0-й дорожки, как более надежной (большая длина и меньшая плотность записи).

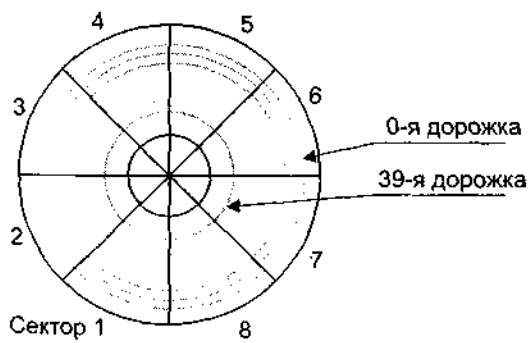



Рис. 1.5. Структура поверхности дискеты (40 дорожек, 8 секторов)

Форматирование дискет производится при инициализации дискеты изготовителем или пользователем с помощью утилиты операционной системы.

В табл. 1.1 приведен перечень основных стандартных форматов гибких дисков, применяемых в IBM PC.

*Таблица 1.1.* **Форматы гибких дисков, используемых в ПЭВМ**

| Количество поверхностей | Количество дорожек на поверхности | Количество секторов на дорожке | Емкость дискеты, Кб | Типоразмер дискеты |
|---|---|---|---|---|
| 2 | 40 | 9 | 360 | 5.25" |
| 2 | 80 | 9 | 720 | 3.5" |
| 2 | 80 | 15 | 1200 | 5.25" |
| 2 | 80 | 18 | 1440 | 3.5" |
| 2 | 80 | 36 | 2880 | 3.5" |

*Кластер* — минимальный размер места на диске, которое может быть выделено файловой системой для хранения одного файла. Определяется он, как правило, автоматически, при форматировании винчестера, по зависимости, указанной в таблице.

Небольшое исключение для системного раздела: если он меньше 2048МБ, то размер кластера всегда 512 байт.

Узнать размер кластера в операционной системе можно несколькими способами. Например, в Windows 2000 можно зайти в Administrative Tools → Computer Management → Storage → Disk Defragmenter. Выбрать нужный диск и нажать на Analyze. Через несколько секунд появится табличка, где есть три кнопки. Нажатие на View Report запускает окошко, в котором содержится информация про выбранный диск, в том числе и Cluster size.

| Размер раздела | Секторов в кластере | Размер кластера |
|---|---|---|
| <512МБ | 1 | 512 байт |
| <1024МБ | 2 | 1K |
| <2048МБ | 4 | 2K |
| <4096МБ | 8 | 4K |
| <8192МБ | 16 | 8K |
| <16384МБ | 32 | 16K |
| <32768МБ | 64 | 32K |
| >32768 МБ | 128 | 64K |

Размер кластера можно выбрать вручную, при форматировании:

```
"format d: /A:size",
```

где size — размер кластера в байтах. Однако существуют некоторые правила, которых следует придерживаться: во-первых, размер кластера должен быть кратен размеру физического сектора, то есть 512 байтам в подавляющем большинстве случаев; во-вторых, есть ограничения по количеству кластеров на разделе.

***файловые системы.*** Всякая операционная система создает на каждом томе (дискете, диске, пакете дисков, CD-ROM и пр.) совокупность системных данных, которая называется *файловой системой* (файловой структурой).

*файловая система (пустая)* создается при инициализации (разметке) тома, затем корректируется ОС (подсистемой управления данными) при текущей работе, б процессе создания, удаления, модификации (увеличения или уменьшения объема) файлов пользователя, содержащих программы или данные.

Файловая система включает в себя *таблицу содержания* и *область данных* — совокупность блоков на диске, идентифицируемых своими номерами / адресами. Обычно адрес блока состоит из 3 чисел — № цилиндра (совокупность дорожек, доступных при фиксированном положении блока головок считывающего устройства), № поверхности (дорожки в цилиндре), № блока на дорожке.

Пример простейшей (абстрактной) таблицы содержания, оглавления тома (диска, пакета дисков), которая в разных ОС имеет различные наименования — VTOC — Volume Table of Content (Таблица Содержания Тома), PAT — File Allocation TaBle (Таблица Размещения Файлов), FDT — File Definition TaBle (Таблица Определения Файлов) и т. п., приведена на рис. 1.6.

Она состоит из трех областей:

- *область файлов.* Это таблица, имеющая обычно ограниченное (в приведенном примере $N = 6$) число строк $N$ (в MS-DOS, например, $N = 500$, т. е. число файлов не более 500). Количество столбцов $M$ (в примере $M = 5$) обычно выбирается из тех соображений, чтобы 85—95 % файлов, создаваемых пользователями, содержало бы не более $M$ блоков, что зависит как от размера блока и типа пользователя, так и от общего уровня развития информационного и программного обеспечения. Первый столбец таблицы в каждой строке *(заглавная запись — Title Record)* содержит данные о файле, в данном примере — имя файла;

| Имя файла (заглавная запись) | Номера блоков, выделенных для размещения файлов | | | | |
|---|---|---|---|---|---|
| File_1 | 1 | 3 | 7 | 5 | 13 |
| File_2 | 41 | 8 | | | |
| PHe_3 | | | | | |
| PHe_4 | 3 | | | | |
| | | | | | |
| | | | | | |
| Область переполнения | | | | | |
| PHe_1 | 23 | | | | |
| | | | | | |
| Список свободных блоков | | | | | |
| 2 | 4 | 6 | 9 | 10 | 11 | 12 |
| 13 | | | | | | |
| Список сбойных блоков | | | | | |
| 12 | 24 | 7 | | | | |

Рис. 1.6. Простейшая таблица оглавления тома

- *область переполнения* — дополнительная таблица аналогичной структуры, в которую записываются номера блоков особо длинных файлов (в примере — File_1). Организация таблицы размещения в форме области файлов и *облает* переполнения, очевидно, позволяет сэкономить на объеме таблицы в целом, не ограничивая в то же время вероятной длины файла;
- *список свободных блоков* — необходимая информация для размещения создаваемых или расширяемых файлов. Список создается при инициализации и включает все блоки, кроме поврежденных, а затем корректируется ири создании, удалении, модификации файлов;
- *список сбойных блоков.* Это таблица, создаваемая при инициализации (разметке) тома (диска), пополняемая программами диагностики (примером которых может служить хорошо известный пользователям NDD — Norton Disk Doctor) и предотвращающая распределение испорченных областей на магнитном носителе под файлы данных.

Здесь не указаны такие известные атрибуты файлов, как длина (в байтах), время создания, тип (архивный, скрытый, только для чтения, не для исполнения и пр.), которые могут содержаться в заглавной записи таблицы (колонка 1 на рис. 1.6).

В развитых системах коллективного пользования такие данные содержатся в специальных таблицах разделения полномочий, по-

с[illegible]льку перечисленные да и другие атрибуты должны быть соотнесены с конкретными пользователями.

Кроме того, где-то должны быть размещены метка тома (имя и тип/объем), количество занятого и свободного пространства и прочая совокупная информация по тому данных.

Перечислим особенности ситуации, зафиксированной на рис. 1.6 в простейшей (искусственной) файловой системе.

File_1 занимает 6 блоков, это число больше максимального, поэтому адрес блока № 6 (23) размещен в таблице переполнения;

File_2 занимает 2 блока, что меньше ограничения, поэтому вся информация сосредоточена в области файлов.

Имеются следующие конфликтные ситуации:

- File_3 не содержит ни одного блока (следовательно, файл был удален, но заглавная запись сохранилась);
- File_4 и File_1 ссылаются на блок № 3. Это ошибка, поскольку каждый блок должен быть закреплен за единственным файлом;
- File_1 содержит ссылку на блок № 7, помеченный как сбойный (нечитаемый). Это приведет к невозможности корректно полностью прочитать данный файл — ситуация, знакомая каждому, работавшему с НГМД;
- в списке свободных блоков содержатся номера блоков № 12 (помеченный как сбойный) и № 13 (распределенный под File_1).

Это очевидные свидетельства начавшегося разрушения файловой системы. Перечисленные конфликты могут иметь своими источниками сбои, программные ошибки (разработчиков ОС), некорректное завершение ОС или целенаправленную деятельность вирусных или иных злонамеренных программ.

Рассмотренный пример таблицы оглавления относится к случаю так называемой прямой адресации доступа (рис. 1.7). Здесь очевидны следующие особенности:

- таблица создается при инициализации и, даже будучи пустой, занимает определенный объем;
- создание файла (даже состоящего из одного байта) приводит к выделению блока и занятию строки таблицы.

Косвенная — списковая (рис. 1.8, *а)* и мультисписковая (рис. 1.8, *б)* адресации создают управляющие структуры по мере необходимости (при заполнении файла). Высвобождение памяти в [illegible]ковой структуре осуществляется автоматически при удалении [illegible]ого промежуточного блока, содержащего также адрес последующего блока файла. Очевидно, это увеличивает и опасность утраты

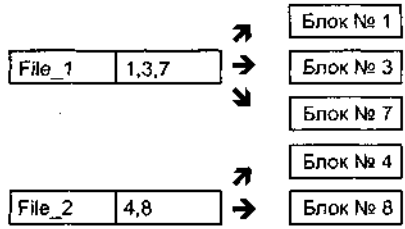



Рис. 1.7. Доступ к данным с прямой адресацией

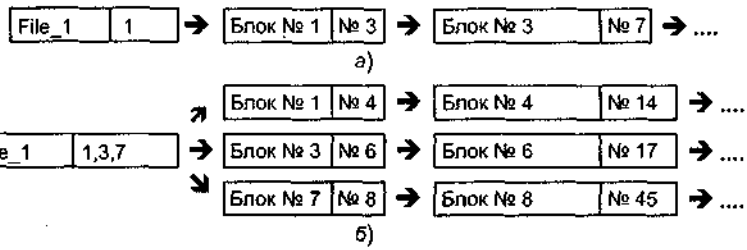



Рис. 1.8, *а)* — списковая организация доступа к данным (косвенная адресация); б) — комбинированная (мультисписковая) организация доступа

данных при ошибочном удалении или разрушении промежуточного блока файла.

Все операционные системы, как правило, поддерживают следующие элементы иерархических файловых систем: обычные файлы, каталоги, специальные байт-ориентированные и блок-ориентированные файлы. Файл является массивом байтов (блоков фиксированной длины). Каталоги обеспечивают связь между именами файлов и собственно файлами. Каждый элемент каталога содержит имя файла и ссылку на конкретный файл. Для именования файлов используются корневой и текущий каталоги.

Внешние устройства (такие, как терминал, принтер) также часто представляются как файлы для упрощения работы с ними.

Устройства, на которые осуществляется вывод данных из программы или с которых осуществляется ввод (это может быть одно и то же физическое устройство, как это было в случае ранних терминалов; в современных, т. н. ANSI-терминалах — монитор и клавиатура рассматриваются как два отдельных, независимых устройства), могут быть подразделены на следующие типы:

- передачи информации битовым потоком;
- посимвольного обмена информацией;
- передачи информации порциями (записями).

Фактически это как бы «портрет» устройства, каким его «видит» прикладная программа, через посредство драйвера устройства и программ операционной системы, ответственных за ввод-вывод информации. Одно и то же устройство может быть представлено как генератор потока символов (потоко-ориентированное устройство) или записей (записе-ориентированное). Поэтому, скорее, стоит говорить о типе файлов, расположенных на том или ином устройстве.

Различают следующие виды файлов:

- по типу записей:
  - √ файлы с записями постоянной длины,
  - V файлы с записями переменной или неопределенной длины,
  - V файлы, образующие байтовый или битовый поток;
- по способу выборки информации:
  - V файлы последовательного доступа,
  - V файлы прямого доступа,
  - √ файлы доступа по индексу (обычно — файлы базы данных).

Далее возникает проблема идентификации данных, размещенных на носителе (в файле). Каким образом можно правильно сопоставить тем или иным битовым комбинациям, размещенным в файле, те или иные области оперативной памяти, куда они должны считываться с носителя, для последующей обработки или обновления. В частности, различные способы идентификации связаны с понятиями базового и стандартного буферизованного ввода-вывода.

*Базовый ввод-вывод.* Базовый уровень ввода-вывода обеспечивает обмен с файлом, интерпретируемым как одномерный массив байтов с прямым последовательным доступом.

Для каждого файла система ведет указатель чтения /записи. При чтении (записи) *n* байтов указатель продвигается вперед по файлу на *n* байтов и устанавливается в позицию очередного читаемого (записываемого) символа.

В начале работы с файлом его создают или открывают. Файлы, открытые данной программой, имеют внутреннюю (в пределах данного процесса) нумерацию начиная с 0. Системный вызов, открывающий файл, возвращает номер открытого файла, который используется при чтении и записи. После того как файл открыт, к нему могут применяться функции чтения или записи. При чтении из файла последовательно читаются очередные байты и возвращается число прочитанных байтов. Оно может оказаться меньше требуемого числа, если до конца файла осталось меньше байтов, чем требуется, или если устройство не передает такого числа байтов.

При записи в файл записываются очередные байты, расположенные в памяти процесса. Если возвращаемое после записи значение не

равно числу записываемых байтов, это свидетельствует об ошибке. Если очередной записываемый байт оказывается за концом файла, то обеспечивается соответствующее увеличение размера файла.

Прямой доступ к файлу реализуется вызовом соответствующей функции, устанавливающей указатель чтения/записи в требуемую позицию. Позиционирование возможно в тех файлах, где оно допускается типом файла или природой внешнего устройства.

Процесс может управлять открытым файлом, получая и задавая значения его атрибутов, а также блокируя участки файла (или файла целиком) от доступа других процессов.

По окончании работы с файлом его следует закрыть. При завершении программы все открытые файлы закрываются автоматически.

Особенность каталога *состоит* в том, что запись в него может делать только система — программа может только читать элементы каталога.

*Стандартный буферизованный ввод/вывод* является надстройкой над базовым уровнем. Подобно базовому уровню, он интерпретирует файл (поток, в терминах данного уровня), как одномерный массив байтов с прямым доступом.

Потоки дают возможность обмениваться с файлом, буферизируя данные в памяти процесса. При чтении из потока происходит считывание блока данных из файла в буфер, а из буфера процессу передается столько байтов, сколько он запросил. Когда при очередном чтении из потока в буфере уже нет требуемых данных, происходит очередное считывание блока данных из файла в буфер. Аналогично при записи в поток передаваемые процессом данные накапливаются в буфере и передаются системе для записи в файл только после того, как буфер заполнится, при вызове специальной функции или при закрытии потока (кстати, при выключении компьютера содержимое буферов операционной системы теряется).

Когда процесс начинает работу, он получает открытыми 3 стандартных потока: стандартный ввод (**'stdin'** — в UNIX, **'CON'** — канал О в MS-DOS), стандартный вывод (**'stdout'** — в UNIX, **'CON'** — канал 1 в MS-DOS), стандартную диагностику (**'stderr'** — в UNIX, канал 2 в М5-ВО3). М3-ВО5 предоставляет дополнительно еще 3 стандартных потока — канал связи (**'AUX'** — канал 3) и стандартное устройство печати (**'PRN '** — канал 4). Стандартный ввод используется как устройство чтения по умолчанию, стандартный вывод — как устройство записи по умолчанию, стандартная диагностика — для вывода сообщений об ошибках.

В операционных системах средства ввода-вывода буферизированного обмена позволяют передавать символы, символьные стро-

ки, форматировать выводимую информацию. Как и на базовом уровне, возможна установка позиции в потоке.

Операционные системы предоставляют следующие системные вызовы: запрос на смену и получение имени текущего каталога; создание, открытие, закрытие, удаление, переименование и получение информации о файле или каталоге, позиционирование в них.

***Разделение доступа к данным в ОС.*** Все рассматриваемые операционные системы поддерживают операции блокировки файла для зашиты доступа к нему со стороны других процессов в многозадачной среде:

Пользователи, которым разрешено входить в систему, перечислены в учетной базе пользователей. Пользователи объединены в группы; последние перечислены в учетной базе групп. Каждому пользователю и каждой группе присвоены целочисленные идентификаторы.

Входя в систему, пользователь сообщает ей свое имя, по которому определяется его идентификатор и права доступа. Вызывая команды, пользователь тем самым порождает процессы, которые наследуют его права, пользовательский и групповой идентификаторы.

С каждым файлом связана пара идентификаторов: пользовательский и групповой. Файл наследует эти идентификаторы от эффективных идентификаторов процесса, создавшего данный файл. Процесс, эффективный пользовательский идентификатор которого совпадает с пользовательским идентификатором файла, считается владельцем данного файла.

Файл можно читать, писать и выполнять. Если файл является каталогом, выполнение означает поиск в нем. Права процессов при доступе к файлу хранятся в атрибутах защиты файла. Эти атрибуты при создании файла могут быть изменены только на основе соответствующих прав.

Проверка прав происходит, когда процесс пытается открыть Файл для чтения или записи, выполнить его.

Все пользователи, имеющие доступ в систему, разделены по отношению к файлу на три категории: владельцы (эффективный пользовательский идентификатор процесса совпадает с пользовательским идентификатором файла), члены группы (эффективный групповой идентификатор процесса совпадает с групповым идентификатором файла) и прочие.

Процесс может иметь зависящие от реализации привилегии, которые дают ему дополнительные права при доступе к файлу.

Если процесс не имеет привилегий, то ему разрешается доступ к файлу в трех случаях:

- процесс является владельцем файла (см. выше) и атрибуты защиты файла разрешают запрашиваемый вид действия владельцу;
- эффективный групповой идентификатор процесса совпадает с групповым идентификатором файла и атрибуты файла разрешают запрашиваемый вид действия группе;
- атрибуты файла разрешают запрашиваемый вид действия всем процессам.

Если ни одно из условий не выполняется, то процесс не получает доступ к файлу.

Системные вызовы операционной системы обеспечивают: получение информации о пользователях и группах в учетной базе (при наличии соответствующих привилегий) и получение информации о защите конкретного файла.

***Управление периферийными устройствами.*** Существует два типа управления периферийными устройствами — прямой и косвенный.

Прямое управление реализует непосредственную связь между процессором и периферийным устройством. Управление осуществляется с помощью последовательности специальных команд. При этом центральный процессор реализует полный алгоритм: инициализацию, проверку готовности, останов.

При косвенном управлении между процессором и периферийным устройством помещается специальный процессор, который осуществляет управление операциями ввода/вывода. Такой процессор называется каналом. Центральный процессор инициирует ввод/вывод и переход на выполнение других операций, а канал управляет периферийным устройством по специальной программе, канальной. Для синхронной работы центрального процессора и канала используются флаги занятости или прерывания. Программа управления периферийным устройством — супервизор ввода/вывода — в ответ на прерывание планирует и организует ввод/вывод, при необходимости организует обновление данных. Для сглаживания скоростей в периферийном канале и управляющем устройстве между ними ставится буфер, роль которого выполняет оперативная память. При определении длины буфера должен обеспечиваться компромисс между эффективностью использования оперативной памяти и внешней памяти (чем длиннее буфер, тем больше времени работает канал независимо от центрального процессора).

*Реализация подсистемы ввода/вывода в операционной системе UNIX* имеетследующие особенности:

- подсистема имеет унифицированный доступ как к периферийным устройствам, так и файлам;

. подсистема работает как синхронная система (каждый процесс, требующий ввода, приостанавливается в точке для завершения операции).

Для управления используются два вида ввода/вывода: байт-ориентированный (интерфейс применяется для доступа к печатающим устройствам, каналам связи и реализуется без буферизации) и блок-ориентированный (интерфейс адресуется к периферийным устройствам как к последовательным блокам по 512 байт).

В состав ОС входят драйверы и специальные таблицы подключения ядра операционной системы к драйверам различных устройств. Каждый драйвер состоит из двух частей:

- набора программных модулей для операций открытия, закрытия, чтения и записи в файл данных;
- модуля обработки прерывания. Для байт-ориентированной передачи прерывание наступает после передачи первого байта, а для блок-ориентированной — после передачи первого блока.

*Форматы файлов.* В зависимости от типа и назначения файлов и возможностей ОС (методов доступа) файл может передаваться в прикладную программу как целое или блоками (физическими записями) либо логическими записями (строками, словами, символами).

В системе OS/360 основную роль играли два типа файлов:

- символьные (исходные программы или данные);
- двоичные (программы в машинных кодах).

В современных системах активно используется значительно большее разнообразие файлов, из которых мы перечислим наиболее типичные файлы данных:

- *текстовые файлы* — обобщенное название для простых и размеченных текстов, ASCII-файлов и других наборов данных символьной информации, которые интерпретируются и обрабатываются текстовыми редакторами, процессорами, анализаторами (Lexicon, Word, TEC, анализаторы SGML, HTML);
- *текст без разметки* (планарный) — файл, содержащий только отображаемые (воспроизводимые на всех печатающих устройствах и терминалах) символы кода ASCII, а также простейшие управляющие символы: CR — возврат каретки; LF — перевод строки; TAB — символ табуляции, иногда LF — новая страница;
- *текст с разметкой* — планарный файл, содержащий бинарную и символьную разметку, управляющую отображением информации (программно и/или аппаратурно);
- *ASCII-файл* — содержит только отображаемые коды левой части кодовой таблицы ASCII (латиница и служебные символы),

обычно применяется для хранения документов с символьной разметкой (RTF, SGML, HTML);

- *табличный файл* — содержит форматированные данные (символьные, численные и др), образующие строки и столбцы таблиц, создаваемых и обрабатываемых табличными СУБД (FoxPro, Clipper, MS Access) и/или табличными процессорами (SuperCalc, MS Excell и др.);
- *графический файл* — бинарный файл, содержащий графическую информацию. Форматы: TIF (Tagged Image File), BMP (Bit-Mapped Picture), а также ряд других — PCX, PIC и т. д.;
- *мультимедиа файлы* — бинарные файлы, содержащие оцифрованную аудио- (типы WAW или MIDI-Sequencer), видео- (формат MPEC) или смешанную информацию.

В табл. 1.2 приведены основные типы файлов, используемых в ОС DOS, Windows, и соответствующие им расширения имени.

***Таблица* 7.2. Основные типы файлов, обрабатываемых в ПЭВМ**

| Тип, расширение имени | Вид информации, содержащейся в файле |
|---|---|
| exe, com | Программа, готовая к исполнению |
| bat | Текстовый командный файл |
| sys | Системный файл |
| ovl, ovr | Оверлейный файл |
| pif | Программно-информационный файл Windows |
| txt, lst | Текстовый файл в формате DOS |
| doc | Документ (чаще всего в формате WinWord) |
| rtf | Размеченный текстовый файл (Rich Text Format) |
| dot | Файл формата документа (Document Type) |
| pdf | Формат документа Adobe Acrobat |
| wri | Документ редактора Write для Windows |
| wps | Документ текстового процессора MS WORKS |
| bak, old | Старая копия файла, создаваемая перед его изменением |
| arj, rar, zip, izh, ain, arc, ice, pak, zoo | Архивные файлы |
| bas | Текст программы на языке Basic |
| pas | Текст программы на языке Turbo Pascal |
| c | Текст программы на ЯП Си |
| bmp, pcx, gif, tif, jpg, ico | Графические файлы |
| dbf | Файлы базы данных формата DBase, FoxBase, Cliper |

*Продолжение табл. 1.2*

| Тип, расширение имени | Вид информации, содержащейся в файле |
|---|---|
| wdb | Базы данных формата MS WORKS |
| wks | Электронная таблица формата MS WORKS |
| xls | Электронные таблицы EXCEL |
| lib, dll | Файлы библиотек |
| dat | Файл данных |
| ini | Файл инициализации |
| hlp | Файл справки (подсказки, помощи) |
| ext | Файл расширений |
| mnu | Файл меню |
| wav, mid, mp3, mod | Звуковые файлы |
| avi, mov, mpg | Файлы видеоклипов |

## 1.3. Управление заданиями (процессами, задачами)

Основными понятиями управления прохождением задач в ЭВМ являются *процесс, задача, работа, программа, ресурс, дисциплина распределения ресурса* [14].

*Процесс* — минимальный программный объект, обладающий собственными системными ресурсами (запущенная программа).

***Классификацияпроцессов.*** *По временным характеристикам различают интерактивные, пакетные процессы и процессы реального времени.* Время существования интерактивного процесса определяется реакцией ЭВМ на запрос обслуживания и составляет секунды. Процессы реального времени имеют гарантированное время окончания работы и время реакции мсек. Пакетные процессы запускаются один вслед за другим и время реакции часы и более.

*По генеалогическому признаку различают порождающие и порожденные процессы.*

*По результативности различают эквивалентные, тождественные и равные процессы.* Все они имеют одинаковый конечный результат, но эквивалентные процессы могут реализовываться как на одном, так и на многих процессорах по одному или разным алгоритмам, то есть они имеют разные трассы, которые определяют порядок и дли-

тельность пребывания процесса в разных состояниях. Тождественные процессы реализуются по одной и той же программе, но имеют разные трассы. Одинаковые процессы реализуются по одной программе и имеют одинаковые трассы.

*По времени развития процессы делятся на последовательные, параллельные и комбинированные* (для последних есть точки, в которых существуют оба процесса, и точки, в которых существует только один процесс).

*По месту развития процессы* делятся на *внутренние* (реализуются на центральном процессоре) и *внешние* (реализуются на внешних процессорах).

*По принадлежности к операционной системе* процессы бывают *системные* (исполняют программу из состава операционной системы) и *пользовательские.*

*По связности различают процессы:*

а) взаимосвязанные, которые имеют какую-то связь (пространственно-временную, управляющую, информационную);

б) изолированные — слабо связанные;

в) информационно-независимые, которые используют совместные ресурсы, но имеют собственные информационные базы;

г) взаимодействующие — имеют информационные связи и разделяют общие структуры данных;

д) взаимосвязанные по ресурсам;

е) конкурирующие.

*Порядок взаимосвязи процессов* определяется правилами синхронизации. Процессы могут находиться в отношении:

а) предшествования — один всегда находится в активном состоянии раньше, чем другой;

б) приоритетности — когда процесс может быть переведен в активное состояние только в том случае, если в состоянии готовности нет процессов с более высоким приоритетом, или процессор свободен, или на нем реализуется процесс с меньшим приоритетом;

в) взаимного исключения — в процессе используется общий критический ресурс, и процессы не могут развиваться одновременно: если один из них использует критический ресурс, то другой находится в состоянии ожидания.

*Ресурс* — любой потребляемый (расходуемый) объект. По запасам ресурсы подразделяются на исчерпаемые и неисчерпаемые. Потребители ресурсов — процессы. Ресурс — средство вычислительной системы, которое может быть выделено процессу на определенный интервал времени.

*Процессор* — любое устройство в составе ЭВМ, способное автоматически выполнять допустимые для него действия (процессоры, каналы и устройства, работающие с каналами). Реализация системы управления процессами в составе ОС предъявляет определенные требования к свойствам процессоров.

*Классификация ресурсов. По признаку реальности* ресурсы делятся на *физические и виртуальные* (последние только в отдельных свойствах схожие с физическими ресурсами).

*По возможности расширения свойств* делятся на *эластичные и жесткие* (не допускающие виртуализации).

*По степени активности* разделяются на *пассивные и активные* (могут выполнять действия по отношению к другим ресурсам).

*По времени существования: постоянные* (доступны во все время процесса: и до, и после его работы) и *временные.*

*По степени важности: основные и второстепенные* (допускают альтернативное развитие процесса при их отсутствии).

*По функциональной избыточности при распределении: дорогой,* но предоставляемый быстро, и *дешевый,* но предоставляемый с ожиданием.

*По структуре: простые* (не содержит составных элементов) и *составные.* Они различаются числом состояний: простой может быть только в двух состояниях — доступен или занят.

*По характеру использования распределяемых ресурсов: потребляемые и воспроизводимые* ресурсы (допускают многократное использование и освобождение).

*По характеру использования: последовательно и параллельно используемые* (используются несколькими процессами).

*По форме реализации: жесткие* (в принципе не допускают копирования) и *мягкие* (допускают тиражирование и подразделяются на программные и информационные ресурсы).

*Дисциплина распределения ресурса* определяет порядок использования многими процессами того или иного ресурса, который в каждый момент времени может обслуживать только один процесс.

*Управление процессами.* Процесс — это программный модуль, выполняемый в центральном процессоре (CPU). Операционная система контролирует следующую деятельность, связанную с процессами:

- создание и удаление процессов;
- планирование процессов;
- синхронизация процессов;
- коммуникация процессов;
- разрешение тупиковых ситуаций.

Не следует смешивать понятия процесс и программа. Программа — это план действий, а процесс — это само действие, поэтому понятие процесса включает:

- программный код;
- данные;
- содержимое стека;
- содержимое адресного и других регистров процессора.

Таким образом, для одной программы могут быть созданы несколько процессов в том случае, если с помощью одной программы в CPU выполняется несколько несовпадающих последовательностей команд. За время существования процесс многократно изменяет свое состояние.

Различают следующие состояния процесса (рис. 1.9):

- *новый* (процесс только что создан);
- *выполняемый* (команды программы выполняются в CPU);
- *ожидающий* (процесс ожидает завершения некоторого события, чаще всего операции ввода-вывода);
- *готовый* (процесс ожидает освобождения CPU);
- *завершенный* (процесс завершил свою работу).

Переход из одного состояния в другое не может выполняться произвольным образом. На рис. 1.9 приведена типовая диаграмма переходов для состояний процессов.

Каждый процесс представлен в операционной системе Набором данных, называемых *таблица управления процессом* (ТУП — PCB — process control block). В PCB процесс описывается набором значений, параметров, характеризующих его текущее состояние и используемых операционной системой для управления прохождением процесса через компьютер.

На рис. 1.10 схематически показано, каким образом операционная система использует PCB для переключения процессора с одного процесса на другой.


Рис. 1.9. Состояния процесса

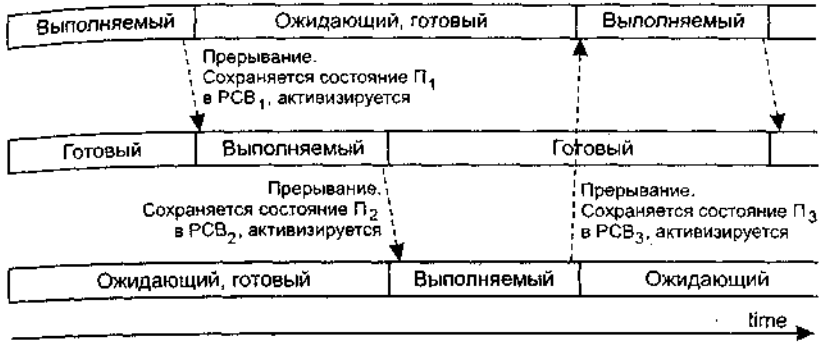



Рис. 1.10. Переходы между процессами

*Планирование процессов. Понятие очереди.* Система управления процессами обеспечивает прохождение процесса через компьютер. В зависимости от состояния процесса ему должен быть предоставлен тот или иной ресурс. Например, новый процесс необходимо разместить в основной памяти, следовательно, ему необходимо выделить часть адресного пространства. Процессу в состоянии готовый должно быть предоставлено процессорное время. Выполняемый процесс может потребовать оборудование ввода-вывода и доступ к файлу [14].

Распределение процессов между имеющимися ресурсами носит название *планирование процессов.* Одним из методом планирования процессов, ориентированных на эффективную загрузку ресурсов, является метод очередей ресурсов. Новые процессы находятся во входной очереди, часто называемой *очередью работ — заданий.*

Входная очередь располагается во внешней памяти, во входной очереди процессы ожидают освобождения ресурса — адресного пространства основной памяти.

Готовые к выполнению процессы располагаются в основной памяти и связаны *очередью готовых процессов.* Процессы в этой очереди ожидают освобождения ресурса *процессорное время.*

Процесс в состоянии ожидания завершения операции ввода-вывода находится в одной из *очередей к оборудованию* ввода-вывода.

При прохождении через компьютер процесс мигрирует между различными очередями под управлением программы, которая называется *планировщик* (scheduler).

Операционная система, обеспечивающая режим мультипрограммирования, обычно включает два планировщика — *долгосрочный* и *краткосрочный.* Например, в OS/360 долговременный планиров-

шик назывался *планировщиком заданий,* а краткосрочный — *супервизором задач.*

На уровень долгосрочного планирования выносятся редкие системные действия, требующие больших затрат системных ресурсов, на уровень краткосрочного планирования — частые и более короткие процессы. На каждом уровне существует свой объект и собственные средства управления им.

Основное отличие между долгосрочным и краткосрочным планировщиками заключается в частоте запуска, например: краткосрочный планировщик может запускаться каждые 100 мс, долгосрочный — один раз за несколько минут.

*Долгосрочный планировщик* решает, какой из процессов, находящихся во входной очереди, должен быть переведен в очередь готовых процессов в случае освобождения ресурсов памяти.

Долгосрочный планировщик выбирает процесс из входной очереди с целью создания неоднородной мультипрограммной смеси. Это означает, что в очереди готовых процессов должны находиться в разной пропорции как процессы, ориентированные на ввод-вывод, так и процессы, ориентированные на преимущественную работу с CPU.

На уровне долгосрочного планирования объектом является не отдельный процесс, а некоторое объединение процессов по функциональному назначению, которое называется работой (приложением). Каждая работа рассматривается как независимая от других работ деятельность, связанная с использованием одной или многих программ и характеризующаяся конечностью и определенностью. По мере порождения новых работ создается собственная виртуальная машина для их выполнения. Например, в ОС Windows 95 для каждого 32-разрядного приложения реализуется своя виртуальная машина. Распределение машин производится однократно в отличие от краткосрочного планирования, где процессор процессу может выделяться многократно.

*Краткосрочный планировщик* решает, какой из процессов, находящихся в очереди готовых процессов, должен быть передан на выполнение в СРП В некоторых операционных системах долгосрочный планировщик может отсутствовать. Например, в *системах разделения времени* (time-sharing system) каждый новый процесс сразу же помещается в основную память.

На уровне краткосрочного планирования объектом управления являются процессы, которые выступают как потребители центрального процессора для внутренних процессов или внешнего процессора для внешних процессов. Причинами порождения процесса могут

быть процессы на том же уровне или сигналы, посылаемые от долгосрочного планировщика.

Выделение процессора процессу производится многократно, с целью достижения эффекта мультипрограммирования, и такой процесс называется диспетчеризацией.

***Взаимодействие процессов.*** Совместно выполняемые процессы могут быть либо *независимыми,* либо *взаимодействующими.* Взаимодействие процессов часто понимается в смысле взаимного обмена данными через общий буфер данных.

Взаимодействие процессов удобно рассматривать в схеме производитель-потребитель. Например, программа вывода на печать производит последовательность символов, которые потребляются драйвером принтера, или компилятор производит ассемблерный текст, который затем потребляется ассемблером.

Для взаимодействия процесса-производителя и процесса-потребителя создается совместный буфер, заполняемый процессом-производителем и потребляемым процессом-потребителем.

Буфер имеет фиксированные размеры и, следовательно, процессы могут находиться в состоянии ожидания, когда:

- буферзаполнен — ожидает процесс-производитель;
- буфер пуст — ожидает процесс-потребитель.

Буфер может предоставляться и поддерживаться самой ОС, например с помощью средств межпроцессной коммуникации, либо должен быть организован прикладным программистом. При этом оба процесса используют общий участок памяти.

Взаимодействие заключается в передаче данных между процессами или совместном использовании некоторых ресурсов и обычно реализуется с помощью таких механизмов, как транспортеры, очереди, сигналы, семафоры.

*Транспортеры* (каналы). Являются средством взаимодействия родственных процессов, представляют собой область памяти, имеющую файловую организацию, для которой обеспечивается запись и считывание данных в транспортере. Реализуется очередь обслуживания. Порядок записи данных на транспортер неизменен, не допускается повторное считывание данных. Обмен данными происходит не непосредственно, а через транспортер. Из вызвавшего процесса задается размер транспортера. Дочерние процессы могут использовать родительский транспортер.

*Очереди.* Эти механизмы могут обеспечивать передачу или использование общихданных без перемещения данных, а с передачей элемента очереди, содержащего указатель данных и объем массива данных. Очередь используется вместе с механизмом общей памяти.

Элемент очереди может быть считан с уничтожением или без уничтожения этого элемента. Чтение элемента может осуществляться в соответствии с механизмом очереди или стека. Чтение элементов очереди осуществляет только создающий очереди процесс, все другие процессы могут только записать элемент в очередь. Создающий процесс может выполнять следующие действия над очередью:

- создание очереди;
- « просмотр очереди;
- чтение очереди;
- закрытие очереди.

Записывающий процесс осуществляет действия:

- открыть очередь;
- записать в очередь;
- закрыть очередь.

Имя очереди, которое присваивается создающим процессом, имеет вид полной спецификации файла. Ожидание элементов в очереди организуется с помощью семафора, сигнализирующего о записи элемента в очередь. Для работы с очередью определены такие дополнительные функции:

- определение количества элементов в очереди в текущий момент;
- очистка очереди создавшим ее процессом.

Преимущества очереди: передача данных по указателю без копирования, гибкое изменение порядка передачи и доступа, возможность просмотра элементов очереди без их удаления.

*Сигналы.* Сигнал является механизмом передачи требования от одного процесса к другому на немедленное выполнение действия. Обработчик сигнала создается процессом и помещается в начале первого потока процесса. Является аналогом обработки прерывания. При передаче управления обработчику передается адрес возврата и тип принятого сигнала. Процесс, посылающий сигнал типа флаг, может передать дополнительную информацию обработчику сигнала. Характер выполняемых действий при возникновении сигнала: обработка системной ошибки при появлении сигнала, блокирование сигнала, передача управления подпрограмме.

*Семафоры.* Являются механизмами передачи сообщений от одного потока к другому о наступлении некоторого события. Различают семафоры системные и оперативной памяти. Семафоры оперативной памяти — двойное слово в памяти системы, его описатель — адрес места в памяти. Такие семафоры не создаются и не открываются, а устанавливаются в определенное состояние. Процессы, использующие семафоры оперативной памяти, должны иметь доступ к

соответствующему сегменту памяти. Операционная система такие семафоры не обслуживает и не сообщает об их освобождении или захвате. При создании семафора или его открытии возвращается описатель семафора, включающий его имя. Операционная система контролирует завершение каждого процесса, владеющего системным семафором, и освобождает его для процессов.

Если семафор свободен, то он захватывается вызывающим его процессом, если семафор занят, то вызвавший его поток переходит в режим ожидания освобождения семафора или ожидает истечения времени. Если семафор освобождается всеми использующими его процессами, то он удаляется из системы.

Управление семафором реализуется с помощью функций:

- установки семафора с целью сигнализации;
- ожидания вызывающим потоком, пока семафор не будет выключен;
- ожидания потоком выключения одного из нескольких семафоров.

Операционные системы используют разные термины для определения способов межпроцессного взаимодействия.

В операционных системах 03/2 и Microsoft Windows существует специальный механизм для взаимодействия процессов в реальном масштабе времени. Этот механизм называется DDE (*Dynamic Data Exchange* — динамический обмен данными). Он стандартизирует процесс обмена командами, сообщениями и объектами для обработки между задачами. Наиболее распространенным процессом, для которого используется DDE, является печать.

Другим интерфейсом для обмена данными является OLE (*Object Linking and Embedding* — связывание и встраивание объектов). Этот интерфейс позволяет хранить объекты, созданные одной программой, в объектах, созданных другой программой, а также редактировать (печатать) их без нарушения целостности информации и связей.

Одним из наиболее простых, удобных и интуитивных интерфейсов межпрограммного взаимодействия является буфер обмена — Clipboard. Буфер обмена может содержать в себе один информационный объект — фрагмент текста, рисунок и т. д. С помощью системного вызова процесс может получить копию информации, содержащейся в буфере обмена, или сам поместить объект в буфер, при этом старое содержимое буфера теряется. Таким образом, программы получают простой, но эффективный способ обмена информацией в процессе своей работы.

***Планирование работы процессора.*** Краткосрочный планировщик выбирает процессы из очереди готовых процессов и передает их на выполнение в CPU. Существуют различные алгоритмы или стратегии решения этой задачи, отличающиеся отношением к критериям планирования.

Известны следующие критерии, позволяющие сравнивать алгоритмы краткосрочных планировщиков:

- утилизация CPU (использование процессора). Утилизация CPU теоретически может находиться пределах от 0 до 100 %. В реальных системах утилизация CPU колеблется в пределах 40 % для легко загруженного CPU, 90 % для тяжело загруженного CPU;
- пропускная способность CPU. Пропускная способность CPU может измеряться количеством процессов, которые выполняются в единицу времени;
- время оборота — для некоторых процессов важным критерием является полное время выполнения, то есть интервал от момента появления процесса во входной очереди до момента его завершения. Это время названо временем оборота и включает время ожидания во входной очереди, время ожидания в очереди готовых процессов, время ожидания в очередях к оборудованию, время выполнения в процессоре и время ввода-вывода;
- время ожидания — под этим понимается суммарное время нахождения процесса в очереди готовых процессов;
- время отклика — для интерактивных программ важным показателем является время отклика или время, прошедшее от момента попадания процесса во входную очередь до момента первого обращения к терминалу.

Очевидно, что простейшая стратегия краткосрочного планировщика должна быть направлена на максимизацию средних значений загруженности и пропускной способности, времени ожидания и времени отклика.

В ряде случаев используются сложные критерии, например так называемый минимаксный критерий, то есть вместо простого критерия минимум среднего времени отклика используется *минимум максимального* времени отклика.

***Стратегии планирования процессора***

1. Первый пришел — первый обслуживается FIFO — first come — first served (FCFS). FCFS является наиболее простой стратегией планирования процессов и заключается в том, что процессор передается тому процессу, который раньше всех других его запросил.

Когда процесс попадает в очередь готовых процессов, УТП (pCB) присоединяется к хвосту очереди.

Среднее время ожидания для стратегии PCP5 часто весьма велико и зависит от порядка поступления процессов в очередь готовых процессов.

Стратегии PCP8 присущ так называемый «эффект конвоя». В том случае, когда в компьютере имеется один большой процесс и несколько малых, то все процессы собираются в начале очереди готовых процессов, а затем в очереди к оборудованию. Таким образом «эффект конвоя» приводит к снижению загруженности как процессора, так и периферийного оборудования.

2. Стратегия «наиболее короткая работа выполняется первой» SJF _ Shortest Job First. Одним из методов борьбы с «эффектом конвоя» является стратегия, позволяющая процессу из очереди выполняться первым. Стратегия SJF снижает время ожидания очереди. Наибольшая трудность в практической реализации SJF заключается в невозможности заранее определить величину времени последующего обслуживания.

Поэтому стратегия SJP часто применяется в долгосрочных планировщиках, обслуживающих пакетный режим. В этом случае вместо величины времени последующего обслуживания используется допустимое максимальное время выполнения задания, которое программист должен специфицировать перед отправкой задания в пакет.

3. Приоритетное планирование. Описанные ранее стратегии могут рассматриваться как частные случаи стратегии приоритетного планирования. Эта стратегия предполагает, что каждому процессу приписывается приоритет, определяющий очередность предоставления ему CPU. Например, стратегия PCP8 предполагает, что все процессы имеют одинаковые приоритеты, а стратегия SJF предполагает, что приоритет есть величина, обратная времени последующего обслуживания.

Обычно приоритет — это целое положительное число, находящееся в некотором диапазоне, например от 0 до 7 или от 0 до 1024. Будем считать, что чем меньше значение числа, тем выше приоритет процесса. Приоритеты назначаются, исходя из совокупности внутренних и внешних по отношению к операционной системе факторов.

Внутренние факторы:

- требования к памяти;
- количество открытых файлов;
- отношение среднего времени ввода-вывода к среднему времени Использования ресурсов CPU И так далее.

Внешниефакторы:

- важность процесса;
- тип и величина файлов, используемых для оплаты;
- отделение, выполняющее работы, и так далее.

Внутренние факторы могут использоваться для автоматического назначения приоритетов самой операционной системой, а внешние для принудительного, с помощью оператора.

Главный недостаток приоритетного планирования заключается в *возможности* блокирования на неопределенно долгое время низкоприоритетных процессов.

Известен случай, когда в 1973 году в Массачусетском технологическом институте при остановке компьютера IBM 7094 в очереди готовых процессов были обнаружены процессы, представленные в 1967 году и все еще не выполненные.

Для устранения отмеченного недостатка используются следующие методы: процессы, время ожидания которых превышает фиксированную величину, например 15 минут, автоматически получают единичное приращение приоритета.

4. «Карусельная» стратегия планирования RR-Round KoЫп — применяется в системах разделения времени. Определяется небольшой отрезок времени $t_k$, названный квантом времени (10...100 мс). Очередь готовых процессов рассматривается как кольцевая. Процессы циклически перемещаются по очереди, получая CPU на время, равное одному кванту. Новый процесс добавляется в хвост очереди. Если процесс не завершился в пределах выделенного ему кванта времени, его работа принудительно прерывается, и он перемещается в хвост, очереди.

Свойства стратегии Round Robin сильно зависят от величины временного кванта $t_k$. Чем больше временной квант, тем ближе стратегия Коипс! KoЫп приближается к PCP5 стратегию. При очень малых значениях временного кванта Коипс! KoЫп стратегию называют разделением процессора — processor snaring. Теоретически это означает, что каждый из $N$ процессов работает со своим собственным процессором, производительность процессора равна $1/N$ от производительности физического процессора.

5. Планирование с использованием многоуровневой очереди (Multilevel queue scheduling). Эта стратегия разработана для ситуации, когда процессы могут быть легко классифицированы на несколько групп, например часто процессы разделяют на две группы: интерактивные (процессы переднего плана) и пакетные (фоновые).

Интерактивные и пакетные процессы имеют различные требования к краткосрочному планировщику, например по отношению ко времени отклика.

Стратегия многоуровневой очереди разделяет очередь готовых процессов на несколько очередей, в каждой из которых находятся процессы с одинаковыми свойствами, и каждый из которых может планироваться индивидуальной стратегией, например Round Robin стратегия для интерактивных процессов и FCFS для пакетных процессов.

Взаимодействие очередей осуществляется по следующим правилам: ни один процесс с более низким приоритетом не может быть запущен, пока не выполнятся процессы во всех очередях с более высоким приоритетом.

Работа процесса из очереди с более низким приоритетом может быть приостановлена, если в одной из очередей с более высоким приоритетом появился процесс.

6. Использование многоуровневой очереди с обратными связями (multilevel feedback queue sheduling). Обычная многоуровневая очередь не допускает перемещения процессов между очередями. Многоуровневая очередь с обратными связями предполагает, что процессы при определенных условиях могут перемещаться между очередями. Здесь организуется $N$ очередей. Все новые запросы поступают в конец первой очереди. Первый запрос из $i$-й очереди поступает на обслуживание лишь тогда, когда все очереди от 1-й до $i$ — 1-й пустые. На обслуживание выделяется квант времени $t_k$. Если за это время обслуживание запроса завершается полностью, то он покидает систему. В противном случае недообслуженный запрос поступает в конец $i$ + 1-й очереди.

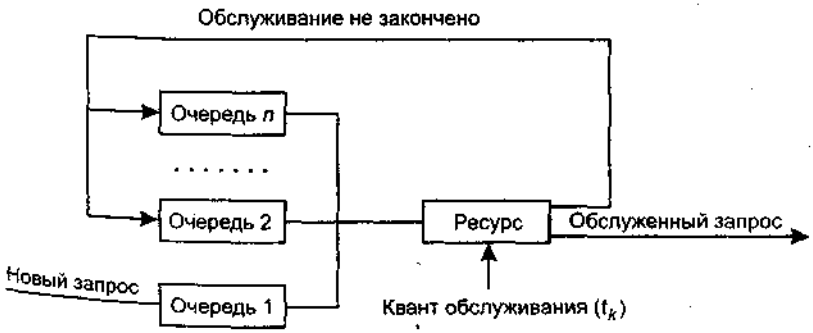



Рис. 1.11. Многоуровневая очередь с обратными связями

После обслуживания запроса из $i$-й очереди система выбирает для обслуживания запрос из непустой очереди с самым младшим номером. Таким запросом может быть следующий запрос из очереди *1* или из очереди $i + 1$ (при условии, что после обслуживания запроса из очереди $i$ последняя оказалась пустой). Новый запрос поступает в 1-ю очередь ($i = 1$). В такой ситуации после окончания времени $t_k$, выделенного для обслуживания запроса из очереди $i$, будет начато обслуживание запроса первой очереди. Если система выходит на обслуживание заявок из $N$-й очереди, то они обслуживаются либо по дисциплине FIFO (каждая заявка обслуживается до конца), либо по циклическому алгоритму. Данная система наиболее быстро обслуживает все короткие по времени обслуживания запросы. Недостаток системы заключается в затратах времени на перемещение запросов из одной очереди в другую.

Данная стратегия является универсальной и сочетает в себе свойства всех рассмотренных раньше стратегий — FCFS, SJF, приоритетная, Round Robin, многоуровневая очередь.

7. Приоритетная многоочередная дисциплина обслуживания (рис. 1.12). Вновь поступающие в систему запросы устанавливаются не обязательно в 1-ю очередь, а в очередь в соответствии с имеющимися приоритетами, которые определяются параметрами обслуживания процессов. Приоритетные многоочередные дисциплины обслуживания могут использовать обслуживание с абсолютным и относительным приоритетом. При обслуживании с абсолютным приоритетом приоритет определяется номером очереди, и первыми обслуживаются запросы, обладающие наивысшим приоритетом (из очереди с меньшим номером запрос из очереди $i - 1$ будет прерывать обработку запроса из очереди $i$).

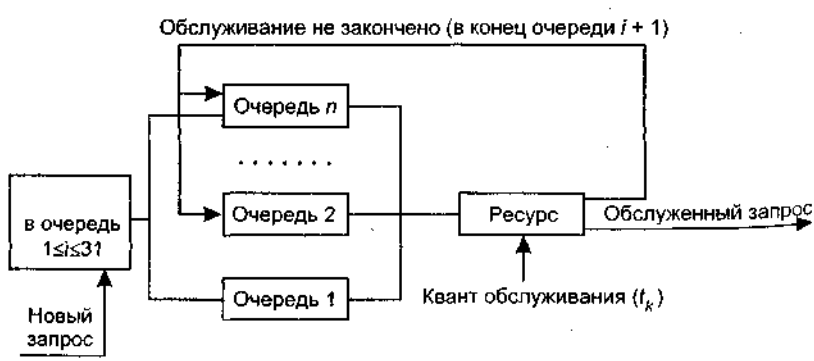



Рис. 1.12. Многоочередная с приоритетами система обслуживания

В данной дисциплине еще более увеличивается степень дискриминации по среднему времени ожидания в очереди между высоко- и низкоприоритетными запросами. Время ожидания высокоприоритетных заявок сокращается, но за счет большей задержки в обслуживании низкоприоритетных заявок. Достигается это за счет усложнения логики системы, дополнительной обработки запросов и выбора правила дообслуживания прерываемых процессов. Обслуживание с относительным приоритетом не вызывает прерывания обслуживаемой заявки до ее завершения, даже если она менее приоритетная.

### *Управление* ***невиртуальной*** *памятью*

1. Свопинг (swapping). Свопингом (перекачкой) называется метод управления памятью, основанный на том, что все процессы, участвующие в мультипрограммной обработке, хранятся во внешней памяти.

Процесс, которому выделен CPU, временно *перемещается в основную память* (swap in/roll т). В случае прерывания работы процесса он *перемещается обратно во внешнюю память* (swap out/roll out).

***Замечание:*** *при свопинге из основной памяти во внешнюю (и обратно) перемещается вся программа, а не ее отдельная часть.*

Свопинг иногда используют при приоритетном планировании CPU. В этом случае с целью освобождения памяти для высокоприоритетных процессов, низкоприоритетные процессы перемещаются во внешнюю память.

Основное применение свопинг находит в системах разделения времени, где он используется одновременно со стратегией Round Robin планирования СРП

В начале каждого временного кванта блок управления памятью выгружает из основной памяти процесс, работа которого была только что прервана, и загружает очередной выполненный процесс. Метод свопинга влияет на величину временного кванта стратегии Round Robin.

Пример. Пусть очередной загружаемый в память процесс имеет размер 100 Кбайт. Диск позволяет читать данные со скоростью 1 Мбайт в секунду, следовательно, 100 Кбайт могут быть загружены за 100 мс. Будем считать, что для первоначального подвода головки чтения-записи потребуется 8 мс. Таким образом, операция свопинг займет 108 мс, а общее время свопинга — 216 мс.

Для эффективной загруженности процессора время свопинга должно быть существенно меньше времени счета. Следовательно, для рассмотренного примера квант времени должен быть существен-

но больше, чем 216 мс. Ясно, что это число значительно увеличится если перемещаемый процесс имеет размер, например, 1 Мбайт.

Недостаток «чистого» свопинга заключается в больших потерях времени на загрузку или выгрузку процессов. Поэтому в современных операционных системах используются модифицированные варианты свопинга.

Так, например, во многих версиях операционной системы UNIX свопинг включается только в том случае, когда количество процессов в памяти становится слишком большим.

2. Смежное размещение процессов. Методы размещения процессов в основной памяти по отношению к расположению участков памяти, выделенных для одной и той же программы, делят на два класса. Первый — *метод смежного размещения,* второй — *метод несмежного размещения.*

Смежное размещение является простейшим и предполагает, что в памяти, начиная с некоторого начального адреса, выделяется один непрерывный участок адресного пространства.

При несмежном размещении программа разбивается на множество частей, которые располагаются в различных, необязательно смежных участках адресного пространства.

*Однопрограммный режим.* Рис. 1.13 иллюстрирует смежное размещение одной программы в основной памяти.

При смежном размещении размер загружаемой программы ограничивается размером ОЗУ. Для того чтобы при смежном размещении загружать программы, размеры которых превышают размеры ОЗУ, используют метод оверлейных сегментов (overlay segments).

В программе, имеющей древовидную структуру, модули второго уровня работают сугубо последовательно, поэтому в памяти может находиться только один из них.

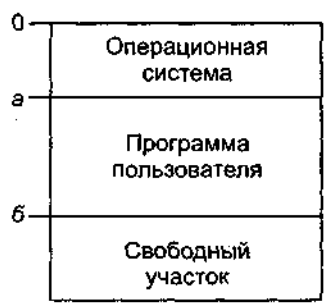



Рис. 1.13. Однопрограммный режим

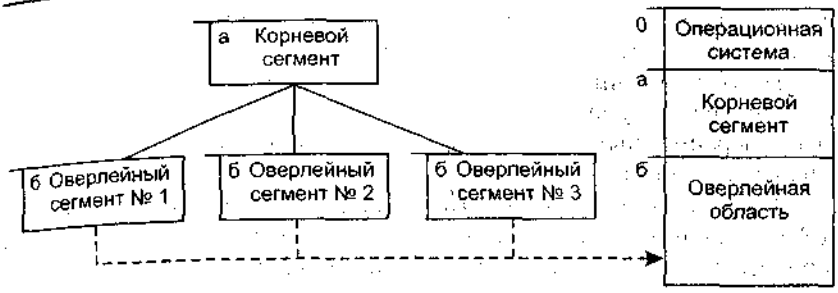



Рис. 1.14. Оверлейные структуры программ

Оверлейную структуру программы и последовательность загрузки оверлейных сегментов планирует сам программист.

В процессе выполнения программы все ее адреса не должны быть меньше числа а. В противном случае возможна Запись какого-либо результата работы программы (поверх операционной системы) и уничтожение некоторых ее частей. Защиту операционной системы в случае смежного размещения при однопрограммном режиме можно осуществить с помощью регистра границы (рис. 1.15).

Во время работы прикладной программы все адреса, генерируемые CPU, сравниваются с содержимым регистра границы. Если генерируется адрес меньше числа *а,* работа программы прерывается.

Мультипрограммирование с фиксированными разделами (MFT — Multiprogramming with a fixed number of tasks) предполагает разделение адресного пространства на ряд разделов фиксированного раздела. В каждом разделе размещается один процесс (рис. 1.16).

В этом случае, если соответствующий адресам процесса раздел занят, процесс остается в очереди во внешней памяти даже в том случае, когда другие разделы свободны.

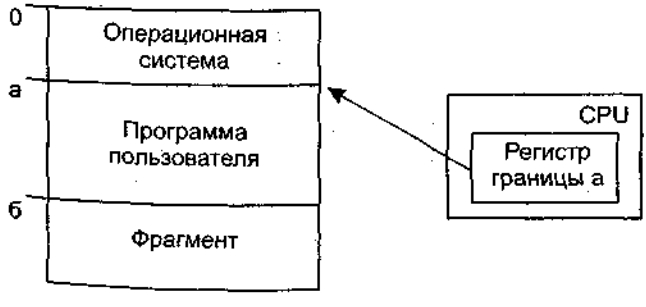



Рис. 1.15. Регистр границы

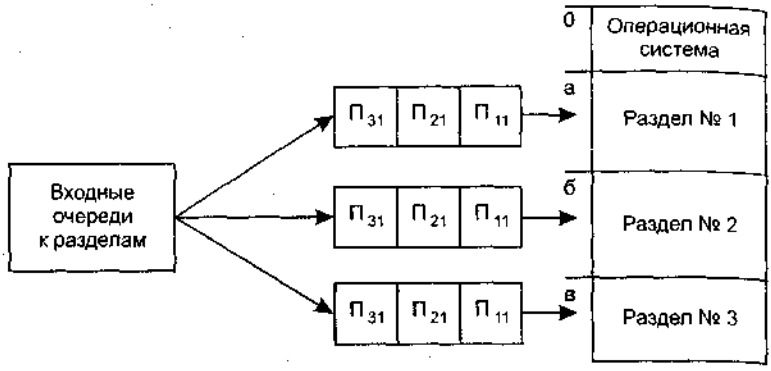



Рис. 1.16. Режим мультипрограммирования с фиксированным количеством разделов (задач)

Уменьшить фрагментацию памяти при мультипрограммировании с фиксированными разделами можно, если загрузочные модули создаются в перемещаемых адресах. Такой модуль может быть загружен в любой свободный раздел после соответствующей настройки.

При мультипрограммировании с трансляцией в перемещаемых адресах имеются две причины фрагментации. Первая — размер загруженного процесса меньше размера, занимаемого разделом (внутренняя фрагментация), вторая — размер процесса в очереди больше размера свободного раздела, и этот раздел остается свободным (внешняя фрагментация).

Для защиты памяти при мультипрограммировании с фиксированным количеством разделов необходимы два регистра. Первый -

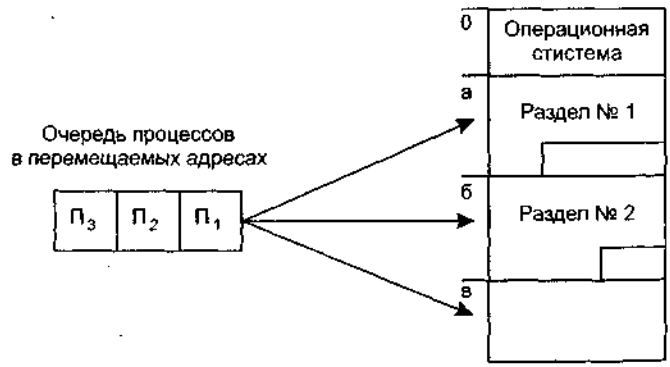



Рис. 1.17. Режим мультипрограммирования с фиксированным количеством разделов (фрагментация памяти).

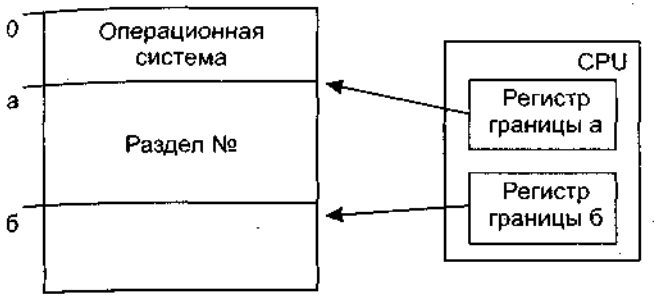



Рис. 1.18. Регистры границ для MFT

регистр верхней границы (наименьший адрес), второй — регистр нижней границы (наибольший адрес).

Прежде чем программа в разделе N начнет выполняться, ее граничные адреса загружаются в соответствующие регистры. В процессе работы программы все формируемые ею адреса контролируются на удовлетворение неравенства $a < Адр. < б$.

При выходе любого адреса программы за отведенные ей границы работа программы прерывается.

Мультипрограммирование с переменными разделами (multiprogramming with a variable number of tasks — MVT) предполагает разделение памяти на разделы и использование загрузочных модулей в перемещаемых адресах, однако границы разделов не фиксируются.

В начальной фазе отсутствует фрагментация, связанная с тем, что размер очередного процесса меньше размера, занимаемого этим процессом раздела. На этой фазе причиной фрагментации является несоответствие размера очередного процесса и оставшегося участка памяти. По мере завершения работы программы освобождаются отдельные разделы. В том случае, когда освобождаются смежные разделы, границы между ними удаляются и разделы объединяются.

За счет объединения или слияния смежных разделов образуются большие фрагменты, в которых можно разместить большие программы из очереди. Таким образом, на фазе повторного размещения действуют те же причины фрагментации, что и для метода MFT.

Мультипрограммирование с переменными разделами и уплотнением памяти. Ясно, что этот метод может создать ситуацию, когда в памяти образуется множество малых фрагментов, каждый из которых может быть недостаточен для размещения очередного процесса, но суммарный размер фрагментов превышает размер этого процесса.

Уплотнением памяти называется перемещение всех заняты разделов по адресному пространству памяти таким образом, чтоб свободный фрагмент занимал одну связную область (рис. 1.19).

На практике реализация уплотнения памяти сопряжена с усложнением операционной системы и обладает следующими недостатками:

- в тех случаях, когда мультипрограммная смесь Неоднородна по отношению к размерам программ, возникает необходимость в частом уплотнении, что *расходует ресурс процессорного времени и компенсирует экономию ресурса памяти*;
- во время уплотнения все прикладные программы переводятся в состояние «ожидание», что приводит к невозможности выполнения программ в реальном масштабе времени.

*Основные стратегии заполнения свободного раздела.* Рассмотренные методы мультипрограммирования предполагают наличие входной очереди/очередей к разделам основной памяти.

В том случае; когда освобождается очередной раздел, операционная система должна выбрать один из процессов для размещения его в памяти. Алгоритм выбора может использовать одну из следующих трех стратегий:

- стратегия наиболее подходящего — выбирает процесс, которому в освободившемся разделе наиболее тесно (выигрыш в памяти);
- стратегия первого подходящего — выбирает первый процесс, который может разместить в освободившемся разделе;
- стратегия наименее подходящего — выбирает процесс, которому в освободившемся разделе наиболее свободно (в этом случае остающийся фрагмент часто достаточен для размещения еще одного процесса).

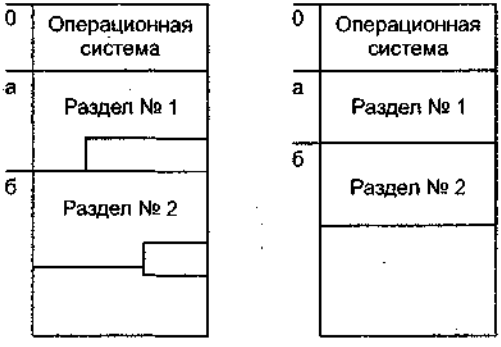



Рис. 1.19. Режим уплотнения памяти

***Страничная организация памяти*** (paging) относится к методам несмежного размещения процессов в основной памяти.

Основное достоинство страничной организации памяти заключается в том, что она позволяет свести к минимуму общую фрагментацию за счет полного устранения внешней фрагментации и минимизации внутренней фрагментации.

*Базовый метод.* Адресное пространство основной и внешней памяти разбивают на блоки фиксированного размера, называемые *страничные рамки* (frames). Логическое адресное пространство программы также разбивается на блоки фиксированного размера, называемые *страницами* (pages). Размеры страничных рамок и страниц совпадают. Процесс загружается в память постранично, причем каждая страница помещается в любую свободную страничную рамку основной памяти.

Каждый адрес, генерируемый процессором, состоит из двух частей: П — номер страницы (page number) и Д — смещение в пределах страницы (offset). Номер страницы может использоваться как индекс для таблицы страниц (page table).

Таблица страниц содержит начальные адреса *f* всех страничных рамок, в которых размещена программа. Физический адрес определяется путем сложения начального адреса страничной рамки *f* и смещения Д.

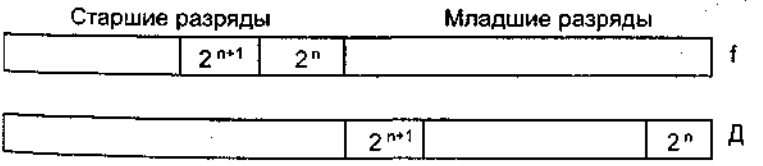



Рис. 1.20. Структура адреса при страничной организации

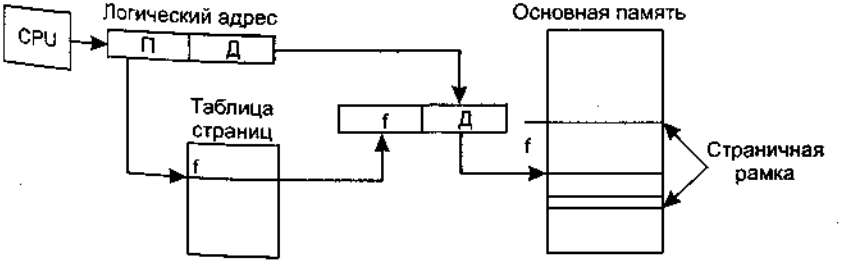



Рис. 1.21. Генерация физического адреса

Рисунок 1.22 показывает, что страничная организация памяти полностью исключает внешнюю фрагментацию. Внутренняя фрагментация не превышает величины **page_size-QElem**, где **page_size** — размер страничной рамки, а **QElem** — минимальный адресуемый элемент основной памяти.

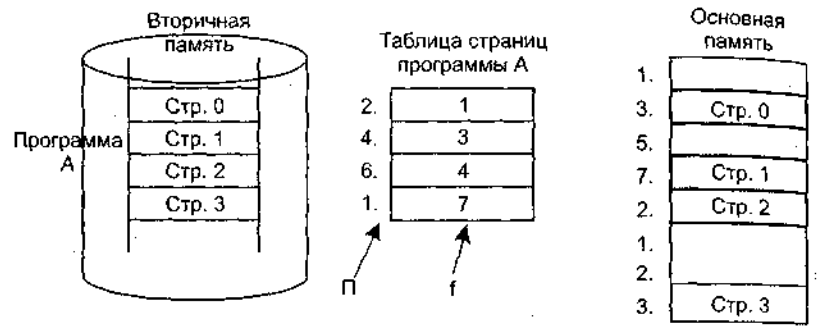



**Рис. 1.22. Размещение содержания страниц на внешней памяти**

Для ускорения вычисления физического адреса операцию суммирования заменяют операцией конкатенации.

На рисунке заштрихованы незаполненные нулевые разряды. Для того чтобы операция конкатенации была возможна, необходимо, чтобы базовые адреса страничных рамок располагались только в старших разрядах ($2^{n+1}$), а следующие — только в младших разрядах ($2^0$, $2^1$, $2^2$).

Например, при $n = 9$ базовые адреса страничных рамок — это следующий ряд: 512, 1024, 1536. Следовательно, размер страничной рамки равен 512 байт. В современных операционных системах типичный размер страницы составляет 2 Кбайт или 4 Кбайт.

Каждая операционная система поддерживает свой собственный метод работы с таблицей страниц. Обычно за каждым процессом, находящимся в основной памяти, закреплена отдельная таблица страниц. В этом случае указатель на таблицу страниц хранится в PCB соответствующего процесса.

*Аппаратная поддержка страничной организации памяти.* Преобразование логического адреса в физические осуществляется для каждого адреса, генерируемого процессором, поэтому часто для ускорения этого процесса применяются аппаратные методы, например ассоциативные регистры (associative registers).

Каждый ассоциативный регистр, кроме операций чтения-записи, может обрабатывать операцию сравнения кода, поступающего

на ... вход с частью кода, хранимого в регистре. Матрица ассоциа-
ти ... ых регистров хранит часть таблицы страниц. Номер страницы
П ..одается одновременно на входы всех ассоциативных регистров,
ко... параллельно выполняют операцию сравнения. На выходе
... ицы ассоциативных регистров образуется начальный адрес
стр...аничной рамки f того регистра, в котором произошло совпадение
кода.

В том случае, если требуемый номер страницы находится в таблице страниц, то есть ни в одном из ассоциативных регистров не произошло совпадения, происходит обращение к таблице страниц, находится искомый номер страничной рамки, а найденная строка таблицы страниц переписывается в один из ассоциативных регистров.

Зашита страничной памяти основана на контроле уровня доступа к каждой странице, возможны следующие уровни доступа:

. только чтение;
- чтение и запись;
- только выполнение.

В этом случае каждая страница снабжается 3-битовым кодом уровня доступа. При трансформации логического адреса в физический сравнивается значение кода разрешенного уровня доступа с фактически требуемым. При их несовпадении работа программы прерывается.

***Управление виртуальной памятью.*** Все методы управления памятью имеют одну и ту же цель — хранить в памяти мультипрограммную смесь, необходимую для мультипрограммирования. Рассмотренные ранее методы предполагали, что вся программа перед выполнением должна быть размешена в основной памяти. *Виртуальная память* — это технология, которая позволяет выполнять процесс, который может только частично располагаться в основной памяти. Таким образом, виртуальная память позволяет выполнять программы, размеры которых превышают размеры физического адресного пространства.

*Перемещение страниц по запросу (demand paging).* Виртуальная память чаще всего реализуется на базе страничной организации памяти, совмещенной со свопингом страниц.

Свопингу подвергаются только те страницы, которые необходимы процессору. Таким образом, перемещение страниц по запросу озн...ает:

- программа может выполняться CPU, когда часть страниц находится в основной памяти, а часть — во внешней;

- в процессе выполнения новая страница не перемещается в основную память до тех пор, пока в ней не возникла необходимость.

Для учета распределения страниц между внешней и основной памятью каждая строка таблицы страниц дополняется битом местонахождения страницы (**valid/invalid bit**).

В том случае, если процессор пытается использовать страницу, помеченную значением **invalid**, возникает событие, называемое *страничная недостаточность* (**paging fault**).

Страничная недостаточность вызывает прерывание выполнения программы и передачу управления операционной системе. Реакция операционной системы на страничную недостаточность заключается в том, что необходимая страница загружается в основную память.

Основные этапы обработки страничной недостаточности (рис. 1.23):

1. Процессор, прежде чем осуществлять преобразование логического адреса в физический, проверяет значение бита местонахождения необходимой страницы.

2. Если значение бита **invalid,** то процесс прерывается и управление передается операционной системе для обработки события страничная недостаточность.

3. Отыскивается необходимая страница во вторичной памяти и свободная страничная рамка в основной.

4. Требуемая страница загружается в выбранную страничную рамку.

5. После завершения операции загрузки редактируется соответствующая строка таблицы страниц, в которую вносится базовый адрес и значение бита местонахождения — **valid**.

6. Управление передается прерванному процессу.

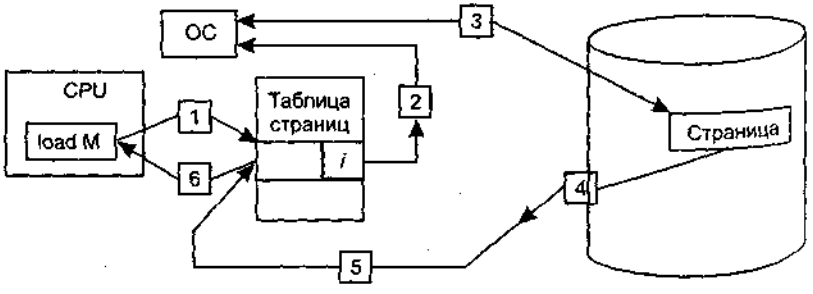



Рис. 1.23. Обработка страничной недостаточности

Метод обмена страниц по запросу позволяет начать выполнение процесса даже в том случае, когда ни одна страница этого процесса не загружена в основную память.

Вторичная память, используемая при обмене страниц по запросу, — это высокоскоростное дисковое устройство, часто называемое *устройством* свопинга (swap device), а часть используемого дискового пространства — пространство свопинга (swap space).

*Замещение страниц.* В процессе обработки страничной недостаточности операционная система может обнаружить, что все страничные рамки основной памяти заняты и, следовательно, невозможно загрузить требуемую страницу. В этом случае возможны следующие режимы: приостановка прерванного процесса, уменьшение на единицу количества процессов мультипрограммной смеси для освобождения всех ею занимаемых страничных рамок, использование метода замещения страниц.

Метод замещения страниц состоит в том, что в основной памяти выбирается наименее важная/используемая страница, называется *страница-жертва* (victim page), которая временно перемещается в пространство свопинга, а на ее место загружается страница, вызываемая страничной недостаточностью.

Обработка страничной недостаточности с учетом замещения осуществляется по следующему алгоритму:

- определяется местонахождение страницы путем анализа бита местонахождения;
- если значение бита ***invalid***, то разыскивается свободная страничная рамка;
- если имеется свободная страничная рамка, то она используется;
- если свободной страничной рамки нет, то используется алгоритм замещения, который выбирает страницу-жертву;
- страница-жертва перемещается в пространство свопинга и таблица страниц редактируется;
- требуемая страница загружается на место страницы-жертвы и соответствующим образом редактируется таблица страниц.

Управление передается прерванному процессу. Приведенный алгоритм замещения требует двухстраничных перемещений:

- страница-жертва перемещается в пространство свопинга;
- требуемая страница перемещается в освободившуюся страничную рамку.

Страницу-жертву можно не копировать в пространство свопинга в том случае, если за время, прошедшее от последнего перемещения ее содержимое не модифицировалось. В этом случае время замещения уменьшается примерно вдвое.

Для учета факта модификации страницы в таблицу страниц вводится дополнительный бит, который меняет свое значение на противоположное в том случае, если содержимое страницы изменилось.

Для практического использования метода обмена страниц по запросу необходимы два алгоритма:

- алгоритм распределения страничных рамок (frame allocation algorithm);
- алгоритм замещения страниц (page replacement algorithm).

***Алгоритм распределения страничных рамок.*** Алгоритм распределения страничных рамок решает, сколько страничных рамок в основной памяти выделить каждому из процессов мультипрограммной смеси. Алгоритм замещения страниц решает, какую из страниц выбрать в качестве жертвы.

*FIFO (first-in-first-out).* Наиболее простым алгоритмом замещения страниц является алгоритм PIPO. Этот алгоритм ассоциирует с каждой страницей время, когда эта страница была помещена в память. Для замещения выбирается наиболее старая страница.

Учет времени необязателен, когда все страницы в памяти связаны в FIFO-очередь, а каждая помещаемая в память страница добавляется в хвост очереди.

Алгоритм учитывает только время нахождения страницы в памяти, но не учитывает используемость страницы. Например, первые страницы программы могут содержать переменные, используемые на протяжении работы всей программы. Это приводит к немедленному возвращению к только что замещенной странице.

*Оптимальный алгоритм.* Этот алгоритм имеет наилучшее соотношение количества замещенных страниц к количеству ссылок. Алгоритм строится по следующему принципу: замещается та страница, на которую нет ссылки на протяжении наиболее длительного периода времени. Для реализации этого алгоритма необходимо каждый раз сканировать весь поток ссылок, поэтому он нереализуем на практике и используется для оценки реально работающих алгоритмов.

*Алгоритм LRU (least recently used).* Алгоритм выбирает для замещения ту страницу, на которую не было ссылки на протяжении наиболее длинного периода времени. Он ассоциирует с каждой страницей время последнего использования этой страницы. Для замещения выбирается та страница, которая дольше всех не использовалась. Обычно применяются два подхода при внедрении этого алгоритма:

- *подход на основе логических часов (счетчика)* — ассоциируют с каждой строкой таблицы поле «время использования», а в CPU

добавляются логические часы. Логические часы увеличивают свое значение при каждом обращении к памяти. Каждый раз, когда осуществляется ссылка на страницу, значение регистра логических часов копируется в поле «время использования». Заменяется страница с наименьшим значением в отмеченном поле путем сканирования всей таблицы страниц. Сканирование отсутствует при использовании подхода на основе стека;

- *подход на основе стека номеров страниц* — стек номеров страниц хранит номера страниц, упорядоченных в соответствии с историей их использования, *на «вершине» стека* располагается только что использованная страница, а *на «дне»* дольше всех не используемая страница. Как только осуществляется ссылка на страницу, она перемещается на вершину стека, а номера всех страниц сдвигаются вниз.

## 1.4. Связь с оператором

В общем случае, конечно, следует говорить о *связи с внешней средой,* поскольку, например, при использовании ЭВМ в системах управления технологическими комплексами (производство, летательные аппараты, корабли и пр.) человек может быть исключен (полностью или частично) из контура управления и внешними устройствами ЭВМ будут *датчики* (скорости, высоты, давления, температуры) и *эффекторы* (приводы рулей, манипуляторы, сервомоторы вентилей и пр.).

*Связь с пользователем,* сокращенно поименованная здесь как связь с оператором, — как говорят англичане, last but not least — последняя в списке, но не по важности функция ОС.

Типология связи с человеком определяется как уровнем развития программного обеспечения, так и техническими средствами. Как это следует из рис. 1.1, связь с пользователем включает:

- командный (или иной) интерфейс по управлению системными процессами в вычислительной системе (собственно функции оператора ОС). Пользователь (привилегированный) осуществляет запуск-останов программ, подключение-отключение устройств и прочие релевантные операции;
- интерфейс по управлению пользовательскими процессами (контроль состояния процесса, ввод-вывод данных в процесс / из процесса).

В состав *пользователей* в общем случае включаются следующие группы лиц, контактирующих с системой:

- администратор системы: лицо или группа, отвечающая за сопровождение данных, назначение уровней доступа, включение/исключение пользователей;
- оператор системы, осуществляющий сопровождение вычислительного процесса;
- прочие пользователи (не обладающие привилегиями доступа к данным), в том числе:
  - √ операторы подготовки данных (ОПД) — персонал, осуществляющий ввод данных с рабочих листов или документов, на основе соответствующих инструкций, в среде специальных программных интерфейсов;
  - √ интерактивные пользователи (ИП) — лица, имеющие доступ на ввод, коррекцию, обновление, уничтожение и чтение данных в рамках, как правило, ограниченной области БД;
  - √ конечные пользователи (КП) — лица, использующие БД для получения справок и решения задач.

Очевидно, что именно *оператор ЭВМ является естественным пользователем ОС;* все же прочие пользователи становятся таковыми лишь вследствие расширения функций пользователя в связи с интеграцией (особенно в случае персональных ЭВМ) функций конечного пользователя, администратора системы и оператора.

Интерфейс — это способ общения пользователя с персональным компьютером, пользователя с прикладными программами и программ между собой. Интерфейс служит для удобства управления программным обеспечением компьютера. Интерфейсы бывают однозадачные и многозадачные, однопользовательские и многопользовательские. Интерфейсы отличаются между собой по удобству управления программным обеспечением, то есть по способу запуска программ. Существуют универсальные интерфейсы, допускающие все способы запуска программ, например Windows 3.1, Windows 95. Например, Windows 95 позволяет реализовать несколько способов запуска программ, в том числе позволяет запускать программы при помощи меню кнопки Пуск.

***Разновидности интерфейсов.*** Интерфейсы отличаются по способу доступа к командным файлам программ.

*Командный (текстовый) интерфейс.* Всякая операционная система имеет командный интерфейс (иногда в скрытой форме).

Если снять шелуху текстовых или графических оболочек или интерфейсов, то «на глубине» вы *всегда* найдете командный интерфейс.

В первой из ОС (OS/360) взаимодействие с пользователями было жестко поделено между следующими компонентами:

- *командный язык оператора ЭВМ* (лицо, ответственное за управление вычислительным процессом). Это *язык диалогового режима* — команды запуска-остановки задач, привязки носителей информации к устройствам, получения информации о заданиях, ожидающих выполнения, вывода, наличии свободной памяти и свободных устройств и др.;
- *язык управления заданиями* (JCL — Job Control Language), на котором прочие пользователи (программисты, разработчики и просто конечные пользователи) описывали состав и структуру процесса обработки данных — последовательность запуска программ, входные и выходные файлы, условия, при которых те или иные программы должны быть выполнены или пропущены и др. Это *язык пакетной обработки, не* допускающий вмешательства пользователя в собственно процесс вычислений, компиляции и пр.

По мере развития ЭВМ, ОС, появления и широкого распространения диалоговых устройств (видеотерминалов) в последующих ОС произошла интеграция данных компонент в единый командный язык. Для разграничения между командами оператора, администратора, конечного пользователя используются методы разделения доступа и назначения привилегий, в то время как формат команд является достаточно единообразным.

Далее, после распространения ПЭВМ данное разграничение сошло на нет (в ОС MS-DOS), поскольку пользователь ПК в едином лице соединяет функции оператора, администратора, конечного пользователя. Затем с появлением локальных сетей и более мощных ПК, Работающих в многопользовательских режимах, в сетевых ОС и ОС ПЭВМ, вновь организуется разграничение доступа и т. д. Таким образом, данный процесс является циклическим (точнее, спиралевидным).

В табл. 1.3 приведена выборка из основных функциональных групп команд различных ОС.

В большинстве ОС в настоящее время сложился более или менее унифицированный *формат командной строки.* Командная строка включает в себя (рис. 1.24, табл. 1.3, а):

- тип операции (имя команды или программы);
- рабочий вход (входные файлы или устройства);
- Рабочий выход (выходные файлы или устройства);
- Управляющий вход (управляющие параметры или ключи команды);
- Управляющий выход (обычно — протокол, содержащий диагностику ошибок, код завершения или другую информацию).

*Таблица 1.3.* Сравнительный анализ некоторых команд различных ОС

| Команда (функция) | OS/360/370 | RSX-11/20 | MS-DOS | UNIX |
|---|---|---|---|---|
| **1. Спецификация устройств, файлов и пр. в разных ОС** | | | | |
| Обозначения устройств (НМД) | UNIT=SYSDA, UNIT=191, 192 | DKx: DPx:... | A:, B:, C: | ROOT (корневой каталог ФС) |
| Устройства печати | SYSOUT=A | LPx: | LPTx: | lpr, lp |
| Терминалы | UNIT=050, 0C0 | TTx: | CON | TTYx: |
| Спецификация файла | DSN, VOL, DCB, SPACE | Устр.:[G,N]имя.тип; версия | Устр:\путь\имя.тип | /путь/имя.тип |
| Спецификация группы файлов | Партиционный файл (НД) (DSORG=PO) Сцепленные НД | Использование масок «*», «?» | Использование масок «*», «?» | Использование масок «*»,,"?" |
| **2. Подготовка носителей к использованию, работа с каталогами** | | | | |
| Инициализация диска, создание файловых систем | INIT | INI, FMT | FDISK, FORMAT | mkfs |
| Создание каталогов в ФС | DISP=NEW | UFD | MKDIR | mkdir |
| Проверка диска, файловой системы | DASDR/CHK | BAD/CHK | CHKDISK | fchk |
| Проверка / установка файловых атрибутов | DISP=SHR, OLD, NEW | – | ATTRIB | file, chmod |
| Удаление каталога | DISP=DELETE | PIP/DE | RD | rmdir |
| **3. Работа с файлами** | | | | |
| Копировать файл | DISP=NEW, DSNAME=NEWFILE | PIP A:=B | COPY | cp |
| Переместить файл | DISP=(OLD, DELETE) | PIP A:=B | MOVE | mv |
| Переименовать файл | DISP=NEW, DSN=NEWFILE | PIP A:=B | REN, RENAME | mv |

*Продолжение табл. 1.3*

| Команда (функция) | OS/360/370 | RSX-11/20 | MS-DOS | UNIX |
|---|---|---|---|---|
| Удалить файл | DISP=DELETE | PIP/DE | DEL, ERASE | rm |
| Поиск в файле по контексту | – | – | FIND | grep, awk |
| Архивация файла на МД или МЛ | DASDR/DUMP-RESTORE | BRU (Backup-restore utility) | BACKUP, RESTORE | TAR |
| Распечатка содержимого файла | IEBGENER-PRINT | PIP LP:=... | TYPE, COPY | cat |
| Сравнение файлов | – | – | FC | comm (Common)<br>cmp (Compare)<br>diff (difference) |
| **4. Некоторые информационные команды** | | | | |
| Текущий диск, каталог | D U (используемые устройства) | UIC | PATH | pwd |
| Содержание диска, каталога | VTOC | PIP/LI | DIR | ll |
| Активные задачи, пользователи | DA (Display Active) | – | – | who |
| **5. Некоторые команды управления процессами** | | | | |
| Начало работы пользователя (задания) | //AAA JOB | HEL (hello) | – | login |
| Окончание работы пользователя | // | BYE | – | <Ctrl+D>, ^D |
| Запуск последовательности программ | JCL-операторы<br>TSO-команды | Командный файл .CMD | Пакетный файл .BAT | Пакетный файл Shell |
| Запуск задачи, программы, процесса | S (Start)<br>E (Exec) | RUN | Имя файла (пакетного .BAT или исполняемого .COM, .EXE) | Имя файла |
| Остановка программы | C (Cancel)<br>P (Stop) | ABO (Abort) | <Ctrl+C>, <Ctrk+Break>, ^C | ^C |


Рис. 1.24. Типовая структура командной строки языков операционных систем

*Таблица 1.3, а.* **Примеры командных строк и их форматов**

| Команда | Рабочий вход | Рабочий выход | Управляющий вход | Информационный выход |
|---|---|---|---|---|
| ОС MS-DOS<br>copy \test.txt a: >prn | Файл test.txt | Копия файла на дисководе a: | Режим по умолчанию | Протокол на устройстве печати (prn) |
| ОС MS-DOS<br>format a: /v >prn | Исходная дискета на устройстве a: | Форматированная дискета a: | Ключ команды /v (запросить метку диска) | Протокол форматирования на устройстве prn |
| ОС RSX<br>BCD LP:=MT00 | Магнитная лента на устройстве MT0: | Нет | Нет (режим по умолчанию) | Содержание ленты на устройстве LP: (печать) |
| ОС UNIX<br>ls -l /bin | Каталог /bin | Нет | Ключ -l (распечатка в полном формате) | Содержание каталога на экране терминала |

*Текстовый или графический полноэкранный интерфейс.* Он имеет, как правило, в верхней части экрана систему меню с подсказками. Меню часто бывает выпадающим (ниспадающим — pull-down).

Для управления компьютером курсор экрана или курсор мыши после поиска в дереве каталогов устанавливается на командные файлы программ (*.exe, *.com, *.bat) и для запуска программы нажимается клавиша <Enter> или правая кнопка мыши. Различные файлы могут выделяться разным цветом или иметь разный рисунок. Каталоги (папки) отличаются от файлов размером или рисунком. Данный интерфейс является основным для всех видов программных оболочек. Пример: Norton Commander и нортонообразные оболочки (DOS Navigator, Windows Commander, Disk Commander). Подобный интерфейс имеют инструменты Windows 3.1 (Диспетчер файлов) и Windows-95 (Мой компьютер и Проводник). Такой интерфейс весьма удобен, особенно при работе с файлами, поскольку обеспечивает высокую скорость выполнения операций, позволяет создавать поль-

зовательское меню, запускать приложения по расширению файлов, что повышает скорость работы с программами.

*Графический многооконный пиктографический интерфейс.* Представляет собой рабочий стол (DeskTop), на котором располагаются пиктограммы (значки или иконки программ). Все операции производятся как правило, мышью. Для управления компьютером курсор мыши подводят к пиктограмме и запуск программы осуществляют щелчком левой кнопки мыши по пиктограмме. Это наиболее удобный и перспективный интерфейс, особенно при работе с программами Примеры: интерфейс компьютеров Apple Macintosh, Windows ЗЛ, Windows 95/98, О5/2, X Windows.

***Терминалы.*** Терминалы, или *терминальные устройства, ЭВМ* являются важнейшей компонентой систем", основанных на человеко-машинном взаимодействии. Это *диалоговые* или *интерактивные* устройства, предназначенные для ввода/вывода небольших количеств информации, первоначально с целью *управления* вычислительным процессом и наблюдения за его ходом, а в дальнейшем — также для ввода-вывода *исходных* данных и *результатов* работы программ. Первоначально в ЭВМ использовались в качестве терминалов *механические* устройства, заимствованные из смежных технологий — связь и оргтехника, — телетайпы (типа ТА-67), телеграфные аппараты (СТА-2М), электрические пишущие машинки (ПМ типа CONSUL). Это был довольно длительный период, в течение которого сложились определенные стандарты, приемы работы оператора и протоколы ввода/вывода и интерпретации данных. Строка информации, вводимая оператором, являлась, как правило, *командой,* требующей выполнения определенных действий от ЭВМ (ОС). Конечная ширина листа (или бумажной ленты) ПМ (80 знаков) ограничивала длину возможных команд. Признаком окончания ввода команды являлось нажатие клавиши <ВК> *(возврат каретки,* она же <CR> — <Carriage Return>, <Return>, <Enter> и пр.). Реакция системы (ответ на запрос, сообщение об ошибке, небольшая порция выходных данных) также выводилась строками по 80 символов, образуя вместе с копиями команд *протокол диалогового сеанса* (или журнал - log) в бумажной форме.

Низкие скорость обмена информацией с ЭВМ и надежность механических терминалов, а также трудности с исправлением информации (*редактированием*)ограничивали применимость и, в частности, делали бессмысленным их использование пользователями-программистами для отладки программ и прочих манипуляций. В ранних версиях операционной системы О5/360 и других систем того времени единственный механический терминал устанавливался

в машинном зале и предназначался для *оператора ЭВМ.* Это устройство получило название *консоль*[*)] (console). На крупных вычислительных установках их могло быть несколько (Master Console, Alternative Con5ole и пр.).

Появление в начале 70-х годов *электронных терминалов,* специально разработанных для использования с ЭВМ, привело к настоящему перевороту в применении машин, существенно приблизив все типы пользователей к вычислительному процессу, облегчив разработку и отладку программ, а также эксплуатацию автоматизированных систем.

Электронный или *видеотерминал* — CRT-device (Catode Яay TиЬe — устройство с электронно-лучевой трубкой), VDU(Video Display Unit — устройство отображения информации), первоначально получивший в отечественной практике наименование *дисплей,* представляет собой клавиатуру (keyboard), сопряженную с экранным устройством (screen). Ранние модели видеотерминалов (ВТ) не были избавлены от наследия ПМ — состав клавиатуры, построчный ввод и исправление ошибок, прокручивание экрана наподобие бумажной ленты (scrolling) и, самое главное, — *символьный* (алфавитно-цифровой) характер выводимой информации, хотя, как это хорошо известно из опыта телевидения, никаких технических ограничений экран (в отличие от каретки ПМ) не вносит. Более совершенные ВТ, разработанные в 80-е годы (IBM 3270, VT-100), во многом определили современное состояние устройств:

- появились возможности прямого доступа к информации на экране (для ввода и корректировки);
- на клавиатуре добавились *функциональные* клавиши, реакция на которые определялась программой, работающей с ВТ;
- *клавиши редактирования* — <Del>, <Ins>;
- *клавиши управления курсором* (для выбора места на экране);
- *управляющая клавиша* <Control> (<Ctrl>), модифицирующая вводимый код, при одновременном нажатии с символьной клавишей и т. п.

Однако это все еще были алфавитно-цифровые устройства, отображающие на экране массив символьной информации размером в 80 столбцов на 17 строк (т. е. до 1600 символов).

---

*) Как известно, в строительстве и архитектуре консолью именуют конструкцию, состоящую из горизонтальной балки, опирающейся на подкос. Именно так выглядит столик для ПМ *(пульт оператора),* прикрепленный к *инженерному пульту* управления большой ЭВМ.

Типовая конфигурация машины (до появления ПЭВМ) включала в себя 8 (или 16, или 32) *терминалов пользователя,* размещенных в специальных помещениях *(дисплейные классы),* и одну или более *машей-консоль* (терминал оператора), размещенную поближе к месту основных событий (в машинном зале).

Конфигурация ПЭВМ, в которую входит единственный ВТ (МОНИТОР) является *частным* (или, как выражаются математики, *вырожденным*) случаем общей конфигурации, при этом ВТ несет бремя нагрузки как консоли, так и пользовательского терминала.

Терминал ПЭВМ (в дальнейшем будет упоминаться как *консоль,* поскольку в MS-DOS это устройство числится под обозначением CON), в отличие от старинных ВТ, базируется на *графическом выводе* информации (в растровой форме) на экран, что дает возможность отображать не только обычную символьную информацию, но и *квазисимвольную* (элементы электронных схем, шахматные фигуры, редкие алфавиты). Наконец, на подобный ВТ может быть выведена произвольная растровая информация*).

Рассмотрим подробнее *клавиатуру и экран* консоли. *Клавиатура* (рис. 1,25) включает следующие области (заметим, что ряд областей или отдельных клавиш продублирован).

1. *Символьная область.* Здесь находятся клавиши, являющиеся основными для ПМ и механических терминалов, — строка цифровых клавиш, три строки буквенных клавиш, пррбел (<Space>). Необходимость совместного использование символов *латиницы (A—Z)* и *кириллицы (А—Я)* создает проблему размещения символов по клавишам. Как известно, месторасположение символов отражает их частоту и совместную встречаемость в словах соответствующего языка, в связи с чем отечественные клавиатуры в первой символьной строке содержат буквы ЙЦУКЕН, англо-американскому стандарту соответствует строка QWERTY, континентально-европейскому стандарту - AZERTY.

Первые отечественные терминалы использовали в качестве основы размещение ЙЦУКЕН, привязывая к символам кириллицы

*) Терминалы ПЭВМ (относящиеся к так называемым ANSI-терминалам) по мере развития технических средств претерпели ряд изменений: улучшение разрешающей способности (количество точек-*пикселей* на экране), увеличение числа уровней яркости и количества отображаемых оттенков цвета. Это развитие осуществлялось не столько за счет усовершенствования «телевизора», сколько путем разработки новых контроллеров (управляющих карт, или *адаптеров).* Ранние ПЭВМ были укомплектованы контроллерами CGA (Color Graphics Adapter), затем появились EGA (Enhanced Graphics Adapter), VGA (Video Graphic Array, обеспечивающий качество, близкое к видеоизображению телевизора) и т. п.

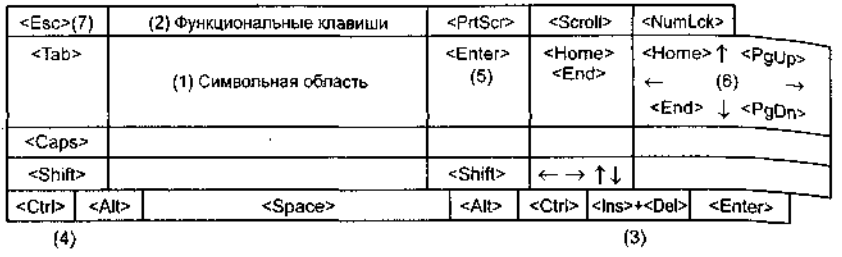



Рис. 1.25. Структура клавиатуры консоли

соответствующую им по *правилам транслитерации* латиницу: Й/J, Ц/C, У/U, К/K, Е/E, Н/N и т. п. На консоли ПЭВМ поддерживаются два стандарта и размещение символов имеет вид Q/Й, W/Ц, Е/У, Я/K, Т/Е, У/Н, что обычно смущает начинающего пользователя.

2. *Функциональная клавиатура* (ФК), сохранившаяся как знак преемственности со старыми терминалами, хотя принципы обмена информацией консоль-ЭВМ здесь таковы, что необходимость в ней отсутствует *(вся клавиатура является программно-управляемой).* За последние годы сложились определенные *стандарты-де-факто* применения ФК, например <P1> — HELP (Помощь, подсказка), <F10—F12> — QUIT (Завершение работы программы) и т. п.

3. *Клавиши редактирования* — <Ins> — включение/выключение режима вставки символов, <Del> — удаление текущего символа, <BS>, <BackSpace> — удаление символа слева.

4. *Управляющие клавиши* (изменяют значение нажимаемого одновременно с ними символа):

<*Shift*> — переключение регистров, имеется также на любой ПМ. В буквенной области <Shift> переключает строчные символы на заглавные, в цифровой области — цифры на служебные символы (@# $ % ^ и т. п.);

<*CapsLock*> — фиксация верхнего регистра, в отличие от ПМ, действует только на буквенные клавиши;

<*Ctrl*> — появился впервые на VT100. Позволяет вводить коды, которым не соответствуют какие-либо обычные символы. Например, <Ctrl+Z> вводит символ EOP — *конец файла*;

<*Alt*> — появляется на ANSI-терминале. Расширяет возможности <Ctrl>. Например, <Alt+2+1+9> вводит | — символ т. н. *псевдографики.*

5. <*Enter*> — *ввод.* Является символом окончания строки, соответствует клавише <BK> механического терминала, клавиша продублирована.

6. *Клавиши управления курсором* — <←> Стрелка влево, <→> — Стрелка вправо, <T> — стрелка вверх. <↓> — стрелка вниз, <Home> — начало, <End> — конец, <PgUp> — страница назад, <PgDn> — страница вперед. Действие клавиатуры рассматривается ниже. Клавиши продублированы. Основная зона расположена на правом краю клавиатуры и совмещена с второй цифровой клавиатурой (основная размещена в символьной области). Переключение регистров на этой клавиатуре (цифры или управление курсором) осуществляется клавишей <NumLock> — зафиксировать цифровой режим. Дубль клавишей управления курсором находится левее, перед символьной областью.

7. Клавиша <Escape> (Выйти) впервые появляется на VT100 и реализует выход из текущей программы. Обычно так же программируется и на ПЭВМ.

В заключение разговора о клавиатуре поясним, что понималось выше под *программируемостью*. Это означает, что интерпретация всех перечисленных клавиш не обязательно соответствует тем или иным символам/действиям, которые на них обозначены. *Нажатие на клавишу вырабатывает не код символа, а номер клавиши* (поэтому основные и дублирующие символы/области в принципе различаемы). Эта информация затем обрабатывается *драйвером клавиатуры* программой, постоянно находящейся в ОП и преобразовывающей номер клавиши в код символа, который выводится на экран и поступает в распоряжение работающей прикладной программы. Этот же драйвер ответствен за переход с *латиницы на кириллицу*. В отличие от механических терминалов и старинных ВТ, на консоли нет клавиши переключения <Лат/Рус>. Поскольку драйверов весьма много (всякий себя уважающий программист в середине своей карьеры, как правило, пишет свой драйвер), надо иметь в виду, что переключение может осуществляться по-разному. Обычно используются сочетания управляющих клавиш (например <Shift+Alt>, <Shift+Shift> — левая и правая клавиши и пр.). Отсутствие лампочки, которая на старых ВТ указывала, что включен регистр «Лат» (или «Рус»), компенсируется обычно появлением на экране цветной Рамки или надписей LAT, RUS в углу экрана.

Очевидно, драйвер управляет привязкой символов к клавишам. Обычно Размещение букв не вызывает проблем — это стандарты QWERTY и ЙЦУКЕН. Размещение же служебных символов (! @ # $ % и пр. может различаться в разных драйверах и может не соответствовать в связи с этим символам, нанесенным на клавиши. И буквами не все однозначно, особенно достается буквам Ъ и Ё (иногда даже Ъ и ъ могут оказаться на разных клавишах).

***Экран.*** В настоящее время сформировались следующие основные *режимы представления и управления информацией* на экране, которым соответствуют определенные сценарии диалога *человек-ЭВМ (в текстовом режиме):*

- режим командной строки;
- режим форматированного экрана;
- режим меню.

1. *Режим командной строки.* Работа в этом режиме мало чем отличается от работы с механическим терминалом или с самыми первыми ВТ.

Экран состоит из двух областей — *командной строки* (аналога строки, на которой находится каретка ПМ) и *протокола диалога* (history) (аналога отрезка бумажной ленты). На экране рис. 1.26 пользователь ввел команду DIR, просмотрел результаты и вводит команду COPY. Однако здесь появляется важный новый объект (как и на первых ВТ) — *курсор, или активная область экрана,* с которой пользователь работает в данный момент. В большинстве ситуаций (и в данном режиме тоже) эта область занимает минимально доступную площадь — *одно знакоместо*, т. е. площадь, на которой для данного терминала отображается один *знак* (символ). Попытки перевести слово *cursor* как *бегунок* и т. п. не прижились. Курсор выделяется яркостью, мерцанием или цветом, так что его всегда мож-

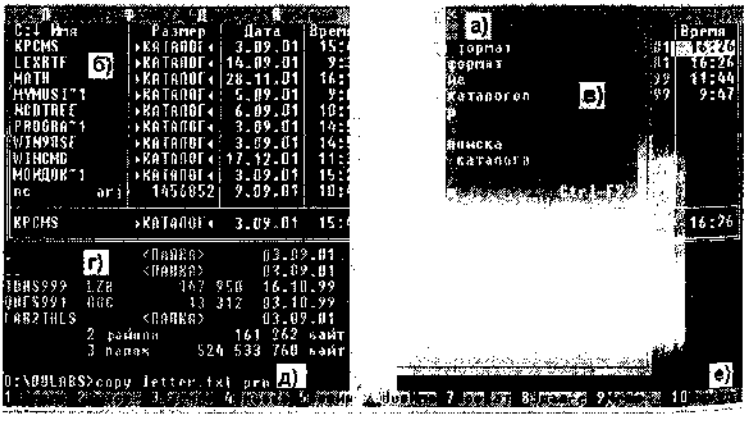



Рис. 1.26. Экран Norton Commander — использование различных элементов интерфейса: а) — ниспадающее меню; б) — левая и правая панели (всплывающие меню); в) — вертикальное меню; г) — видимый участок протокола; д) — командная строка; е) — комментарий к ФК (горизонтальное меню)

но р... • познать. При вводе символ в строке появляется в позиции курсо ...па, а тот автоматически смещается вправо.

П... необходимости откорректировать строку используют кла-ви... <←> и <→>, перемещающие курсор к месту исправления. *О... льные клавиши управления курсором в командном режиме не задействованы.* Из этого правила, конечно, есть исключения: в неко-т... ых системах <T> позволяет вызвать в командную строку предшествующую команду, что создает известные удобства при вводе серии сходных команд. Клавиши <T> и <↓> здесь дают возможность *перелистать* журнал, где хранятся введенные команды, выбрать, исправить и выполнить требуемую команду.

2. *Режим форматированного экрана (ФЭ)* — рис. 1.27. В этом случае экран представляет собой совокупность *окон*, каждое из которых содержит некоторое элементарное данное и обычно снабжено текстовым комментарием (как правило, название данного). Если командный режим в основном ориентирован на управление вычислительными процессами (хотя и может быть применен для *ввода/вывода данных),* ФЭ в основном именно на эти функции (здесь, как и везде, есть исключения, например какое-то из окон может использоваться для ввода команд).

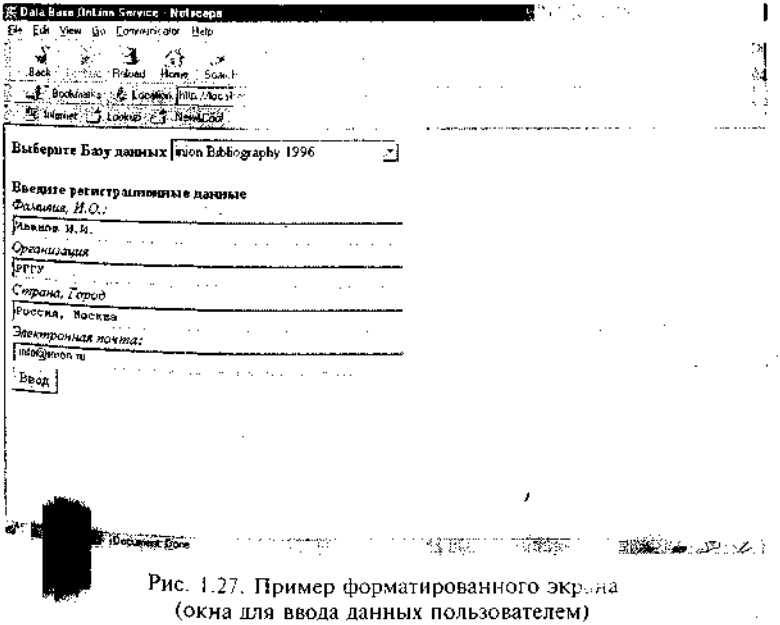



Рис. 1.27. Пример форматированного экр...а (окна для ввода данных пользователем)

Традиционно основным способом использования ФЭ является работа с *файлами данных,* или совокупностями *агрегатов данных (записей),* одинаковой структуры.

Видимая на экране в режиме ФЭ запись является т. н. *текущей* или *активной.* Данные, составляющие содержимое ее полей, находятся в оперативной памяти и могут быть изменены путем подвода курсора и редактирования в окне. Интерпретация клавиш управления курсором в ФЭ зависит от программы, с которой осуществляется работа, однако сложились следующие стереотипы, справедливые для многих популярных программных средств:

<←> <→> — переход внутри окна на одну позицию;

<T> <!> — переход к следующему/предшествующему окну;

<Home> — переход к началу поля;

<End> — переход к концу поля;

<PgUp> — вызвать предшествующую запись (переместить указатель текущей записи к началу файла);

<PgDn> — вызвать последующую запись (переместиться к концу файла).

***Замечания.***

*1. В окне может быть представлена только часть соответствующего поля, если таковое имеет слишком большую длину, и тогда осуществляется горизонтальная прокрутка (scrolling) содержимого при нажатии клавиш <←>, <→>, <Home>, <End>.*

*2. Запись большого размера может занимать несколько экранов, и тогда <PgUp>, <PgDn> вначале перелистывают экраны текущей записи, а затем вызывают соседнюю запись.*

*3. Нажатие <←>, <→> в начале (конце) поля (окна), как правило, вызывает переход к предшествующему/последующему окну. Клавиши <↑>, <↓>, нажатые в первом (последнем) окне экрана, вызывают переход к предшествующей/последующей записи.*

Важным частным случаем ФЭ является окно *во весь экран,* характерное для *текстовых редакторов,* программных продуктов, предназначенных для манипуляции с *текстовыми файлами.*

*Графический интерфейс* ***пользователя.*** Графический интерфейс пользователя (GUI — Graphics User Interface). Появление операционных систем и оболочек с развитыми диалоговыми графическими средствами (О8 Macintosh, Windows 3.1, а особенно Windows 95/98/ME, а также NT/2000) и средств программирования, позволяющих создавать графические интерфейсы (PoxPro for Windows и пр.), а особенно — объектно-ориентированных систем программирования — привело к внедрению и широкому распространению элементов экранного интерфейса.

Графические интерфейсы иногда обозначают следующей аббревиатурой *WIMPD (WIndows, Menu, Pointng Device)* — окна, меню, указывающее устройство, как основные действующие элементы в подобном интерфейсе.

Оболочка Microsoft Windows не была изначально операционной системой, так как она существует «поверх» операционной системы типа MS-DOS. Она возникла в виде стандартизатора графического интерфейса и прижилась исключительно потому, что пользователь хотел видеть программу, с которой ему часто приходится работать, красивой, практичной, удобной и легкой в освоении и использовании.

Для ОС UNIX также был создан специальный графический интерфейс — X Window; фирма IBM выпустила вместе с операционной системой OS/2 свой вариант графического интерфейса пользователя — Presentation Manager.

Функции, используемые программой пользователя при работе с графическим пользовательским интерфейсом, схожи, как и сами интерфейсы.

Операционная система (оболочка), ориентированная на графический интерфейс пользователя, предоставляет не только функции,

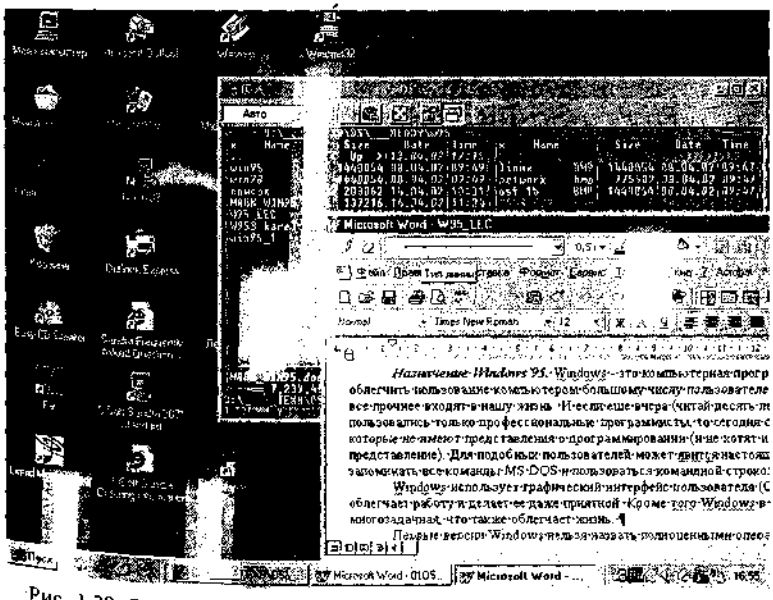


Рис. 1.28. Рабочий стол Windows 98, пиктограммы, окна приложений

поддерживающие ввод-вывод, но и широкий спектр системных вызовов, позволяющих использовать различные графические примитивы: от самых простых (точки, линии, дуги) до самых сложных (области, окна, курсоры). Основным преимуществом использования графического интерфейса операционной системы является то, что с помощью него программа может создавать графические изображения, которые будут выглядеть одинаково на всех устройствах, поддерживаемых операционной системой (принцип WYSIWYG — What Уои Зее Is What Уои Get — что видим, то и получаем).

Большое внимание в графическом интерфейсе операционной системы обычно уделяется шрифтам. Исторически сложилось так, что первыми и долгое время единственными шрифтами для компьютеров оставались растровые (точечно-матричные) шрифты. Такие шрифты занимали малый объем памяти, однако их символы невозможно было вращать, наклонять, уменьшать без искажений или увеличивать можно было только в целое число раз. С появлением графического интерфейса операционные системы стали предоставлять системные средства для поддержки использования векторных шрифтов, которые не только легко масштабируются, меняют наклон и толщину, но и выглядят одинаково на всех устройствах, поддерживаемых операционной системой. Каждая операционная система поддерживает свой стандарт векторных шрифтов (TrueType для Microsoft Windows; Adobe Type Manager для О5/2; GhostScript для LINUX).

*Графический интерфейс включает следующие понятия — рабочий стол, окна, пиктограммы, элементы графического интерфейса (виджеты), указывающее устройство (мышь).*

После запуска программа обычно создает окно, с которым она ассоциируется и работает. Пользователь, работая с окном и находящимися в нем объектами, заставляет операционную систему (или программную оболочку) посылать программе сообщения, активизирующие необходимые пользователю возможности программы. В процессе работы программа также может создавать другие окна (выбора, диалога, обрабатываемого файла и др.) и получать от них сообщения, таким образом, стандартизируются часто используемые элементы диалога с пользователем.

При уменьшении некоторого окна до пиктограммы освобождается место для другого окна, которое может быть увеличено или уменьшено в соответствии с потребностями.

Если продолжить аналогию с поверхностью рабочего стола, то каждое работающее приложение можно рассматривать как, например, скоросшиватель с бумагами по определенной теме. Раскрывая скоросшиватель не полностью, можно получить возможность рабо-

тать с бумагами, одновременно не теряя возможности наблюдать за ситуацией на столе. «Распахнув» скоросшиватель в полный формат, получите возможность «с комфортом» работать над содержащимися в нем бумагами, но они при этом занимают всю поверхность стола, накрывая все остальное. Завершив сегодня работу с данным скоросшивателем, можно свернуть все бумаги и закрыть скоросшиватель, не убирая его, однако, с поверхности стола.

Представление и расположение окон в значительной мере зависят от того, сколько приложений одновременно выполняется в среде. Если активно всего одно приложение, то целесообразно представить соответствующее окно в полноэкранном варианте. Работа одновременно с двумя приложениями предполагает наличие двух окон нормального размера, размещенных на двух половинах экрана, верхней и нижней (или левой и правой). При работе с большим числом приложений удобно часть приложений, в которых в данный момент пользователь не испытывает острой необходимости, представить пиктограммами.

Интерфейс оболочки представляет собой набор наглядных и естественным образом организованных средств управления приложениями. Работая в графической среде, пользователь уже для вызова приложений не вводит имена и директивы с клавиатуры, а оперирует с соответствующими пиктограммами с помощью мыши.

Графические оболочки делают технологию работы с компьютером более естественной и ясной. Большую роль здесь играет мышь как основной инструмент управления машиной. В целом ряде случаев для вызова некоторых (довольно сложных) операций достаточно просто *«перетащить и положить» (Drag-and-Drop)* пиктограмму или другой объект с помощью мыши. Например, в оболочке Windows 3.1 для распечатки некоторого документа достаточно с помощью мыши «вытащить» из окна Менеджера Файлов (File Manager) пиктограмму соответствующего файла и «положить» ее поверх пиктограммы Менеджера Печати. Перетаскивание пиктограммы файла Документа в открытое окно редактора Write загружает соответствующий документ в окно.

***Основные элементы графических интерфейсов (виджеты, widgets).*** Виджет — это заготовка части пользовательского интерфейса (кнопка, часть меню, пиктограмма и т. д.) с параметрами, привязываемая к окну экрана терминала. Наиболее распространенные: кнопка (Button); радиокнопка (Radio Button); флажок (Check Box); список (List); полосы прокрутки и т. д.

Основные элементы диалогового окна, создаваемого для взаимодействия с активным приложением, приведены на рис. 1.29.

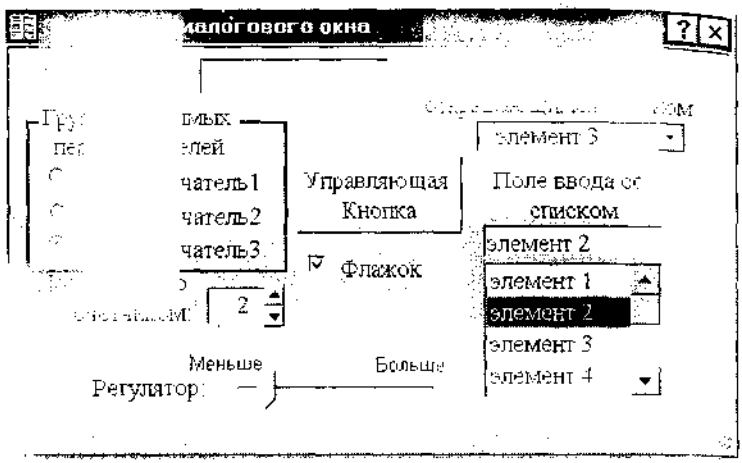



Рис. 1.29. Элементы диалогового окна

*Управляющие кнопки (Button)* — предназначены для выполнения действий. Какое именно действие выполняет кнопка, написано непосредственно на ней. Кнопка приводится в действие нажатием мыши на ней. Если в конце названия кнопки присутствуют три точки, то такая кнопка вызовет новое диалоговое окно (см. также рис. 1.30, 1.31, 1.32).

*Поле ввода* — область, где пользователь может вводить информацию с клавиатуры. В этой области указатель мыши принимает новую форму. Если в этот момент щелкнуть кнопкой мыши, то в поле появится курсор и можно вводить данные.

*Список* — элемент, содержащий все возможные в каждом конкретном случае значения, которые пользователь может установить. Добавить или изменить эти значения непосредственно в списке нельзя.

*Раскрывающийся список* (List) — при нажатии на пиктограмму со стрелкой открывается список всех возможных значений, которые можно выбрать для установки в этом элементе. Если список длинный, то появится линейка прокрутки, с помощью которой можно просмотреть все элементы списка (рис. 1.30).

*Поле ввода с раскрывающимся списком* — это комбинация элементов *поле ввода* и *раскрывающегося списка*. Такой элемент позволяет как непосредственно (вручную) вводить данные в *поле ввода*, так и заполнить его значением из раскрывающегося списка. Аналогично работает *поле ввода со списком*. Отличие только в том, что список виден постоянно, а не открывается (рис. 1.31).

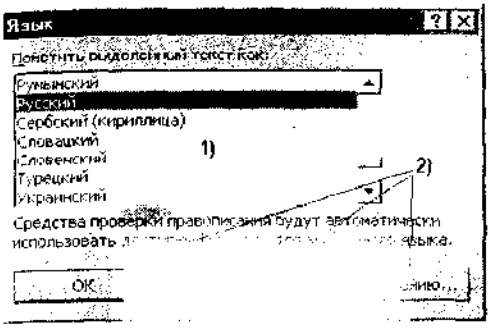



Рис. 1.30. Раскрывающийся список с фиксированным словарем 1), управляющие кнопки 2)

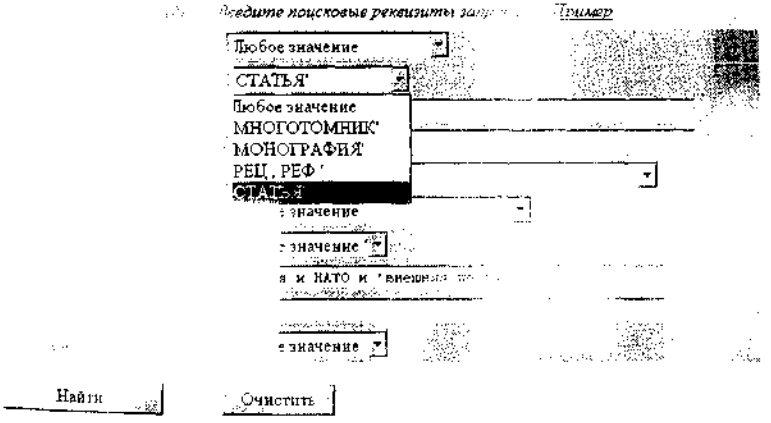



Рис. 1.31. Поля ввода, поля ввода с раскрывающимся списком, раскрывающиеся списки

*Поле ввода со счетчиком* — обычно используется для ввода числовых значений. Его можно заполнить как обычное поле ввода или воспользоваться кнопочками, расположенными справа. В этом случае значение в поле будет изменяться (соответственно увеличиваться и уменьшаться) с наиболее оптимальным шагом и при этом не превысит предельных значений. Поэтому рекомендуется пользоваться именно счетчиком.

*Флажок* (или *независимый переключатель, CheckBox*) — переключатель для режима работы, описание которого находится справа от квадрата. Он может быть включен (установлен) — внутри квадрата

изображен значок, или выключен (сброшен) — внутри пусто. Для установки или сброса флажка необходимо щелкнуть мышью в квадрате или на его описании. Такой элемент вполне самостоятелен, определяет свой параметр и поэтому называется независимым в отличие от следующего элемента (рис. 1.32).

*Зависимые переключатели (радиокнопки, RadioButton)* — группа переключателей для выбора одного из нескольких возможных взаимоисключающих режимов работы. Описания режимов находятся справа от кружков. В одной группе может быть включен только один из переключателей остальные автоматически сбрасываются Включенный (активный) режим индицируется точкой внутри кружка. Для включения нужного режима необходимо щелкнуть мышью в кружке или на его описании (рис. 1.33).

*Регулятор* — используется для установки параметров от минимального до максимального с помощью «движка».

При этом такие элементы экрана, как меню различных типов, строки ввода данных и команд также широко встречаются в современных интерфейсах (рис. 1.34). Просьба к читателю самостоятельно идентифицировать на данном экране как перечисленные выше. так и иные элементы интерфейса.

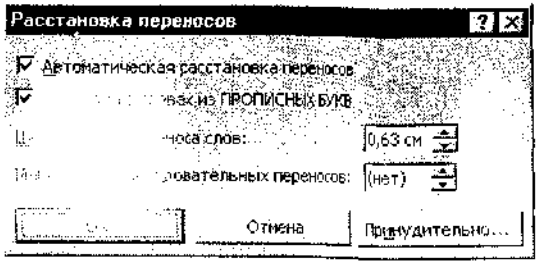



Рис. 1.32. Блок независимых переключателей (флажки, CheckBox)

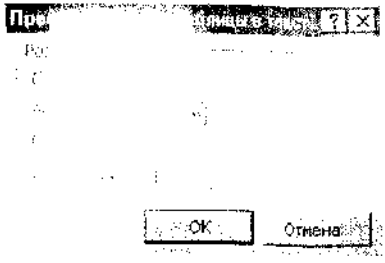



Рис. 1.33. Блок зависимых переключателей (радиокнопки, RadioButton)

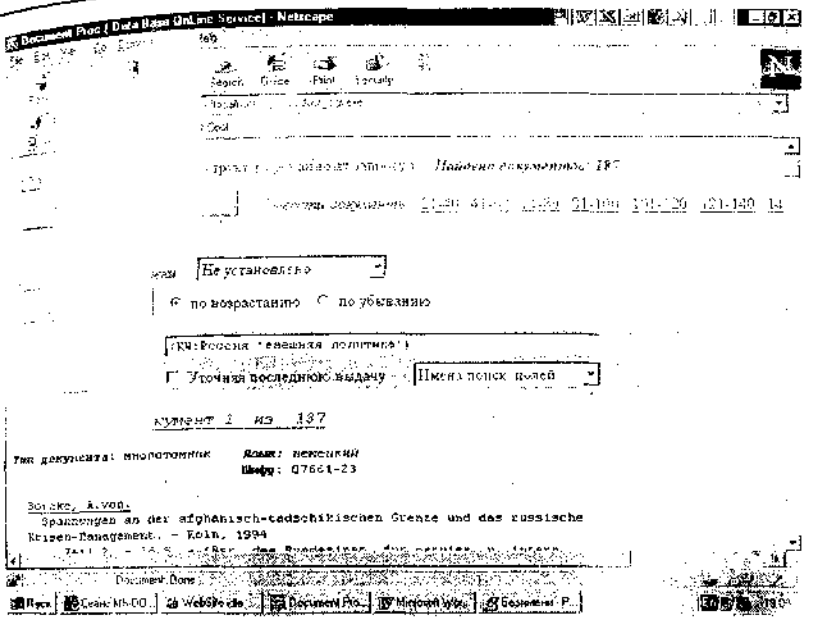



*Рис. 1.34.* Комбинация элементов интерфейса в реальном приложении WinIrbis

## Вопросы к главе 1

1. Каков состав программного обеспечения ЭВМ?
2. В чем заключаются функции операционных систем?
3. В чем заключается управление данными?
4. Какова организация файлов на МЛ?
5. Что такое логическое начало ленты? Физическое начало ленты?
6. Для чего используется блокирование записей на магнитных носителях?
7. Приведите аргументы за и против увеличения длины блока на МЛ (МД).
8. Какие виды нарушения целостности данных в простейшей ФС вы можете привести?
9. Изобразите схематическую структуру ФС с косвенной адресацией и приведите примеры нарушения целостности данных.
10. Охарактеризуйте основные типы файлов. Дайте перечень основных типов текстовых файлов.

11. Дайте основные характеристики накопителей на гибких магнитных дисках.
12. В чем заключается управление задачами в ОС?
13. Что такое процесс в вычислительной системе? Какие типы процессов вам известны?
14. Какие состояния процессов вам известны? Изобразите диаграмму перехода процессов из одних состояний в другие.
15. Какие дисциплины очередей используются в ОС, каковы их достоинства и недостатки?
16. Что такое управление памятью?
17. Охарактеризуйте режимы MPT и MVT.
18. Что такое страничная память? Приведите примеры организации страничной памяти.
19. Какие типы пользовательских интерфейсов вам известны?
20. Какие элементы диалогового окна используются для управления приложениями?
21. Что такое виджеты? Перечислите известные типы виджетов.
22. Укажите основные элементы интерфейсов, входящих в экран, пример которого приведен на рис. 1,34.

# Глава 2. Операционные системы персональных компьютеров — однопользовательские, однозадачные и многозадачные

Как это отмечалось выше, по своим функциональным свойствам ОС могут быть разделены по меньшей мере на 3 группы:

- *однопользовательские однозадачные;*
- *однопользовательские многозадачные;*
- *многопользовательские многозаданные.*

Несмотря на то, что в исторической последовательности первыми на сцену вышли (в 60—70-е годы XX века) именно *многопользовательские многозадачные* ОС (OS 360/370. RSX и пр.), наибольшее распространение (вместе с ПК) получили *однозадачные* ОС (MS-DOS и ее аналоги).

В настоящей главе предполагается рассмотреть ОС для ПК — MS-DOS, графическую оболочку Windows 3.1.x, а также ОС Windows 95/98/ME, NT/2000. Надо отметить, что начиная с Windows 3.11 for Workgroups данные ОС и оболочки начинают приобретать *многопользовательский многозадачный* характер.

Тем не менее все они рассматриваются в настоящей главе в связи с тем, что в подавляющем большинстве случаев данные ОС, несмотря на свои потенциальные возможности, эксплуатируются в *однопользовательском* (пусть и *многозадачном)* режиме. Что же касается *многопользовательских многозадачных* ОС, то они будут описаны в следующей, 3-й главе.

## 2.1. Операционная система MS-DOS

***Краткая история операционной системы MS-DOS.*** Краткая история операционной системы MS-DOS, получившей широчайшее распространение во всем мире и используемой в количестве, по различным оценкам от 100 до 150 млн экземпляров, начинается со скромной системы 86-DOS, написанной в середине 80-х годов

Т. Петерсоном для компании Seattle Computer Products. При разработке 86-DOS были учтены требования совместимости с весьма популярной в то время системой CP/M-80, предназначенной для восьмиразрядных микрокомпьютеров на базе процессоров Intel 8080, Zylog 2-80. В результате и в нынешних вариантах MS-DOS можно найти немало структур данных и программных средств, характерных для CP/M-80 [15].

В июле 1981 г. фирма Microsoft приобрела права на систему 86-DOS, существенно переработала ее и выпустила на рынок под названием MS-DOS (Microsoft Disk Operating System). Когда осенью 1981 г. появились первые персональные компьютеры фирмы IBM, система MS-DOS 1.0 и ее аналог фирмы IBM PC-DOS 1.0 быстро стали основными системами для этих машин. В то же время непрерывное развитие аппаратных средств компьютеров и накопление опыта работы с ними привели к необходимости столь же непрерывного совершенствования исходных систем MS-DOS и PC-DOS. В дальнейшем они развивались параллельно и их новые версии соответствовали друг другу. К настоящему времени выпущено уже 6 версий MS-DOS (и еще большее число вариантов, если считать подверсии) и готовится к выпуску MS-DOS 7.0.

Первое серьезное усовершенствование MS-DOS (версия 2.0) было выполнено в 1983 г. Фактически была выпущена новая операционная система, хотя разработчикам удалось обеспечить полную совместимость с MS-DOS 1.0. В систему MS-DOS 2.0 были включены следующие новшества:

- поддержка дискет с повышенной плотностью записи и, главное, появившихся к этому времени жестких дисков;
- иерархическая структура каталогов (пришедшая из системы UNIX) вместе с группой команд ее поддержки (CD, MD, RD и др.);
- перенаправление ввода-вывода, конвейеры и фильтры (тоже средства, характерные для системы UNIX);
- утилита PRINT, обеспечивающая вывод на печать в фоновом режиме с возможностью одновременного выполнения любой программы;
- атрибуты файлов и их системная поддержка (в частности команда ATTRIB);
- метка тома и соответственно команды LABEL и VOL;
- устанавливаемые драйверы внешних устройств;
- драйвер ANSI.SYS для расширения возможностей экрана и клавиатуры;
- файл конфигурирования CONFIG.SYS;

- поддержка блоков окружения и соответственно команда SET;
- динамическое выделение и освобождение памяти;
- поддержка национальных форматов;
- расширение возможностей командных файлов (команды ECHO, FOR, COTO и др.);
- большая группа новых команд, утилит и драйверов устройств (BACKUP, RESTORE, EXIT, FIND, KEYB, PATH, PROMPT, SET, VDISK.SYS и др.).

Система MS-DOS 3.0 появилась в августе 1984 г., одновременно с выпуском компьютеров IBM PC/AT на базе процессоров 80286. Начиная с этой версии, в MS-DOS входит поддержка расширенной памяти, жестких дисков увеличенного объема, разделяемых файлов (команда SHARE).

Начиная с версии 3.1, выпущенной в ноябре 1984 г., в MS-DOS включается поддержка сетевых структур.

В версиях MS-DOS 3.2, и особенно 3.3, получили дальнейшее развитие возможности установки национальных форматов, введена поддержка дискет диаметром 3,5 дюйма и жестких дисков с емкостью более 32 Мбайт за счет создания на них нескольких разделов по 32 Мбайт (или менее) каждый, включен ряд новых команд и утилит (APPEND, CALL, CHCP, FASTOPEN, NLSFUNC, REPLACE, XCOPY), а также драйверов устройств (DISPLAY.SYS, DRIVER.SYS).

В 1988 г. появилась версия MS-DOS 4.0, для которой фирма Microsoft разработала собственную оболочку SHELL (в версии 4.01 был разработан русифицированный вариант оболочки). Кроме этого, в версию 4.01 включена поддержка разделов на жестких дисках, превышающих 32 Мбайт, средства эмуляции дополнительной памяти, а также ряд новых команд (APPEND, MEM, TRUNAME).

В MS-DOS версии 5.0 существенно улучшена поддержка расширенной и дополнительной памяти, усовершенствована оболочка SHELL, включен улучшенный интерпретатор QBASIC (вместо утил BASIC и BASICA предыдущих версий), добавлен ряд новых команд, утилит и драйверов (DOSKEY, EDIT, PC, HELP, MIRROR, SETVER, UNDELETE, UNFORMAT, HIMEM.SYS, RAMDRIVE.SYS, SMARTDRV.SYS). Пожалуй, наиболее привлекательной чертой MS-DOS 5.0 явилась возможность организации на компьютерах с расширенной памятью специальных областей — области старшей памяти (HMA) и блоков верхней памяти (UMB), куда можно загружать устанавливаемые драйверы, резидентные программы и бо́льшую часть самой DOS. Это позволяет существенно увеличить объем памяти, отводимой прикладным програм-

мам (до 600—610 Кбайт), и в настоящее время является общепринятой методикой конфигурирования системы.

Операционная система MS-DOS 6.0, выпущенная в 1993 г, брала в себя все лучшие качества предыдущих версий и отвечает современным взглядам на программные продукты для персональных компьютеров. Из MS-DOS 6.0 удалены некоторые устаревшие средства (ASSIGN, BACKUP, COMP, EDLIN, GRAFTABL, JOIN, MIRROR, RECOVER, TRUNAME). В то же время в систему включен целый ряд полноэкранных инструментальных утилит, охватывающих практически весь диапазон потребностей пользователей персональных компьютеров. Утилиты имеют развитый интерфейс пользователя, могут управляться как от клавиатуры, так и мышью, включают контекстные справочники и элементы обучающих систем. В MS-DOS 6.0 входят следующие утилиты:

- оболочка MS-DOS SHELL — многофункциональная программа, существенно упрощающая работу пользователя с файлами, каталогами и программами и предоставляющая ему ряд дополнительных возможностей, отсутствующих в самой DOS, например объединение программ в программные группы, защита их паролем, организация многозадачного режима с удобным переключением между задачами и другие;
- утилита резервного копирования MSBACKUP, осуществляющая получение резервных (архивных) копий файлов жесткого диска на архивных дискетах. Утилита обеспечивает все основные режимы резервного копирования (полное, инкрементное и разностное) и отличается высокой эффективностью;
- утилита DEFRAG, служащая для оптимизации файловой структуры на диске путем дефрагментации файлов и диска в целом;
- антивирусная программа-утилита MSAV, позволяющая вылечить диск, инфицированный вирусами;
- утилита MSD для получения технической информации о вычислительной системе;
- системный отладчик DEBUG, позволяющий отлаживать и изучать работу выполнимых программ, а также выполнять операции с памятью и портами компьютера;
- текстовый редактор MS-DOS EDIT, позволяющий создавать и редактировать текстовые файлы;
- интерактивный полноэкранный интерпретатор с языка Бейсик QBASIC;
- утилита MEMMAKER, позволяющая организовать оптимальное использование наличной памяти;

- утилита сжатия диска DBLSPACE, осуществляющая сжатие (компрессию) файлов в процессе их записи на диск и автоматическое развертывание при загрузке в память, что позволяет существенно увеличить эффективную емкость диска.

Состав команд MS-DOS 6.0 в целом совпадает с предыдущими версиями, чем обеспечена совместимость версий DOS на уровне интерфейса пользователя. С другой стороны, многие команды DOS приобрели дополнительные свойства; добавлен ряд новых команд (CHOICE, FASTHELP, LOADFIX, NUMLOCK, POWER, VSAVE и др.). Существенно развиты средства межмашинной связи (драйвер INT-ERLNK.EXE и команды INTERLNK и INTERSVR).

Важнейшим усовершенствованием, введенным в версию MS-DOS 6.0, является возможность задания в процессе начальной загрузки альтернативных конфигураций системы (методика использования расширенной и дополнительной памяти, состав загружаемых драйверов устройств, наличие и характеристики электронных дисков и пр.). Альтернативное конфигурирование осуществляется с помощью специальных директив файла CONFIG.SYS. Данная версия операционной системы MS-DOS 6.0 является весьма совершенным программным продуктом, обеспечивающим эффективное использование персональных компьютеров всех моделей — от исходных IBM PC и PC/XT до современных PC/AT и P8/2 на базе процессоров 80386, 80486 и Pentium, оборудованных расширенной памятью, магнитными и лазерными дисками большого объема и средствами межмашинной связи.

*Совместимость операционных систем.* Обычно системное программное обеспечение DOS подстраивается к конкретной машине. При этом оно конструируется так, чтобы могло подойти для любой машины, совместимой с данной (например, для операционных систем PC-DOS или MS-DOS версий COMPAQ или Cordata). Единственная область, где программное обеспечение разных операционных систем Дифференцировано, это файл IO.3У5. Он непосредственно связан с физическим устройством электронного оборудования и организуется независимо каждой фирмой-изготовителем. Однако электронное оборудование разных систем сходно по своему строению, и это обеспечивает совместимость IO.SYS по основным параметрам.

Благодаря такой совместимости пользователь может без затруднений сменить операционную систему на своей вычислительной машине. Однако при переходе на другую систему следует помнить, что системные файлы, как правило, отличаются по размеру от системных файлов системы, работавшей ранее. Если системные файлы

новой системы больше системных файлов предыдущей (не уклад-ваются в отведенное предыдущей системой место), то переход в новую операционную систему может вызвать затруднения. В дополнение в некоторых операционных системах предусмотрены автоматические процедуры, которые устанавливают строго определенный размер каждого системного файла. Тогда. если размеры системных файлов той и другой системы не совпадают, процедуры данной операционной системы могут оказаться неработоспособными.

***Некоторые основные понятия, связанные с функционированием MS-DOS.*** *Устройства ПЭВМ* — зарегистрированы в ОС под зарезервированными именами, типовые значения и содержание которых следующие (см. также табл. 2.1.):

- накопители на гибких магнитных дисках (НГМД) обозначаются латинскими символами A:, B:, емкость ГМД от 180 Кб 2,8 Мб;
- накопители на жестких магнитных дисках (НЖМД) — символами C:, D:, емкость от 100 Мб до 100 Гб;
- LPT — принтер;
- CON— клавиатура ЭВМ (консоль);
- COMx — последовательный порт (разъем, предназначенный для подключения устройств, коммутируемых по предусмотренным стандартам, например аналогичной ПЭВМ).

***Таблица 2.1.*** **Стандартные устройства, зарегистрированные в MS-DOS**

| Имя | Назначение файла |
|---|---|
| AUX | Асинхронный интерфейс |
| CLOCK$ | «Часы» |
| COM1 | Порт последовательного ввода/вывода |
| COM2 | Порт последовательного ввода/вывода |
| COM3 | Порт последовательного ввода/вывода |
| COM4 | Порт последовательного ввода/вывода |
| LPT1 | Порт параллельного ввода/вывода |
| LPT2 | Порт параллельного ввода/вывода |
| LPT3 | Порт параллельного ввода/вывода |
| CON | Консоль (клавиатура) |
| NUL | «Нулевое устройство» |
| PRN | Принтер (аналог LPT1) |

С точки зрения пользователя эти устройства (табл. 2.1) ничем не отличаются от обычных файлов (с ними можно производить те же операции, что и с обычными файлами).

*Набор данных (файл)* — именованная совокупность данных, размещаемых на НГМД или НЖМД. Наименование файла включает *имя и расширение*, строящиеся по принципам, аналогичным принятым для ДРУГИХ ОС.

К полном имени файла разрешается использовать только следующие символы:

A — Z 0 — 9 $ & # ` ~ () - % ! _ ^

В полном имени файла запрещается использовать все остальные символы, а также пробел.

Примеры допустимых имен файлов:

```
Format.com, Read.me, MyFyle.txt, 28-03-99.doc, 123.45.
```

Примеры недопустимых имен файлов:

```
123456789.txt, aa?.doc, 35*.? It.F.doc.txt
```

Наиболее типичные значения расширений отражают следующие типы файлов (см. также табл. 1.2):

- .COM,.EXE,.BI N— исполняемые программные модули;
- .BAT — пакетный (командный) файл;
- .TXT — текстовый файл;
- .ARC — архивный файл.

*Каталог (директория)* — именованная логическая область на диске, содержащая группу файлов, которая в свою очередь включает подкаталоги (субдиректории). Совокупность каталогов образует на Диске иерархическое дерево каталогов, при этом единственная директория, не входящая в другие, называется корневой. При инициализации диска на нем создается только корневой каталог.

Юг отличие от корневого каталога, остальные каталоги (подкаталоги) создаются с помощью команд MS-DOS.Основная цель такой структуры каталогов — организация эффективного хранения большого количества файлов на диске. Каждый каталог (кроме корневого) «знает» «родителя», т. е. другой каталог, к которому «привязан» данный каталог (термин «привязан» иногда заменяется термином «зарегистрирован»). MS-DOS рассматривает каждый каталог (кроме корневого) как файл.

Спецификация файла есть совокупность обозначений, обеспечивающих поиск данных на диске, и имеет следующий вид:

```
устройство\путь\имя.расширения
```

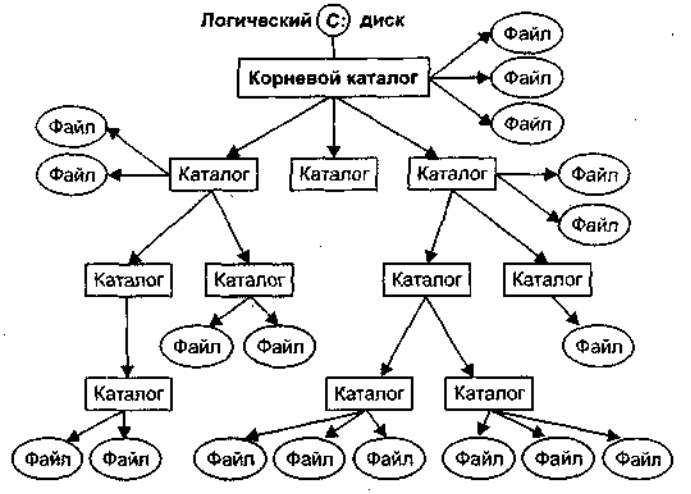



Рис. 2.1. Логическая структура файловой системы MS-DOS

Здесь **путь** — список субдиректорий, входящих друг в друг младшей из которых, собственно, содержится файл.

Примеры:

**a:\command.com** — программный файл, содержащийся в I . невой директории устройства A:;

**b:\sapr\acad.exe** — программный файл, входящий в директорию **sapr** диска **b:**;

**c:\sapr\graph\graphed.com** — программный файл, входящий в подкаталог **graph** 2-го порядка. Здесь путь — **\sapr\graph**.

Часто приходится производить одни и те же действия над многими файлами. Например, пусть необходимо скопировать все файлы какого-либо каталога в другой каталог. Для решения такого рода проблем существуют средства, которые помогают производить однотипные операции над целой группой файлов в одной командной строке. Так называемые символы подстановки позволяют «фильтровать» файлы, используя их имена. К ним относятся символы ? и *. Эти символы можно использовать в любом месте собственно имени файла (имени и расширении).

Символ ? означает, что команда (при фильтрации файлов) принимает любой символ в имени или расширении файла, в позиции которого находится символ ?.

Символ * означает, что команда (при фильтрации файлов; принимает цепочку символов в имени или расширении файла, начиная с позиции, где находится символ *.

Символы ? и * действуют независимо друг от друга применительно к имени или расширению.

примеры.

Выполнить операцию над следующими группами файлов:

*.* — все файлы без исключения;

*.txt — файлы с любым именем, но с расширением.txt;

GG*.* — файлы, имена которых начинаются с цепочки символов GG и имеют любое расширение,

YE??0198.* — файлы, имена которых начинаются с цепочки символов YE, два следующих символа не имеют значения, следующие четыре символа должны быть 0198, расширение — произвольное.

*Текущий дисковод* — устройство, адрес которого подразумевается по умолчанию, при отсутствии его явного указания пользователем. Обычно текущее (или активное) устройство указано в подсказке, выводимой системой на экран, при готовности приема команд пользователя, которая имеет обычно вид:

A:>, B:> и т. д.

Текущий дисковод устанавливается командой A:, B: и пр.

*Текущая директория, каталог (субдиректория)* или текущий путь — совокупность вложенных каталогов, подразумеваемая по умолчанию при поиске файла, если пользователь явно не задал ее в спецификации. Обычно описывается в подсказке типа A:\ACAD>. Устанавливается командой **cd**.

*Управление вводом-выводом.* Ввод и вывод — это процессы, осуществляющие пересылку входных и выходных данных. MS-DOS предусматривает достаточно сложное программное обеспечение для Управления этими процессами по желанию пользователя. Управление данными осуществляется с помощью процедур, называемых направленный ввод и вывод, фильтры и коммуникации. Используя эти процедуры, пользователь может организовать свою линию перебое информации. Он может адресовать поток информации на люцию тройство или в любое место памяти, упорядочить информапропустив ее через фильтр, направляя затем выходной поток, напри... Р, на вход системной программы или обработчика команд.

*Стандартные Устройства ввода-вывода.* Для ввода информации в больш стве случаев используют клавиатуру. В результате выполнения бо ьшинства операций полученные данные выводятся на экран дисп Поэтому клавиатура считается стандартным устройством ввода, а экран — стандартным устройством вывода.

MS-DOS предусматривает средства, позволяющие назначать нестандартны устройства ввода или вывода, Такие устройства назы-

ваются периферийными устройствами ввода/вывода, т. к. они ются внешними по отношению к машине.

*Запуск вычислительного процесса в MS-DOS* осуществляетс тем ввода (полностью или частично) спецификаций програм (типа .EXE, .COM, .BIN) или пакетного (.BAT) файла, распол ного в текущем каталоге текущего устройства.

Основные команды для работы с каталогами: DIR, MK RMDIR, CHDIR.

Основные команды для работы с файлами: TYPE, DEL COPY, RENAME.

Основные команды для работы с дисками: FOK DISKCOPY, VOL, LABEL, CHKDSK, 5У5.

Основные команды конфигурирования системы и управ устройствами: CLS, DATE, PATH, PROMPT, T1ME, VER.

Перечислим наиболее употребительные команды, не вы специально внутренние и внешние (подробный алфавитный пере- чень приведен в табл. 2.2).

FORMAT — подготовка диска к работе (форматирование е или инициализация). Команда внешняя, поэтому подразумевается нали- чие файла FORMAT.COM. Команда предусматривает ключи:

V — форматирование с записью на диск метки (имени) то

8 — перенос операционной системы с резидентного на ф матируемый диск;

MKDIR (MD) — создание каталога (подкаталога). Аргум ентом является имя создаваемого каталога.

COPY — копирование файлов, их групп и обмен между ус ройствами ЭВМ. Формат команды:

```
COPY исходная_спецификация результирующая_спецификация
```

Примеры:

COPY A:ACAD.EXE B:\SAPR — копирование с диска на диск , заданный каталог;

COPY *.* B: — копирование всех файлов из текущего ката текущего устройства на диск B;

COPY CON LPT — копирование текста, вводимого с кла ры, на печать;

COPY *.PAS LPT — распечатка всех ПАСКАЛЬ-программ

COPY COM1 A: — прием данных через последовательный по внешней связи и запись на дисковод A:;

ERASE (DEL) — удаление файлов или их групп;

DEL A:*.FOR — удаление с диска всех ФОРТРАН-програм

*Таблица 2.2.* **Алфавитный перечень основных команд MS-DOS (символом * в третьей колонке отмечены внутренние команды)**

| Команда | Функция | Внутр.(*) |
|---|---|---|
| ANSY.SYS | Установка драйвера консоли | |
| ASSIGN | Переназначение дисковых устройств | |
| ATTRIB | Установка атрибута файла | |
| BATCH | Пакетные командные файлы (*.bat) | * |
| BACKUP | Создание резервных копий для файлов | |
| BREAK | Прерывание программы | * |
| BUFFERS | Создание буферов в ОЗУ | * |
| CHDIR (CD) | Переход в новый каталог | * |
| CHKDSK | Проверка дисков | |
| CLS | Очистка экрана | * |
| COMMAND | Второй командный процессор | * |
| COMP | Сравнение дисковых файлов | |
| COPY | Копирование файла | * |
| COUNTRY | Установка формата даты и времени | |
| CTTY | Переназначение консоли | |
| DATE | Установка даты | * |
| DEBUG | Отладчик программ | |
| DEVICE | Установка новых драйверов устройств | * |
| DIR | Просмотр каталогов | * |
| DISKCOMP | Сравнение дисков | |
| DISKCOPY | Дублирование дискет | |
| DRIVER.SYS | Установка драйвера блочно-ориентированных устройств | |
| ERASE (DEL) | Удаление файлов | * |
| FCBS | Блоки управления файлами | * |
| FDISK | Разбиение жесткого диска на разделы | |
| FILES | Установка числа одновременно открытых файлов | * |
| FIND | Поиск данных | |
| FORMAT | Форматирование диска | |
| GRAFTABLE | Загрузка дополнительных символов для графического режима | |
| GRAPHICS | Распечатка графических изображений | |

*Продолжение табл. 2.*

| Команда | Функция | Внутр.(*) |
|---|---|---|
| JOIN | Логическое объединение каталога на одном диске с другим диском в один каталог | |
| KEYBxx | Загрузка нерезидентных драйверов клавиатуры | |
| LABEL | Создание и замена метки диска | |
| LASTDRIVE | Установка максимального числа доступных дисководов | * |
| LINK | Загрузчик (редактор) связей | |
| MKDIR (MD) | Создание каталога | * |
| MODE | Изменение режимов работы выходных устройств | |
| MORE | Постраничный вывод файлов на экран | |
| PATH | Указание пути поиска | * |
| PRINT | Вывод на печать данных | |
| PROMPT | Изменение формата приглашения DOS | * |
| RENAME (REN) | Переименование файлов | * |
| REPLACE | Селективная замена и копирование файлов | |
| RESTORE | Восстановление файлов, резервированных по команде BACKUP | |
| RMDIR (RD) | Удаление пустого каталога | * |
| SELECT | Установка MS-DOS на новый диск с заданным типом клавиатуры, форматом даты и времени | |
| SET | Установка переменной окружения | * |
| SHELL | Применение дополнительного командного процессора | * |
| SORT | Сортировка данных | |
| SUBST | Создание виртуальных дисков | |
| SYS | Копирование MS-DOS | |
| TIME | Установка времени | * |
| TREE | Вывод дерева каталогов | |
| TYPE | Вывод на дисплей содержимого файла | * |
| VDISK.SYS | Установка драйвера виртуального диска | |
| VER | Вывод версии MS-DOS | * |
| VERIFY | Проверка записи на диск | * |
| VOL | Вывод метки диска | * |
| XCOPY | Выборочное копирование групп файлов и каталогов | |

RMDIR (RD) — удаление каталога, предварительно вычищенного командой DEL;

установка текущего устройства — ввести имя устройства, например «A:»;

CHDIR (CD) — изменить (установить) текущую директорию;

DIR — вывести на экран оглавление каталога;

REN— переименование файла;

TYPE — вывод файла на экран монитора.

В MS-DOS также присутствует категория командного или пакетного файла, состоящего из командных строк MS/DOS, наименований пользовательских программ и командных слов (операторов) типа:

[p — проверка условия;

GOTO — передача управления указанной инструкции внутри файла;

POK — многократное применение одной и той же команды;

PAUSE — остановка процесса до нажатия пользователем какой либо клавиши.

***Основные составные части MS-DOS.*** MS-DOS состоит из следующих компонент:

- блок начальной загрузки;
- модуль взаимодействия с BIOS (**io.sys** для версии 5.0 и выше);
- модуль обработки прерываний (**msdos.sys** для версии 5.0 и выше);
- командный процессор(**command.com**);
- внешние команды (программы) MS-DOS;
- драйверы устройств;
- файл **config.sys**;
- файл **autoexec.bat**.

*Базовая система ввода-вывода* (BIOS) находится в постоянной памяти (постоянном запоминающем устройстве, ПЗУ) компьютера. Эта часть операционной системы является «встроенной» в компьютер. Ее назначение состоит в выполнении наиболее простых и универсальных услуг операционной системы, связанных с осуществлением ввода-вывода. Базовая система ввода-вывода содержит также тест функционирования компьютера, проверяющий работу памяти и устройств компьютера при включении его электропитания. Кроме того, базовая система ввода-вывода содержит программу вызова загрузчика операционной системы.

*Блок начальной загрузки* — это короткая программа, находящаяся в первом секторе каждой дискеты с операционной системой DOS. Функция этой программы заключается в считывании в память

еще двух модулей операционной системы, которые и завершают процесс загрузки DOS.

На жестком диске (винчестере) загрузчик операционной системы состоит из двух частей. Это связано с тем, что жесткий диск может быть разбит на несколько разделов (логических дисков). Первая часть загрузчика находится в первом секторе жесткого диска, она выбирает, с какого из разделов жесткого диска следует продолжить загрузку. Вторая часть загрузчика находится в первом секторе этого раздела, она считывает в память модули *DOS* и передает имуправление.

Загрузчик просматривает корневой каталог системного диска. Проверяет, являются ли первые два файла в каталоге файлами **io.sys** и **msdos.sys**. Если *да* — загружает их в ОЗУ и передает управление MS-DOS. Если *нет* — сообщение на экране и ожидание нажатия какой-либо клавиши пользователем:

`Non-System disk or disk error` *(Несистемный диск или ошибка диска)*

`Replace and press any keyвдКеп ready` *(Замените и нажмите какую-либо клавишу, когда будете готовы)*

Именно поэтому при «изготовлении» *системной дискеты* необходимо переносить файлы **io.sys** и **msdos.sys** на системную дискету с помощью команды **sys.com.**

*Файлы* **io.sys** и **msdos.sys** (они могут называться по-другому, например **ibm.com** и **ibmdos.com** для PC DOS; **drbios.sys** и **drdos.sys** для DR DOS — в зависимости от версии операционной системы). Они загружаются в память загрузчиком операционной системы и остаются там постоянно.

*Модуль взаимодействия с BIOS* **(io.sys)** — это резидентный модуль (всегда находится в ОЗУ после загрузки). Взаимодействует с *BIOS*. Расширяет возможности BIOS или изменяет ее свойства (где необходимо) с помощью дополнительных драйверов.

*Модуль обработки прерываний* **(msdos.sys)** — это резидентный модуль, который обеспечивает интерфейс высокого уровня для прикладных программ, содержит программные средства для управления файлами, устройствами ввода-вывода, обработки исключительных ситуаций (ошибок) и др. Прикладная программа вызывает функции этого модуля через механизм прерываний, передавая (принимая) информацию к (от) MS-DOS через регистры центрального процессора или (и) области памяти ОЗУ. **Msdos.sys** транслирует (переводит) запрос прикладной программы в один или несколько вызовов, адресованных к **io.sys** и BIOS.

*Командный процессор DOS* обрабатывает команды, вводимые пользователем. Командный процессор находится в дисковом файле COMMAND.COM на диске, с которого загружается операционная система. Некоторые команды пользователя, например **type, dir** или copy, командный процессор выполняет самостоятельно. Такие команды называются *внутренними.* Для выполнения остальных *(внешних)* команд пользователя командный процессор отыскивает на дисках программу с соответствующим именем и передает ей управление. По окончании работы программы командный процессор удаляет программу из памяти и выводит сообщение о готовности к выполнению команд (приглашение DOS).

Функции:

- прием команд с клавиатуры или из bat-файлов и их выполнение;
- выполнение команд файла **autoexec.bat** при загрузке MS-DOS;
- загрузка в ОЗУ и запуск на выполнение прикладных программ в среде MS-DOS.

Командный процессор состоит из 3 частей:

- резидентной — она размещается в ОЗУ сразу после **msdos.sys**, включает процедуры обслуживания некоторых прерываний, процедуры обработки стандартных ошибок MS-DOS, процедуру загрузки транзитной,части командного процессора;
- инициализирующей — в ОЗУ она следует сразу за резидентной частью, во время загрузки ОС ей передается управление, она выполняет файл **autoexec.bat** и некоторые другие действия. Эта часть командного процессора стирается из ОЗУ первой же загруженной программой;
- транзитной (загружается в старшие адреса ОЗУ; обрабатывает все внутренние команды, команды с клавиатуры и из **bat**-файлов; выдает системную подсказку MS-DOS, загружает в ОЗУ программы и передает им управление).

*Внешние команды (программы)* — дополнительные программы, входящие в MS-DOS, выполняющие определенные функции. Это программы, поставляемые вместе с операционной системой в виде отдельных файлов, которые выполняют действия обслуживающего характера, например форматирование дискет, проверку дисков и т. д.

*Драйверы устройств* — это специальные резидентные программы, которые дополняют систему ввода-вывода DOS и обеспечивают обслуживание новых или нестандартное использование имеющихся Устройств. Например, с помощью драйверов возможна работа с «электронным диском», т. е. с частью памяти компьютера, с которой можно работать так же, как с диском. Драйверы загружаются в память

компьютера при загрузке операционной системы, их имена указываются в файле **config.sys**. Такая схема облегчает добавление новых устройств, позволяя делать это, не затрагивая системные файлы DOS.

*Файл конфигурации системы* **config.sys.** Текстовый файл, содержащий информацию о подгружаемых дополнительных драйверах и некоторую другую информацию, касающуюся непосредственно MS-DOS и выполняемых в ее среде прикладных программ/ MS-DOS выполняет этот файл автоматически, сразу после загрузки **command.com**.

*Файл автозапуска программ при загрузке ОС* **(autoexec.bat).** Текстовый файл, содержащий дополнительную настроечную информацию. MS-DOS выполняет этот файл автоматически, сразу после выполнения **config.sys**.

***Начальная загрузка MS-DOS.*** При включении ПК вначале выполняются программы BIOS.

После тестирования и других действий процедура P08T (Power On Self Testing — самотестирование после включения питания — из модуля BIOS) осуществляет поиск и загрузку блока начальной загрузки:

- вначале производится поиск на устройстве A:;
- если не найдено — поиск на устройстве C:;
- если не найдено, то вызывается встроенный в ПЗУ BASIC или производятся другие действия, «указанные» в ПЗУ.

```
00500     512             DOS communication area
00700    1424  IBMBIO
00C90    5184  IBMDOS
020E0   35584             System data
0ABF0      64             System data
0AC40      16             Data
0AC60     336  RK         Environment
0ADC0      32             Data
0ADF0    6928             Program
0C910   10880  RK         )*+6-ю/0123456789%жБ=Ю...   43 F4
0F3A0     336             Data
0F500    3536             Program                      21
102E0    3360             Program                      33 5C 67
11010     400  CONAGENT   Environment
111B0   14704  CONAGENT   /new                      09 10 8E
14830     384  CONAGENT   Data
14CC0     416  SHELL      Data
14E70    5712  SHELL                                   22 23 24 2E 2F
16AD0    2064  SHELL      Environment
16CF0     384  MI         Environment
16E80   16352  MI                                      00
1AE70  545168             - Free -
```

Рис. 2.2. Размещение в оперативной памяти программ после загрузки MS-DOS

Блок начальной загрузки производит поиск в корневом каталоге системной дискеты (диска) файлов **io.sys и msdos.sys.**

Блок начальной загрузки производит загрузку файла **io.sys** и передает ему управление.

**io.sys** выполняет следующие действия:

- загружает и настраивает **msdos.sys**;
- определяет состояние подключенных устройств;
- инициализирует подключенные устройства;
- загружает необходимые драйверы устройств;
- передает управление **msdos.sys**.

**msdos.sys** выполняет следующие действия:

- инициализирует (настраивает) свои внутренние рабочие таблицы;
- загружает драйверы, указанные в файле **config.sys**;
- загружает командный процессор (файл **command.com**).

Командный процессор «выполняет» команды, указанные в файле **autoexec.bat,** выдает на экран монитора системную подсказку MS-DOS и ожидает команд пользователя.

Размещение в памяти (рис. 2.2):

- таблицы векторов прерываний;
- IO.SYS;
- MSDOS.SYS;
- резидентной части Command.com;
- около 530 Кбайт для прикладных программ.

***Файловые системы MS-DOS.*** Одно из основных понятий файловой системы MS-DOS — *логический диск.* В некотором приближении можно считать, что это отдельный магнитный диск. Каждый логический диск имеет свое уникальное имя.

В качестве имени логического диска используются буквы английского алфавита от A до 2 (включительно). Количество логических дисков, таким образом, не более 26. Буквы A и B отведены строго под имеющиеся в IBM PC дисководы гибких магнитных дисков (НГМД, FDD). Начиная с буквы C: именуются логические Диски (разделы) НЖМД (HDD), затем — дисководы оптических Дисков (CD KOM). В случае, если Данный компьютер имеет только один НГМД, буква B: пропускается. Только логические диски A: и C: могут быть системными (содержать модули MS-DOS).

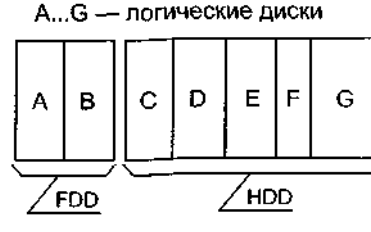



Рис. 2.3. Типичное распределение имен логических дисков

Для обеспечения доступа к файлам — файловая система MS DOS организует и поддерживает на логическом диске определенную файловую структуру.

Элементыфайловойструктуры:

- стартовый сектор (сектор начальной загрузки,Boot-сектор);
- таблица размещения файлов (PAT — File Allocation Table);
- корневой каталог(Root-Directory);
- область данных (оставшееся свободным дисковое пространство).

Эти элементы создаются утилитами в процессе инициализации диска.

Физическое размещение ОС MS-DOS: 0-й сектор — загрузчик, 1—18-й секторы — основная и дублирующая таблицы PAT, 19—20-й секторы — корневой каталог, 33—... IO.SYS, MSDOS.SYS.

*Стартовый сектор (сектор начальной загрузки, Boot-сектор).*

Здесь записана информация, необходимая MS-DOS для работы с диском:

- *идентификатор ОС* (если диск системный);
- *размер сектора* диска;
- *количество секторов в кластере*;
- количество резервных секторов в начале диска;
- количество копий PAT на диске (стандарт — 2);
- количество элементов в каталоге;
- количество секторов надиске;
- тип формата диска;
- количество секторов в PAT;
- количество секторов на дорожку;
- количество поверхностей;
- блок начальной загрузки ОС.

За стартовым сектором располагается *ГА Т.*

Команда FORMAT формирует таблицу размещения файлов (PAT) и директорию диска. Обе эти структуры тесно связаны с организацией доступа к файлам. На каждом диске имеется две копии PAT. Эта таблица имеет исключительное значение при обслуживании файлов, поэтому в случае потери первой копии PAT система получает доступ ко второй.

На стандартной дискете, имеющей по 8 секторов на дорожке, PAT занимает 1 сектор. На стандартной дискете с 9 секторами на дорожке для таблицы отводится 2 сектора.

Каждый диск имеет справочник хранящихся на диске файлов. Кроме секторов данных, используются 3 специальных сектора. Первый — содержит загрузочную запись, которая создается при Ф°Рма-

тирова..... диска, даже если он не является системным, и использу-
ется загрузке в оперативную память системных файлов про-
грамм операционной системы; два последующих сектора хранят
табли размещения секторов файлов на диске (РАТ); следующие 7
сек ов диска содержат справочник диска. Каждый элемент спра-
воч ика занимает 32 байта, поэтому в 7 секторах может храниться
112 элементов. Каждый элемент справочника содержит: имя файла,
р ширение имени, атрибуты файла (системный, только для чтения
о ) дату и время последнего изменения файла, размер файла, но-
мер начального кластера; байты 12—21 зарезервированы.

MS-DOS обеспечивает две технологии обслуживания файлов. Первая была разработана при создании версий 1.X. Эта технология основана на использовании структур данных, называемых блоками управления файлом (FCB). В то время подавляющее большинство компьютеров работало под управлением операционной системы CPM. Блоки FCB обеспечивали совместимость файлов MS-DOS с файлами этой системы. При разработке MS-DOS версий 2.X, когда была предложена иерархическая структура организации файлов, была разработана вторая технология их обслуживания. Она основана на использовании ссылок на управляющую запись файла и не требует организации FCB. После того как эта технология была опробована в операционной системе UNIX, она получила широкое распространение.

***Управление оперативной памятью.*** Память состоит из отдельных элементов, каждый из которых предназначен для хранения минимальной единицы информации — *одного байта.* Каждому элементу соответствует уникальный числовой адрес. Первому элементу присвоен адрес 0, второму — 1 и т. д., включая последний элемент, адрес которого определяется общим количеством элементов памяти минус единица. Обычно адрес задается шестнадцатеричным числом (в тексте шестнадцатеричные числа помечаются заглавной «Н», например ЮН).

*Сегменты.* Процессор компьютера делит память на блоки, называемые сегментами. Каждый сегмент занимает 64 Кбайт и каждому сегменту соответствует уникальный числовой адрес. Процессор имеет четыре регистра сегмента. Регистр — это участок сверхоперати й памяти процессора, предназначенной для хранения информац . Регистры сегмента предназначены для хранения адресов отдельных сегментов. Они называются CS (сегмент кода), DS (сегмент данных), 38 (сегмент стека) и ES (запасной сегмент).

К ме указанных, процессор имеет еще 9 регистров, а именно регистры IP (указатель команды) и SP (указатель стека). Реги-

стры C8 и IP в паре составляют длинный адрес команды, которая будет выполняться следующей. Регистры 88 и 5P в паре составляют длинный адрес стека.

*Доступ к памяти.* Доступ к ячейкам памяти осуществляется посредством соединения содержимого регистра сегмента с содержимым того или другого регистра. Таким образом определяется адрес требуемого участка памяти. Например, адрес следующей команды определяется содержимым регистров C8 и IP (записывается «CS:IP»). После выполнения команды и ее удаления из памяти содержимое IP изменяется так, чтобы в регистрах C8 IP находился адрес команды, которая будет выполнена после данной.

Способ объединения регистров для определения адреса ячейки памяти не накладывает ограничений на количество доступной памяти. Верхнее ограничение зависит от физического строения памяти (т. е. от общего количества ячеек). Первые версии MS-DOS разрабатывались для процессора Intel 8088 CPU. Каждый регистр этого процессора рассчитан на хранение 16-битового числа. То есть CPU 8088 комбинирует содержимое сегментного регистра (скажем, C8) с содержимым другого регистра (скажем, IP), получая 20-битовый адрес памяти, что ограничивает доступную память до $2^{20}$ байтов или 1 Мб.

Позже появились усовершенствованные процессоры CPU 80286 и 80386 и соответственно им усовершенствованные версии M8-DOS, позволяющие производить доступ к ячейкам, расположенным за пределом 1 Мб памяти.

Доступ к памяти организуется соединением содержимого одного из регистров сегмента с содержимым одного из оставшихся регистров. Значение сегментного регистра называется адресом сегмента. Значение остальных регистров в этом случае называется относительным адресом ячейки памяти (от начала сегмента) или ее коротким адресом. Таким образом, адрес байта вычисляется посредством умножения адреса сегмента на 16 и к полученному значению добавляется короткий адрес.

*Сегментные регистры.* Сегментные регистры используются при идентификации сегмента памяти. Сегментные регистры применяются в комбинации с регистром указателя или индексными регистрами и в этом случае идентифицируют конкретную ячейку памяти.

Всего сегментных регистра четыре. Регистр C8 обычно используется при идентификации блока памяти, в котором хранится код программы. Регистр DS — при идентификации участка памяти, в котором находятся данные этой программы. С помощью регистра 88 организуется доступ к стеку. (Стек — это временно выделенная область памяти, обеспечивающая интерфейс «MS-DOS-прикладная

программа».) Регистр E8 — дополнительный (или запасной) сегментный регистр. На него возложены разнообразные функции, часть из которых рассматривается ниже.

*Регистры стека.* Имеются два регистра стека. Они применяются в комбинации с регистром 53 и определяют местонахождение стека. Регистр 8P называется указателем начала стека и в комбинации с регистром 88 идентифицирует первый байт стека. Регистр BP называется указателем базы стека и в комбинации с регистром 88 идентифицирует последний байт стека.

*Индексные регистры.* Индексных регистров тоже два. Регистры SI и DI применяются в комбинации с одним из сегментных регистров и определяют местонахождение конкретной ячейки памяти. Регистр SI обычно комбинируют с регистром DS, регистр DI — с регистром E8.

*Регистры общего назначения.* К регистрам общего назначения (их четыре) относятся регистры AX, BX, CX и DX. Это многофункциональные регистры.

*Регистр указателя команды.* Регистр IP обычно применяется в комбинации с регистром C8 и определяет адрес следующей команды.

*Регистр флагов состояния.* В регистре флагов обычно находятся девять флагов состояния процессора (каждый флаг занимает 1 бит). Эти флаги определяют результат конкретных операций, выполняемых под управлением MS-DOS.

*Регистры памяти.* Регистр памяти включает 2 байта данных (или 16 битов). Реально регистры общего назначения однобайтные. Так, регистр AX включает регистр AH (который составляет старший байт регистра AX) и регистр AL (который составляет младший байт регистра AX). Аналогично регистры BH, BL, CH, CL, DH и DL — однобайтные.

*Драйверы MS-DOS.* Два важнейших компонента электронного оборудования компьютера — центральный процессор и память. Остальные компоненты (дисководы, клавиатура, дисплеи, принтеры и т. д.) являются внешними по отношению к компьютеру. Эти внешние компоненты электронного оборудования называются *периферийными устройствами* или просто *устройствами.*

Связь машины с периферийным устройством осуществляется в строго определенном порядке. Каждому периферийному устройству в операционной системе соответствует программа, отвечающая за его контакт с компьютером. Эти программы называются *драйверами.*

Одна из основных функций операционной системы — это обеспечение работоспособности драйверов, доступных системным и прикладным программам. Если работающей программе необходим

контакт с периферийным устройством, то она сообщает операционной системе, какое из устройств ей необходимо, и MS-DOS предоставляет ей соответствующий драйвер.

*Устройства посимвольной и поблочной передачи данных.* Устройства посимвольной передачи данных осуществляют пересылку информации по одному символу. К этим устройствам относятся порты последовательных и параллельных адаптеров и дисплеи. В MS-DOS каждому из этих устройств соответствует конкретное название (имя). Драйвер MS-DOS может управлять только одним устройством посимвольной передачи. Устройства поблочной передачи данных осуществляют пересылку информации блоками. Каждый блок, как правило, составляет 512 байт. К этим устройствам относятся дисководы для гибких дискет, дисководы для жесткого диска и другие накопители информации. Устройства поблочной передачи не обладают конкретным названием. Драйвер MS-DOS может обслуживать несколько устройств поблочной обработки данных.

## 2.2. Графические программные оболочки Windows 3.x

***Операционная оболочка Windows 3.1.*** Операционная оболочка Windows 3.1 — надстройка над DOS, обеспечивающая более удобный и наглядный интерфейс для пользователей (графический интерфейс), т. е. набор средств для вывода изображений на экран и манипулирования ими, построения меню, окон на экране и т. д., мультипрограммирования (т. е. возможность одновременного выполнения нескольких программ), имеющая расширенные средства для обмена информацией между программами [16].

В течение долгих лет с момента своего появления персональные компьютеры (IBM-совместимые) обходились без специальных «пользовательских оболочек», работая непосредственно под управлением операционной системы (MS-DOS, DR DOS, PC DOS...). Пользователи, садившиеся за такой компьютер, видели после включения на пустом экране только подсказку `C:\>`. Все операции управления компьютером производились путем ввода с клавиатуры некоторых слов-директив. Неудобство такого алфавитно-цифрового интерфейса порождало претензии и к самим компьютерам (возможно, и не совсем обоснованные).

Версия 3.0 оболочки Windows (и появившаяся следом 3.1) исповедует совершенно другие принципы в части интерфейса пользователя с ЭВМ. (Можно считать эти принципы новыми, но машины

фирм Apple строятся на этих принципах уже в течение многих л новная идея, заложенная в основу оболочки Windows, — есте ственность представления информации. Информация должна пред ться в той форме, которая обеспечивает наиболее эффектив усвоение этой информации человеком. Несмотря на простоту же тривиальность) этого принципа, его реализация в интерфе сах прикладных программ персональных ЭВМ по разным причинам оставляла желать лучшего. Да и реализация его в рамках Windows 3.1 тоже не лишена недостатков. Но эта оболочка представляет собой существенный шаг вперед по сравнению с предыдущими интерфейсами пользователя с ЭВМ. Наиболее важными отличительными чертами ее являются следующие:

- *Windows представляет собой замкнутую рабочую среду.* Практически любые операции, доступные на уровне операционной системы, могут быть выполнены без выхода из Windows. Запуск прикладной программы, форматирование дискет, печать текстов — все это можно вызвать из Windows и вернуться в Windows по завершении операции. Опыт работы в DOS пригодится и здесь: многие основополагающие принципы и понятия среды Windows не отличаются от соответствующих принципов и понятий среды DOS;
- *основными понятиями пользовательского интерфейса в среде Windows являются окно и пиктограмма.* Все, что происходит в рамках оболочки Windows, в определенном смысле представляет собой либо операцию с пиктограммой, либо операцию с окном (или в окне). Стандартизована в среде Windows и структура окон и расположение элементов управления ими. Стандартизованы наборы операций и структура меню для сервисных программ. Стандартны операции, выполняемые с помощью мыши для всех сервисных и прикладных программ;
- *Windows представляет собой графическую оболочку.* От пользователя не требуется ввод директив с клавиатуры в виде текстовых строк. Необходимо только внимательно смотреть на экран и выбирать из предлагаемого набора требуемую операцию с помощью манипулятора мышь. Курсор мыши следует позиционировать на поле требуемой директивы меню, или на интересующую пиктограмму, или на поле переключателя (кнопки). На выбранном объекте необходимо зафиксировать курсор кнопкой мыши — и операция выполняется. С помощью того же манипулятора можно перемещать пиктограммы и окна по экрану, менять их размер, открывать и закрывать их — и все это при минимальном использовании клавиатуры для ввода

каких бы то ни было директив. Кроме того, для любителей традиционного интерфейса DOS реализована возможность выхода на этот уровень. Понятие «графически-ориентированный» включает в себя для Windows также и соответствие изображения на экране последующему изображению на твердой копии (распечатке). В этом плане можно считать, что в оболочке Windows реализован принцип WYSIWYG (What You See Is What You Get — То, что вы видите, то и получаете), до сих пор бывший привилегией относительно небольшого числа программ. С помощью TrueType-шрифтов этот принцип нашел в рамках Windows 3.1 свое дальнейшее развитие;

- *Windows обеспечивает независимый запуск и параллельное выполнение нескольких программ.* Большинство других оболочек и операционных систем рассчитано на выполнение в данный момент только одной программы. В рамках Windows пользователь может запустить несколько программ для параллельного (независимого) выполнения. Каждая из выполняемых программ имеет свое собственное окно. Переключение между выполняемыми программами производится с помощью мыши фиксацией курсора в окне требуемой программы;
- *Windows — интегрированная программа.* Под управлением оболочки Windows могут работать не только специальные программы, разработанные для эксплуатации в среде Windows (Windows-приложения), но и «обычные» программы, работающие в среде DOS, т. н. DOS-приложения (DOS-прикладные программы). Оболочка Windows обеспечивает эффективный и комфортабельный обмен информацией между отдельными программами, выполняемыми под ее управлением. Здесь речь в первую очередь идет о Windows-приложениях. С понятием «интегрированное» связывают обычно также возможность совместного использования ресурсов компьютера различными программами. Например, принтер, подключенный к компьютеру, может с одинаковым успехом использоваться всеми программами на конкурентной основе. Причем все операции, связанные с необходимостью перекодировок, смен драйверов (например, при переходе от печати текстов к выводу иллюстраций) берет на себя оболочка.

Существенно упростилась работа с документами вообще. Можно говорить о документо-ориентированной организации работ. При этом можно расположить пиктограмму часто используемого документа в окне Диспетчера Программ (Ргодгат Мападег) и в дальнейшем вызывать процесс обработки этого элемента (например, Редак-

тирова ие) просто двойной фиксацией данной пиктограммы. Той же це жит и возможность автоматического запуска Диспетчера файл после загрузки оболочки — перед пользователем сразу же откры ается поле выбора файлов документов.

С среде Windows 3.1 можно составлять документы из частей, кото готовятся в различных приложениях, но при этом сложность раб ты с таким документом не выше, чем если бы он готовился в р одного приложения. Так, работая в новой версии Windows, вставке в текст, подготовленный в Write, рисунка, созданного в р intbrush, рисунок рассматривается как объект. Он может сохра- няться загружаться и печататься совместно с документом. Главной особенностью такого связывания рисунка и текста является простота внесения изменений. Например, работая с текстом в редакторе Write, достаточно выполнить двойную фиксацию курсора на рисунке, чтобы вызвать графический редактор Paintbrush. Рисунок загрузится в него автоматически. Все внесенные после этого в рисунок изменения автоматически отобразятся и во вставке в текстовой документ. Создается впечатление, что текстовой редактор Write снабжен дополнительными возможностями редактирования рисунков (в полном объеме Paintbrush).

Работа с объектами предусматривает также и операции с пиктограммами. Пиктограммы можно использовать внутри документов для выполнения функций, подобных тем, которые эти пиктограммы выполняют в среде Windows.

В среде Windows 3.1 реализован новый набор шрифтов — так называемые True Type-шрифты. Эти шрифты похожи на PostScript-шрифты, но легко адаптируются после небольшой настройки практически к любому типу принтера. Небольшими усилиями можно Добиться успеха в работе с этими шрифтами в большинстве Windows-приложений. TrueType-шрифты легко поддаются масштабированию, различного рода деформациям, вращению и т. п. Дополнительный комфорт для любителей выбирать и создавать шрифты ни спечивает специальная программа демонстрации и использова- я отдельных литер шрифтов — Charater Map.

мул аконец, в оболочку Windows 3.1 включены две небольшие гает имедиа-программы (Multimedia). Их использование предпола- аличие специальной аппаратной поддержки (акустический адаптер, возможно, накопитель CD-ROM). С помощью упомянутых программ, называемых Sound Recorder и Media Player, можно оформить п цесс прохождения программ звуковыми эффектами. Можно раб ать с цифровым представлением речи и музыки, с картинками, п лученными, например, с проигрывателя видеодисков.

Для начинающих в системе предусмотрена обучающая программа, к которой можно обратиться уже на этапе инсталляции.

*Суммируем основные преимущества Windows.*

1. Независимость программ от внешних устройств (монитора, клавиатуры, принтера). Драйверы для поддержки этих устройств входят в состав Windows или поставляются вместе с указанными устройствами.

Windows — программы могут обращаться к внешним устройствам только через Windows, а DOS — программы обращаются к устройствам, минуя DOS. Это снимает с разработчиков проблему совместимости с конкретным внешним устройством.

2. Имеются средства для построения пользовательского интерфейса программ (окна, меню, запросы, списки программ и т. д.).

3. Доступность всей оперативной памяти (а не 640 Кбайт, как в DOS).

4. Динамическое подключение библиотек (dll-файлов). Библиотеки расширяют возможности Windows и могут быть вызваны автоматически любой Windows-программой.

5. Обмен данными между приложениями Windows.

6. Использование масштабируемых шрифтов типа Тгие Type.

7. Организация встроенных справочников программ.

8. Единый пользовательский интерфейс.

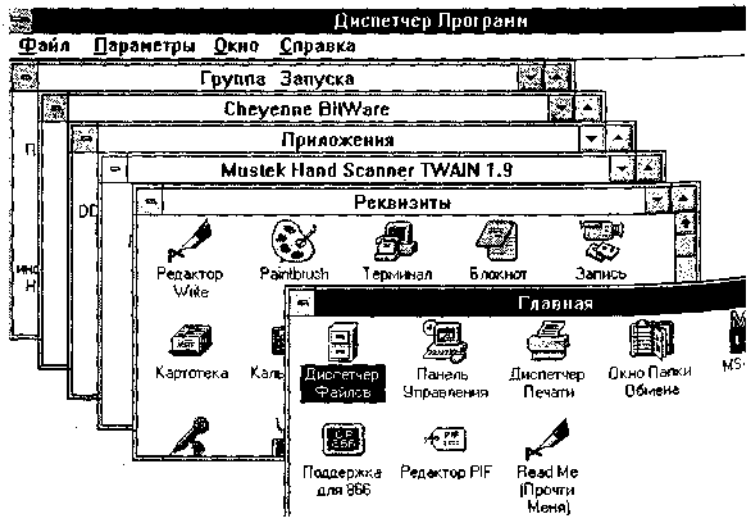



Рис. 2.4. Многооконный интерфейс, типичный для Windows 3.1

9. Многозадачность — одновременное выполнение нескольких программ, переключение с одной задачи на другую, управление приоритетами выполняемых программ.

10. Совместимость с DOS-приложениями. Многие DOS-программы запускаются под управлением Windows, но работают медленнее.

11. Поддержка мультимедиа (подключение CD-ROM, медиаплеера, микрофона, видеокамеры и других средств обмена информацией с окружающей средой).

***Окна в Windows.*** Рассмотрим способы управления окнами и пиктограммами в Windows 3.1, их структуру и назначения. Эти сведения помогут глубже понять принципы функционирования оболочки и станут основой для осмысленного оперирования пиктограммами и окнами как основными элементами среды Windows.

Если оболочка Windows запущена и работает, то общение пользователя с ней происходит посредством многооконного интерфейса _ через систему окон. Окна или представляющие их пиктограммы расположены на поле экрана, как бумаги на рабочем столе (кстати, поле экрана называется Desktop — поверхность стола). Причем, очевидно, что речь идет о маленьком столе, на котором много бумаг, лежащих «в беспорядке».

Посредством окон и пиктрграмм выполняются все манипуляции с программами и файлами документов в среде Windows. Можно запустить одно или даже несколько приложений одновременно и в открывшихся окнах наблюдать процесс их выполнения. (Если продолжить нашу аналогию, то этому соответствует вытаскивание из ящиков стола различных бумаг и разворачивание их на столе.) Можно «переключать внимание» с одного окна на другое. Конфигурация и расположение окон производятся в соответствии с личными вкусами и потребностями.

*Окно.* Если предельно упростить проблему, то можно сказать, что новинкой оболочки Windows является окно как элемент экрана. В среде Windows мы видим через окна (само название оболочки Windows переводится как «Окна»), начиная от отдельных приложений, работающих под управлением Windows, и кончая самой оболочкой.

Каждое окно в обязательном порядке содержит поле заголовка и рабочее поле (или поле индикации).

*Заголовок и рабочее поле окна.* Каждое окно представляет собой ограниченную рамкой часть поверхности экрана. Оно может иметь различные размер и находиться в разных местах экрана. Внутри каждого окна что-нибудь показывается (приложением или систе-

мой) или что-нибудь делается самим пользователем. Другими словами, окно представляет собой пространство для:

- размещения объектов (текста, рисунков, пиктограмм и других окон);
- выполнения действий (написание текста, рисование, ввод директив, вывод сообщений).

Как уже известно, окно может существовать в полноэкранном представлении, занимая все поле экрана, или в нормальном представлении, занимая только его часть, или в виде пиктограммы. Почему стандартизованы именно эти три представления?

Один из основных принципов, на котором базируется оболочка Windows и который лег в основу стандартизации способов представления окон, чрезвычайно прост: окна должны служить целям простого и наглядного оперирования с несколькими параллельно работающими приложениями.

В среде Windows предусмотрены окна двух типов:

- окна, в которых выполняются приложения (прикладные окна);
- окна, подчиненные другим. В них не выполняется никаких приложений. Они служат для индикации документов или пиктограмм и называются окнами документов или групповыми окнами.

Окно любого типа может быть увеличено до полноэкранного или уменьшено до нормального размера, а также свернуто до пиктограммы (причем существуют различные типы пиктограмм, соответствующие типам окон). Подчиненное окно не может быть выведено за пределы окна-хозяина или превысить его по размеру.

*Прикладное окно.* Большинство окон, с которыми имеет дело оболочка Windows, — прикладные окна, в которых выполняются конкретные приложения. Распознать прикладное окно легко по двум отличительным признакам (рис. 2.5):

- в заголовке указывается имя приложения, которому принадлежит это окно;
- под заголовком расположена еще одна строка, называемая строкой меню, в которой перечисляется ряд операций, доступных приложению. Как правило, первая из операций — File (Работа с файлами).

Если запустить двойным щелчком приложение Write, представляющее собой редактор текста и входящее в группу Accessories, то открывающееся окно в точности совпадает по виду с приведенным выше на иллюстрации. Но оно, в отличие от уже опробованных приложений, не содержит никакой информации, а только чистое

рабочее поле (что, кстати, характерно для большинства приложений). Назначение рабочего поля зависит от приложения: в одних из них — рисуемая посредством мыши картинка, в третьих — результаты каких-то расчетов.

*Подчиненное окно (групповое окно).* Внутри окна Менеджера (диспетчера) программ видно по крайней мере еще одно окно — подчиненное окно Реквизиты (Accessories).

Отличить групповые и подчиненные окна от окон прикладных легко по следующим признакам:

- в заголовке такого окна стоит не имя приложения, а специальное имя, которое вы можете задать сами. Специальным является и содержание такого окна. Так, подчиненные Диспетчеру Программ окна содержат ряд (группу) пиктограмм — они называются групповыми окнами. Заголовки окон, подчиненных другим приложениям, как правило, содержат имена документов, которые располагаются в этих окнах. По этой причине такие подчиненные окна называются также окнами документов;
- подчиненные окна не имеют меню;
- третий отличительный признак хорошо согласуется с названием «подчиненное окно». Он заключается в том, что подчиненные окна могут перемещаться только внутри окон, которым они подчиняются. Это же касается и операций изменения размера и представления.

Наконец, для групповых окон существует особая клавиатурная комбинация закрытия <Ctrl+F4>.

Если при инсталлировании была заказана полная конфигурация оболочки, то в окне Диспетчера Программ имеется пять подчиненных групповых окон:

- Main — главная группа;
- Games — игры (нет на рис. 2.5);
- StartUp (Группа запуска) — группа автоматически запускаемых при загрузке оболочки программ;
- Accessories (Реквизиты) — группа сервисных программ, инструментарий;
- Applications (Приложения) — группа дополнительных программ (обычно DOS-приложений).

Кроме того, на рис. 2,5 расположены еще 2 программные группы, созданные пользователями (Mustek Hand Scanner и BitWare).

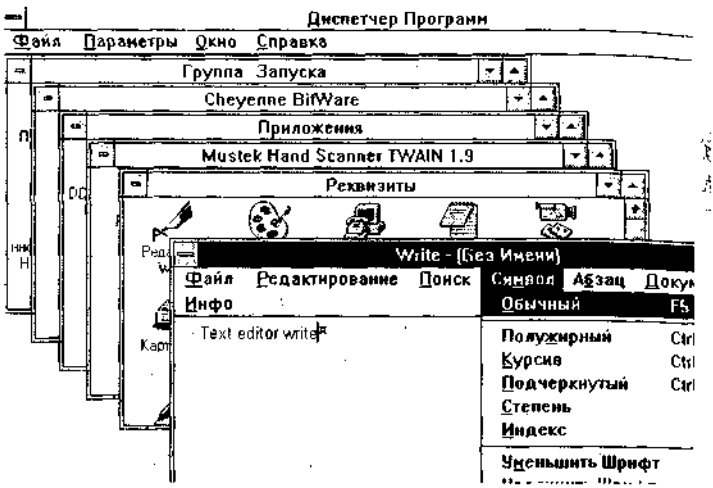



Рис. 2.5. Пиктограмма прикладного окна текстового редактора Write

В любом случае в окне Диспетчера Программ будут находиться либо сами эти групповые окна, либо их пиктограммы. На рис. 2.4, 2.5 приведены все четыре окна в открытом состоянии.

Представления групповых окон можно выбирать по своему усмотрению. Можно смешать на одном экране различные представления для разных групп. Очевидно, что проблемы ограниченной площади «рабочего стола» приводят к тем же результатам для групповых окон, что и для окон прикладных. Одни окна могут накладываться на другие, вплоть до полного сокрытия последних.

***Пиктограммы*** располагаются внутри групповых окон. Они всегда находятся на одном и том же месте в групповом окне, независимо от того, запущено ли соответствующее им приложение или нет. Сопоставляя внешний вид этих пиктограмм с групповыми пиктограммами, можно сделать вывод о том, что окнам разного типа соответствуют пиктограммы разного вида, или о том, что существует несколько разновидностей пиктограмм.

*Групповые пиктограммы.* Если свернуть некоторое групповое окно (например, Main) до состояния пиктограммы путем фиксации курсора мыши на переключателе пиктограммы (поле справа от заголовка со стрелкой вниз), то на экране появится следующая пиктограмма группы Main.

Групповая пиктограмма — форма представления группового окна. Она располагается всегда в пределах окна-хозяина (если она

не видна, то вероятно, ее загораживает некоторое окно). Подобные пикто р существуют еще в пределах окна Диспетчера Файлов.

Пиктограммы всех групповых окон одинаковы и отличаются только подписями.

*Программные пиктограммы.* Особенностью групповых окон является то что они содержат в себе не рабочее поле или документ, а набор пиктограмм, посредством которых запускаются приложения. Внешний вид этих пиктограмм, называемых программными пиктограммами не отличается от тех пиктограмм, которые возникают в нижней части экрана при сворачивании прикладных окон.

Для запуска приложения по программной пиктограмме следует выполнить на ней двойную фиксацию курсора мыши.

Программные пиктограммы также существуют только в пределах Диспетчера Программ.

*Пиктограммы приложений (прикладные пиктограммы).* Выше на примере некоторых приложений (часов, калькулятора и редактора текстов) мы уже познакомились с внешним видом и расположением прикладных пиктограмм. Также известно, что они — результат сворачивания прикладных окон. На иллюстрации показана ситуация на экране после сворачивания прикладного окна редактора текстов.

Прикладная пиктограмма выглядит точно так же, как и программная пиктограмма, по которой запускалось данное приложение. Но в подписи к прикладной пиктограмме, как правило, присутствует наименование обрабатываемого приложением документа. В приведенном на иллюстрации примере редактором текстов обрабатывается документ с именем, установленным по умолчанию, — [Без имени] (Untitled). Такая форма подписи (с указанием документа) является признаком прикладной пиктограммы.

Пиктограммы всех трех видов обладают общим свойством: при выполнении двойной фиксации на них открывается соответствующее окно. Но содержимое и назначение открываемого окна в каждом случае свои:

- групповая пиктограмма открывает групповое окно;
- программная пиктограмма открывает окно приложения и запускает соответствующую программу;
- прикладная пиктограмма открывает окно уже работающего приложения (меняет представление прикладного окна).

***Составные части окна.*** Все окна независимо от типа имеют одинаковый состав и структуру. Но не во всех окнах присутствуют все возможные составные элементы. Запустим еще раз текстовой

редактор и рассмотрим структуру прикладного окна в нормальном представлении.

У этого окна простая рамка только с одной стороны — слева. Сверху к рамке примыкают две служебные строки, снизу — одна. К правой стороне рамки примыкает служебная колонка. Рамка вместе со служебными строками и колонкой ограничивает рабочее поле окна.

При переводе окна в полноэкранное представление левая сторона рамки исчезает, а с трех других сторон остаются только служебные элементы. Служебные элементы позволяют управлять состоянием окна, причем расположенные сверху служебные элементы управляют представлением и содержимым окна, расположенные сбоку и/или снизу служебные элементы позволяют просматривать в маленьком окне большие документы.

*Заголовок*. Верхняя строка окна (в данном случае — прикладного окна редактора текста) называется заголовком. Она присутствует во всех окнах. Если окно активно, то заголовок этот содержит надпись «белым по черному», т. е. светлым шрифтом на темном фоне. В противном случае надпись в строке заголовка темная на светлом фоне.

На левом конце строки заголовка находится поле (кнопка) вызова системного меню, или системный переключатель. Он имеется во всех окнах. Если щелкнуть на этом поле, то появляется системное меню, содержащее ряд директив управления окном. Это же меню можно вызвать и фиксацией курсора мыши на пиктограмме. С помощью этого меню можно управлять окном и, следовательно, выполняемой в нем программой. Посредством системного меню можно:

- изменить представление и местоположение окна;
- закрыть окно;
- переключиться на другое окно.

Мы еще вернемся к этим средствам управления окном в этой и следующей главе, а пока продолжим разбор структуры окна.

В центре строки заголовка находится имя приложения, выполняемого в данном окне (для прикладных окон), или имя группы (для групповых). Следом за именем приложения в заголовке прикладного окна идет имя обрабатываемого документа (прописными буквами). Например: Write — MEMO.WRI или Paintbrush — WINLOGO.BMP.

На правом конце строки заголовка располагаются два поля размерных переключателей, позволяющих быстро управлять размером (представлением) окна. Левый переключатель иногда называют еще

переключателем пиктограммы. Он содержит изображение стрелки вниз. Фиксацией курсора на этом поле можно свернуть окно до пиктограммы. Вид правого переключателя . зависит от представления окна — полноразмерное или нормальное (подробнее см. ниже).

Можно менять размер и перемещать окно, используя поле заголовка. Достаточно зафиксировать на нем курсор и, не отпуская кнопки мыши, переместить его в нужном направлении — окно переместится следом. Двойная фиксация на поле заголовка меняет представление окна с полноэкранного на нормальное и обратно.

*Строка меню.* В окне редактора под строкой заголовка находится еще одна служебная строка — строка меню. Она содержит ряд элементов, начинающийся словом File. Если в окне имеется строка меню, то это прикладное окно.

Строка меню позволяет управлять приложением, выполняемым в окне. При фиксации курсора мыши на элементах этой строки, как правило, открываются дополнительные меню (подменю), содержащие директивы обслуживания приложения. Но об этом подробнее в следующей главе.

*Линейки прокрутки.* Левую и нижнюю сторону окна окаймляют специальные сервисные средства, называемые линейками прокрутки.

Эти линейки позволяют прокручивать в окне документ подобно рулону. Это необходимо, когда размеры документа превышают размеры окна. Линейки прокрутки в большинстве приложений появляются автоматически сразу же по открытии окна. Иногда наличие линеек прокрутки (особенно для случая групповых окон) позволяет судить о наличии элементов изображения (для групповых окон — пиктограмм) за пределами поля зрения в окне. Чтобы их увидеть, следует подвигать изображение, пользуясь линейками прокрутки.

Прокручивать изображение в окне можно, щелкая на полях стрелок, расположенных по краям линеек (окно перемещается по документу в указываемом стрелкой направлении), или перемещая (проскивая) белый прямоугольник прокрутки (маркер прокрутки) по линейке с помощью мыши.

*Стандартный вид указателя мыши.* В процессе работы с оболочкой (например, при перемещении по экрану) вид курсора мыши может и меняться. Это может быть двойная стрелка различной направленности, черточка, дорожный знак, часы или некоторая пиктограмма.

Курсор мыши указывает на то место, где находится кончик стрелки. В зависимости от точки, на которую указывает курсор

мыши, пользователю предоставляется возможность вызвать то или иное действие, например маркировать или переместить некоторый объект.

Маркер перемещается по экрану синхронно с перемещением манипулятора по поверхности стола. Манипулятор можно перемещать в любом направлении — влево, вправо, вверх, вниз и по диагонали. Скорость перемещения курсора по экрану (точнее, скорость реакции его на перемещения манипулятора) может изменяться с помощью специальных средств конфигурирования мыши, которые сосредоточены в приложении Control Panel (Панель Управления).

Для вызова тех или иных действий в большинстве случаев используется левая кнопка манипулятора. Иногда (в особенности в специальных приложениях) используется и правая кнопка. Кроме того, они могут использоваться и совместно. Главные манипуляции, доступные мыши, уже известны:

- фиксация курсора (щелчок) — кратковременное нажатие на левую кнопку мыши;
- двойная фиксация (двойной щелчок) — двухкратное с коротким промежутком кратковременное нажатие левой кнопки мыши;
- перетаскивание (транспортировка) — нажатие левой кнопки мыши и перемещение манипулятора при нажатой кнопке. Перетаскивание заканчивается при освобождении кнопки мыши.

*Специальные действия.* Ориентация оболочки Windows на работу с манипулятором обеспечивает пользователю доступ к достаточно сложным операциям посредством сравнительно простых манипуляций. Простым щелчком можно открыть и закрыть окно или меню. Многие приложения используют пиктографический интерфейс, позволяющий выполнять определенные действия при фиксации курсора на соответствующих пиктограммах.

Двойной щелчок на программной пиктограмме запускает соответствующую программу или открывает ее окно. Тот же двойной щелчок на заголовке окна меняет его представление. Двойная фиксация курсора на имени файла в диалоговом окне загрузки файлов позволяет открыть соответствующий документ.

Важной манипуляцией является перетаскивание (транспортировка). Она используется не только для перемещений окон или пиктограмм. При работе в рамках приложений эта операция используется для выделения фрагментов текста или перемещения их. Весьма полезной является также предусмотренная в среде операция Drag & Drop (Перетащи и Положи). Например, при перетаскивании пиктограммы некоторого документа и «наложении» ее

на пиктограмму Менеджера Печати (Print Manager) соответствующий файл будет отпечатан.

Если мышь используется при работе с оболочкой и с Windows-приложениями, то в первую очередь необходимо уяснить доступный набор операций с помощью стандартных манипуляций (двойной фиксации, перетаскивания).

Существуют наряду с мышью и другие устройства указания, в том числе и с похожим принципом действия. К ним в первую очередь следует отнести шаровой манипулятор Trackball, который можно рассматривать как «перевернутую мышь». Этот манипулятор неподвижен. Но выступающий над его корпусом шарик (размером с биллиардный) можно «прокатить» ладонью в любом направлении. Расположенные на корпусе манипулятора кнопки служат тем же целям, что и у мыши. Неподвижность шарового манипулятора позволяет эффективно использовать его, когда свободное рабочее пространство на столе ограничено. Этот вид манипулятора часто используют также в носимых компьютерах Laptop и Notebook (ориентированных на работу «на коленях»).

*Меню.* Набор средств управления оболочкой Windows не ограничивается пиктограммами, линейками прокрутки и прочими графическими элементами. Эффективным средством управления, доступным практически из любого окна и пиктограммы, является меню. В общем случае меню представляет собой прямоугольное поле, в котором перечислены доступные операции (или доступные для установки опции). Как правило, обозначения элементов меню состоят из одного-двух слов.

Меню практически постоянно находятся в свернутом состоянии и на экране присутствуют только их заголовки (в строке меню прикладного окна). А может не быть и их (если речь идет о системном меню прикладной пиктограммы). Меню, вызванное из строки меню, разворачивается вниз, а вызванное из прикладной или групповой пиктограммы, расположенной у нижнего края окна, — вверх. Для вызова любой из перечисленных в меню директив достаточно щелкнуть на ней мышью.

В рамках Windows существует две разновидности меню: системное и операционное меню. Для получения доступа к системному меню (вызова системного меню) следует щелкнуть мышью на кнопке системного меню. Она располагается всегда в левом верхнем углу окна (в строке заголовка). Форма кнопки системного меню напоминает форму продолговатой клавиши пробела, и этой ассоциацией можно воспользоваться, чтобы запомнить второй способ вызова системного меню (клавиатурный) <Alt+Space>.

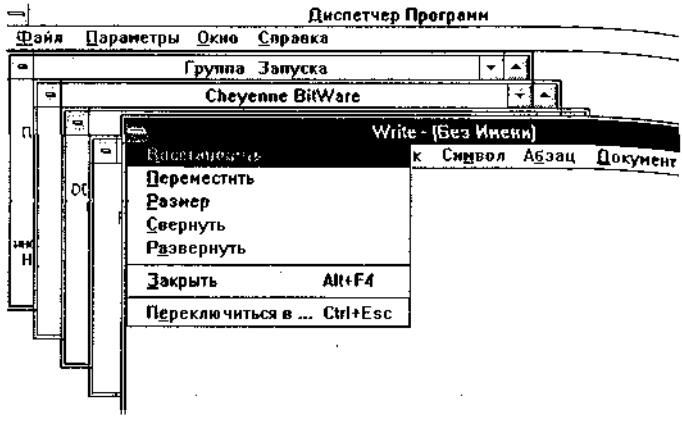



Рис. 2.6. Пример системного меню (редактор Write)

Меню содержит ряд директив, в большинстве своем предназначенных для изменения размеров и представления данного окна.

Посредством директив Свернуть (Minimize) и Развернуть (Maximize) можно менять представление окна точно так же, как с помощью размерных переключателей, расположенных в правом верхнем углу окна. Директива Восстановить (Restore) доступна только в случае представления окна в виде пиктограммы или в полноэкранном варианте. Она обеспечивает восстановление нормального представления окна из пиктограммы или из полноэкранного представления.

Воспользовавшись директивами Restore и Maximize из этого меню, можно превратить пиктограмму в нормальное или полноэкранное окно соответственно.

Директивы Move (Перемещение, Перетаскивание, Транспортировка, Буксировка) и Size (Размер) позволяют с помощью клавиш управления курсором менять соответственно положение и размер окна. Если щелкнуть на директиве Move, то курсор примет вид четырехсторонней стрелки. Теперь с помощью клавиш управления курсором можно перемещать окно. Для закрепления его на новой позиции следует нажать <Enter>.

Аналогичным образом производится изменение размеров по директиве Size. После щелчка на ней курсор превращается в четырехстороннюю стрелку. Следует выбрать ту кромку окна, которую нужно переместить для изменения его размера. Выбор осуществляется нажатием соответствующей клавиши управления курсором. После этого курсор превращается в расположенную на выбранной кромке двунаправленную стрелку, показывающую возможные направления

е перем щения. Клавишами управления курсором можно переме-
щать кромку окна, а зафиксировать ее на новом месте можно с по-
мощью клавиши <Enter>.

Д ктива Закрыть (Close) закрывает окно и завершает работу с
данным приложением (в данном случае с Write).

Самой нижней в системном меню является директива Переключить в (Switch To...), позволяющая переключиться на другие активные в данный момент задачи или на Менеджер Программ.

***Диспетчер Программ Windows (ДП, Program Manager)*** предназначен для запуска и завершения программ Windows, переключения между программами, имеет меню, содержащее 4 пункта: Файл Параметры, Окно, Справка (рис. 2.5). Меню — это список выполняемых команд или действий.

P1 — вызов справочной системы, получение помощи в Windows. Для лучшей структуризации имеющихся программ ДП позволяет объединять программы в группы.

Как правило, ДП используется в качестве оболочки Windows, то есть программы, запускаемой сразу после старта Windows. В этом случае при выходе из ДП происходит и выход из Windows. ДП может быть свернут (минимизирован) в пиктограмму на экране Windows. Для разворачивания ДП в виде окна надо дважды щелкнуть по его пиктограмме мышью или щелкнуть один раз и указать пункт Развернуть.

При помощи ДП можно организовать приложения и файлы по смысловым программным группам. ДП выполняет следующие функции с помощью директив в пункте меню Файл:

1. Упорядочивание окон и пиктограмм.
2. Изменение свойств программных групп и программных элементов.
3. Копирование и перемещение программных элементов.
4. Создание и удаление программных групп и программных элементов.

С помощью пункта меню Окно можно выбрать и открыть любое из предложенного списка окон, в том числе и Дополнительные окна. Если программные группы расположены на рабочем столе Диспетчера Программ хаотично, то их можно упорядочить, используя меню Окно, Упорядочить. Аналогично поступают, если программные элементы расположены в окне беспорядочно.

Программы, которые запускаются ДП, отображаются в его окне в виде пиктограмм, которые объединены в программные группы. Главное окно ДП называется Рабочим столом Windows (DeskTop). Программные группы представлены внутри ДП в виде окон (они

свернуты в пиктограммы, которые все одинаковы и отличаются только подписями). Программные элементы, находящиеся в окнах (программных групп), имеют свои оригинальные пиктограммы, по которым, а также по подписям, находят необходимую программу, которая запускается двойным щелчком мыши по пиктограмме. В системном меню Диспетчера Программ используются следующие кнопки: кнопка разворачивания окна на весь экран, кнопка сворачивания окна в пиктограмму, кнопка восстановления прежнего размера окна. Эти кнопки расположены справа.

Слева находится кнопка системного меню «-», которая используется для закрытия и сворачивания окна, для перемещения и для изменения его размеров. Чтобы закрыть окно и превратить его в пиктограмму, можно нажать <Ctrl+F4>.

Имеется три типа окон:

1. Окно программы (окно Диспетчера Программ — Рабочий стол), содержит выполняемую программу.

2. Окно документа — это окно программной группы, содержит значки (иконки) программных элементов.

3. Окна диалогов — в них выводятся запросы (типа Да, Нет).

Можно открыть несколько окон программных групп на Рабочем столе. Для перемещения окна надо, ухватившись мышью за его заголовок, перетащить его на новое место. Аналогично мышью перетаскивают на новое место программную группу и программный элемент.

Для изменения размеров окна надо указать мышью в его границу, курсор мыши превратится в двунаправленную стрелку, и, ухватившись мышью за край границы окна, можно менять его размеры. Можно менять сразу оба размера окна, если указать мышью в его угол. Окно можно сделать активным, если указать в его заголовок мышью. Заголовок при этом будет выделен темным цветом. Можно одновременно открыть сразу несколько окон, но активным будет только одно окно.

Для перемещения программного элемента из одного окна в другое надо его просто перетащить мышью. Если при этом нажать клавишу <Ctrl>, то можно скопировать программный элемент из одного окна в другое.

Для создания Программной группы необходимо указать в меню мышью пункт Файл, Создать, Группа Программная, ввести имя группы на русском или английском языке, OK.

Для создания Программного элемента внутри программной группы необходимо указать в меню мышью пункт Файл, Создать, Программный элемент, ввести описание или согласиться с тем, что Программному элементу будет дано имя исполнимого файла,

листать, найти на диске имя исполнимого файла (*.exe, *. сот, *.bat, *.pif), ОК.

Программы DOS получают пиктограмму MS-DOS, программы Windows имеют обычно свою собственную пиктограмму.

Пиктограммы (иконки) — это небольшие значки (картинки), предоставляющие собой свернутые объекты, такие, как:

1. Документ (пиктограмма программной группы).
2. Программа (пиктограмма программного элемента).
3. Файл (для Диспетчера Файлов).
4. Директории (для Диспетчера Файлов).

Для смены Пиктограммы надо указать в меню мышью пункт Файл Свойства, Изменить значок, Пролистать, выбрать библиотеку пиктограмм, которую содержит сам Диспетчер Программ, выбрать подходящую пиктограмму, ОК. Есть в Windows большая библиотека пиктограмм, которая содержится в файле *moricons.dll*. Из нее тоже можно выбрать подходящую пиктограмму. Пиктограммы (иконки) также содержатся в графических файлах *.ico. Их создают с помощью графического редактора Icon Editor из комплекта Norton Desk Top.

Стандартные графические файлы иконок *.ico имеют размер 766 байт.

Для смены описания (имени элемента или группы) надо их выделить, взять пункт меню Файл, Свойства, стереть с помощью <BackSpace> старое имя и ввести новое, ОК.

Для удаления Программного элемента (или Программной группы) ее сперва выделяют мышью, затем меню Файл, Удалить, ОК. Можно также использовать клавишу <Del>. Основные Программные группы и Программные элементы удалять нельзя. На их восстановление уйдет много времени. Удаление Программного элемента не означает Удаления самой программы с диска.

В пункте меню Файл есть подпункт Выполнить, где есть Командная строка. Если ввести в нее, как в командной строке DOS, команду, то можно, нажав на <Enter>, выполнить эту команду.

Подпункт Открыть в меню Файл позволяет запустить программу с выделенным программным элементом.

После инсталляции в Windows имеется несколько основных Программных групп: Главная, Группа Запуска (в ней должен быть только индикатор клавиатуры), Реквизиты, Приложения и Игры.

*Многозадачность или одновременная работа нескольких приложений* у[с]танная возможность осуществляется при помощи Списка Задач.

Вызов Списка Задач — <Ctrl+Esc> или кнопкой «-» системного Меню Диспетчера Программ, Переключиться в... можно запус-

кать по очереди разные загруженные приложения, указывая их в Списке Задач (Переключиться в... или Завершить Задачу). Сворачивание открытого приложения — <Alt+Tab>. При этом можно запустить другое приложение, реализовав режим многозадачности Повторно <Alt+Tab> — снова развернуть приложение. Количество одновременно открытых приложений ограничено объемом оперативной памяти. Однако приложения все равно работают не одновременно, а по очереди, хотя и загружены в ОЗУ. При выходе из Windows надо закрыть все активные приложения.

***Диспетчер Файлов (File Manager) в Windows 3.1*** является вспомогательной программой, открывающей особые возможности работы и выдвигающей при этом иногда и особые требования. В повседневной работе с Windows и Windows-приложениями не всегда есть необходимость в Диспетчере Файлов. Однако существует ряд задач, которые не могут быть достаточно эффективно решены без его помощи.

Среди многих полезных свойств Диспетчера Файлов следует выделить следующие два:

- с помощью Диспетчера Файлов в Windows-среде можно так организовать свою работу, что почти не придется вникать в подробности приложения (т. е. запоминать назначение его команд) и тем более беспокоиться о размещении результатов его работы, т. е. документов;

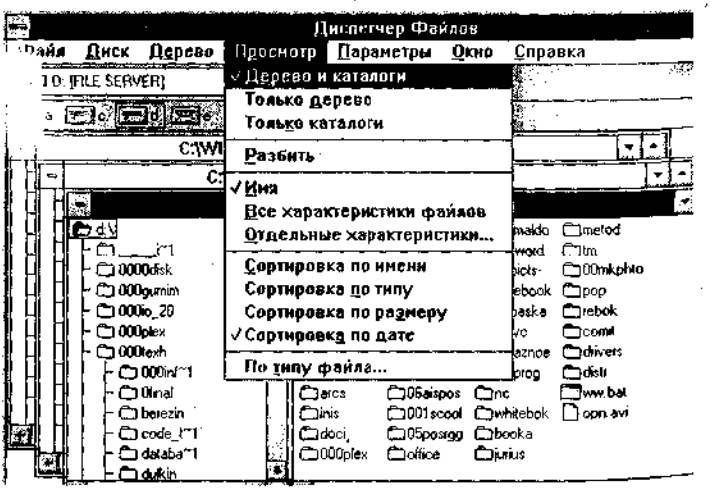



Рис. 2.7. Окно Диспетчера Файлов (File Manager)

* Диспетчер Файлов является вспомогательной программой, обеспечивающей управление файловой системой DOS.

В обеих средах (DOS и Windows) принципы хранения информации на диске одинаковы: данные организуются в файлы, имеющие уникальные имена. Файл представляет собой организованную особым образом порцию информации, которой присвоено определенное имя. В основном существует два различных типа таких файлов, которые в Windows-среде различаются, в частности, и по расширениям, а именно:

- прикладные программы (приложения), т. е. файлы программ, исполняемые файлы, с помощью которых что-либо может быть обработано;
- документы, т. е. рабочие файлы, то, что обрабатывается с помощью прикладных программ: тексты, таблицы, записи данных, графические данные и др.

Диспетчер Файлов показывает, на каких дисководах (дисках), в каких директориях и какие именно файлы доступны. При этом он также условно показывает (по мере необходимости и возможности), к какому типу эти файлы относятся. С его помощью можно управлять файлами и обрабатывать их. При этом можно работать «документоориентированно», а также и обычным способом, как принято в DOS. Это совершенно непринципиально.

Еще одно полезное своеобразие Диспетчера Файлов заключается в том, что он, так же как и Диспетчер Программ, использует оконный интерфейс подчиненными окнами. Всегда существует «основное» прикладное окно Диспетчера Файлов, внутри которого может располагаться еще несколько подчиненных окон (но не более девяти).

Эти подокна показывают структуру и содержание директорий, поэтому они еще называются окнами директорий. Можно увеличить или уменьшить различные окна, расположить их каскадом или без перекрытий, но только в пределах прикладного окна Диспетчера Файлов.

*Пиктограммафайла-документа.* Эта пиктограмма соответствует документам, точнеефайлам, для обработки которых определено некоторое приложение.Доступ к такому приложению для обработки отмеченного такой пиктограммой файла можно получить двойным щелчком на пиктограмме (илиимени файла с такой пиктограммой). Если в правой части окна ничего нет (например, по причине некорректной конфигурации Диспетчера Файлов), то следует обратиться к меню View (Вид) и установитьтам опцию Treeand Directory(Дерево и Список).

*Диспетчер Файлов как сервисная программа DOS.* DOS (дисковая операционная система MS-DOS) представляет собой комплекс программ, которые обеспечивают эффективную эксплуатацию ресурсов компьютера. DOS управляет, например, выводом информации на накопители, так и на экран дисплея или устройства печати (принтеры), и вводом информации при нажатии клавиш.

Работая в среде Windows, постоянно приходится сталкиваться с характерными для среды DOS понятиями, в особенности когда речь идет о загрузке и сохранении объектов. Запуск в меню File всякого Windows-приложения при запоминании и извлечении информации требует оперировать с такими понятиями, как имя файла, директория и накопитель. Это связано с тем, что Windows — система, которая хранит, управляет и загружает информацию так же, как реализовано в DOS.

При обычной работе прикладных программ в среде Windows Диспетчер Файлов обычно не используется. Прямое участие этой сервисной программы в процессе изготовления документов не предусмотрено. Он применяется в большинстве случаев как сервисная программа для обеспечения комфортной работы с жесткими дисками и дискетами, точнее, с информацией, запоминаемой в виде файлов.

Как уже было сказано, файлы представляют собой собранную, упорядоченную и сохраненную информацию, в роли которой могут выступать тексты, числа, картотеки, рисунки и т. п., а также программы. Диспетчер Файлов является прекрасным помощником для всех случаев управления файлами, таких, как копирование, удаление или переименование, а также подготовка дискет к использованию (их формирование). Он предоставляет возможность просмотра всех хранящихся на носителях данных (винчестерах или дискетах). С его помощью можно не только просматривать, какие файлы находятся в директориях, но также и создавать такие директории, редактировать и удалять. Кроме всего этого, можно с помощью Диспетчера Файлов также и запускать программы.

*Приложения Windows 3.1.* В фирменной поставке пакета Windows находится несколько приложений. Все они объединены в группу Accessories (реквизиты, аксессуары, инструменты). Это не совсем большие по размеру и возможностям прикладные программы, составляющие «джентльменский набор» пользователя. Им далеко до профессиональных специализированных пакетов, но они прекрасно иллюстрируют возможности оболочки и обеспечивают некоторый минимальный сервис. Более того, весьма полезно начинать знакомство с серьезными пакетами именно с соответствующих средств этой группы. Так, например, поработав некоторое время с текстовым ре-


Рис. 2.8 Стандартные приложения (Реквизиты, Accesories) Windows 3.1

дактором Write, в дальнейшем можно легко перейти к использованию таких профессиональных пакетов обработки текстов, как Word для Windows, Lotus Ami Professional, WordPerfect для Windows и т. п.

Основные приложения Windows, входящие в его состав, расположены в программной группе Реквизиты (т. е. их программные элементы):

1. Графический редактор PaintBrush.
2. Текстовый редактор Write.
3. Простейший текстовый редактор txt-файлов Блокнот (Note-Pad).
4. Калькулятор (в стандартном виде и научный с элементарными Функциями).
5. Часы (с цифровой или стрелочной индикацией).
6. Календарь (с будильником).
7. Картотека (простейшая база данных).
8. Таблица символов.
9. Раскладка клавиатуры.
10. Звукозапись (с микрофона в мультимедийных машинах).
11. Медиаплеер (для воспроизведения видео- и аудиозаписей в мультимедийных машинах).
12. Упаковщик объектов.

***Проблемы использования DOS-приложений в Windows.*** В связи с широким распространением оболочки Windows все более обостряется немаловажный с практической точки зрения вопрос: как работают в оболочке приложения, изначально не предназначенные для эксплуатации в среде Windows?

В настоящее время эксплуатации находится много таких «не-Windows» или DOS-приложений которые на практике доказали свою полезность и пользуются популярностью у пользователей. С точки зрения таких

программ оболочка Windows вообще не нужна. Они захватывают все ресурсы компьютера в свою собственность и не нуждаются в выделении им окон, пиктограмм, памяти и т. п.

Работа под управлением Windows имеет свою специфику. Оболочку Windows 3.0 или 3.1 можно рассматривать как некоторое DOS-приложение, способное управлять другими DOS-приложениями. Под управлением здесь, как минимум, понимается возможность запуска других приложений, переключения между параллельно работающими программами и обмен данными через буфер промежуточного хранения. Степень такой управляемости и глубина интеграции зависят не в последнюю очередь от режима работы самой оболочки.

Под DOS-приложениями в данном случае мы будем понимать программы и комплексы программ, не предназначенные изначально для работы под управлением оболочки Windows. Наша задача — попытаться заставить их работать под управлением оболочки. К подобным DOS-приложениям могут быть отнесены практически все эксплуатируемые в MS-DOS прикладные программы. Основные особенности таких программ:

- DOS-приложения ориентированы на монопольное владение ресурсами компьютера и автономное управление ими;
- возможность переключения на параллельно работающую программу, так же как и использование общего буфера промежуточного хранения, в них не предусмотрена;
- они работают, как правило, не в окне, а захватывают для своих нужд целый экран;
- они не предусматривают возможности разделения ресурсов памяти, вводных и выводных устройств (за исключением разве что принтера) с другими программами;
- они имеют собственные средства управления и оригинальные форматы данных (по этой причине большинство DOS-приложений даже после интеграции их в оболочку не способны обмениваться данными с другими приложениями через Clipboard).

Несмотря на претензии DOS-приложений в отношении монопольного владения всеми ресурсами компьютера, они все-таки могут быть «ограничены рамками законности» и интегрированы в среду Windows. Пользователи, решившие связать свою судьбу с Windows, вовсе не обязаны распрощаться с привычным набором DOS-приложений. Если в рамках оболочки отсутствуют средства подобного класса, то «старые привязанности» вполне можно сохранить.

Следует обратить внимание на то, что ряд специальных DOS-приложений конфликтует с оболочкой Windows и не работает (или работает не всегда корректно) под ее управлением. Это касается

первую очередь сервисных программ, работающих напрямую с винчестер (например, для дефрагментации его, как Norton Speed D выполняющих сжатие данных (динамическое).

К нфликтовать с оболочкой могут и резидентные программы — драй и т. п. Следует также отметить, что сложности, встречавш при работе под управлением оболочки Windows 3.0 таких пакетов, как Lotus 1-2-3 (версия 3.0), Lotus Freelance и AutoCAD, в о лочке Windows 3.1, практически преодолены.

*Интеграция DOS-приложений.* Использование DOS-приложения в среде Windows не означает простого функционирования этого приложения под управлением оболочки. Оболочка обеспечивает ряд дополнительных возможностей.

У DOS-приложения появляется свой «идентификационный символ» — пиктограмма. Посредством программной пиктограммы приложение может быть запущено (двойным щелчком), а затем свернуто до прикладной пиктограммы, из которой его можно опять развернуть.

Возможно параллельное выполнение нескольких DOS-приложений (но только в расширенном режиме). Но даже в стандартном режиме работы оболочки возможен некоторый псевдопараллелизм в работе DOS-приложений посредством реализации простого механизма межзадачного переключения.

Возможен (хоть и с определенными ограничениями) обмен данными с другими приложениями через буфер промежуточного хранения. Например, имеется (иногда) возможность импортировать в DOS-приложение текст из буфера или (чаще) экспортировать в буфер содержимое экрана DOS-приложения.

В расширенном режиме DOS-приложения могут выполняться в окнах (включая и показ высококачественной графики — новинки версии 3.1). Они при этом выполняются параллельно с другими программами, как обычные Windows-приложения. При достаточно высокой производительности процессора можно выполнять до 15 DOS-приложений одновременно под управлением Windows (не считая Windows-приложений).

При интеграции DOS-приложения в среду Windows система не в кает в подробности функционирования интегрируемого продукта в место этого оболочка создает для приложения такую среду, тор полностью отвечает его потребностям. Попав в такие условия, DOS Уверен риложение бесконфликтно существует, пребывая в твердой Ро ости, что весь компьютер находится в его распоряжении.

риложения, не предназначенные изначально для работы в среде Windows, автоматически распознаются данной оболочкой как

«не-Windows» приложения и интегрируются в оболочку некоторым стандартизованным способом. Пользователь, однако, имеет возможность вмешиваться в процесс интеграции и вносить в него свои изменения.

Способ интеграции DOS-приложений в оболочку, примененный в Windows 3.1, доказал свою надежность. Но «беспроблемная» интеграция удается, к сожалению, не во всех случаях. В большинстве случаев зависания системы (ставших с появлением версии 3.1 гораздо более редкими) виновными являются, как правило, DOS-приложения.

*Простой запуск.* При необходимости запуска некоторого DOS-приложения из среды Windows (однократно) вовсе не обязательно интегрировать его для этого в оболочку. Для такого запуска предусмотрены следующие возможности:

- из Диспетчера Файлов;
- посредством директивы Run меню File Диспетчера Программ;
- с помощью утилиты MS-DOS Prompt (Подсказка MS-DOS), принадлежащей группе Main.

Во всех трех случаях требуется знать точное имя файла программы и директорию, в которой она находится.

В Диспетчере Файлов запуск производится двойным щелчком на пиктограмме файла или на его имени.

При запуске приложения с помощью директивы Кип меню File следует ввести полную спецификацию файла программы (путь и имя) в поле ввода.

***Помощь (справочная подсистема).*** Оболочка Windows 3.1 располагает хорошо структурированной справочной подсистемой с большим объемом справочных текстов. Иногда эту подсистему называют подсистемой помощи (Help) по имени директивы, открывающей доступ к ее ресурсам. Справочная подсистема содержит информацию по всем компонентам системы, по каждому интегрированному в нее Windows-приложению. Кроме того, доступна контекстная справочная информация, т. е. информация о текущей ситуации в системе — ее можно вызвать в любой момент работы с оболочкой.

Интеграция дополнительного приложения в систему автоматически добавляет к справочной подсистеме справку о нем. Конечно, объем, способ организации и качество этой справки остаются на совести и являются предметом забот (а зачастую и делом чести) фирмы-поставщика данного приложения. Ниже описывается только та часть справочной подсистемы, которая является частью фирменной поставки оболочки Windows 3.1.

Принцип построения справочной подсистемы позволяет соблюдать единообразие в действиях при получении справки в различных ситуациях. Сама справочная подсистема содержит специальную справку о методах работы с ней. Доступ к этой справке можно получить, если в операционном меню Help выбрать директиву How to Use Help (Как пользоваться справкой) или, находясь в справочном окне, нажать функциональную клавишу <F1>.

Главный принцип, заложенный в основу справочной подсистемы, — высокая структурированность текста и поддержка системы перекрестных ссылок. Отдельные фрагменты текста, входящего в справочную подсистему, упорядочены по контекстным связям.

На винчестере находится ряд справочных файлов (по одному на приложение). Например, там имеется файл справки по Диспетчеру Программ.

Вызвать справку можно различными способами.

Наиболее удобным является так называемый контекстный вызов. Под этим термином кроется весьма удобное свойство справочной подсистемы — ее способность в момент вызова определять, какая именно справка нужна в данной ситуации. Определив, подсистема находит на диске требуемый файл, загружает его и выдает на экран подобранную справку.

Прикладные окна имеют в строке меню обязательный элемент Help для обращения к справочной подсистеме. При фиксации курсора на этом слове вызывается справка по данному приложению. Тем же подходом можно воспользоваться и в диалоговых окнах, помня, однако, о том, что там для вызова справки имеется специальное поле кнопки Help.

Доступ к любой справке можно получить и посредством принудительной загрузки соответствующего справочного файла «вручную» (т. е. посредством директивы Open меню File любого справочного окна).

После загрузки справки в справочное окно дальнейшая работа со справочной подсистемой выглядит во всех трех случаях одинаково.

*Ситуационная справка.* Ситуационная или контекстная справка вызывается с помощью клавиатуры. Обычно для этих целей используется функциональная клавиша <F1>. В некоторых случаях дальнейшую (более детальную) информацию можно получить, нажав <Shift+F1>. Справка, вызываемая в диалоговом окне щелчком на кнопке Help, также является контекстной.

Попробуйте нажать на функциональную клавишу <F1> в момент, когда активен Диспетчер Программ. На экране появляется окно с заголовком Groups and Applications Help (Справка по Диспетчеру

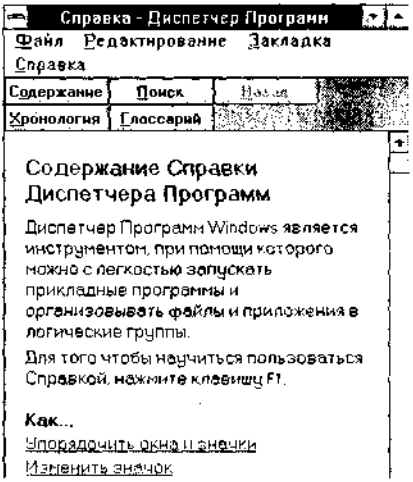



Рис. 2.9. Пример ситуационной справки

Программ). В данный момент в нем выведено Содержание справки по Диспетчеру Программ (Contents for Program Manager Help).

*Редактор PIF-файлов.* Программно-информационный файл (PIF-файл) предназначен для запуска программ в Windows. Для создания и редактирования PIF-файлов служит редактор PIF-файлов PIF-Editor. Его пиктограмма находится в программной группе Главная. Для создания PIF-файла необходимо сделать следующие операции:

1. Запустить редактор PIF-Editor.

2. В окне Имя файла программы надо указать имя файла с расширением exe, cot или bat файл той программы, которую надо запускать в Windows. Причем этот файл надо указать с полным путем, например: `c:\dos\msd.exe`.

3. В окне Заголовок окна можно указать текст, который будет являться подписью под пиктограммой запускаемой программы. Например: MS Diagn.

4. В пункте меню редактора Файл выбрать подпункт «Сохранить как», затем выбрать каталог для размещения PIF-файла и дать ему имя, например: msd.pif. В итоге надо нажать кнопку OK и сохранить PIF-файл. Чаще всего PIF-файлы сохраняют в каталоге Windows, но необязательно.

5. С помощью вновь созданного PIF-файла можно запускать указанную в нем программу при помощи Диспетчера Программ или Диспетчера Файлов Windows.

Все файлы *.pif имеют одинаковый размер, равный 545 байт.

*Запуск и настройка Windows 3.1.* Для запуска Windows можно набрать командную строку.

`win` - чаще всего;
`win /s` — стандартный режим;
`win /3` — расширенный режим;
`win:_` пропустить заставку.

*Выход из Windows.* Если окно Диспетчера Программ видно, то сперва щелкнуть мышью в слово Файл и затем в слово Выход из Windows, далее подтвердить выход (Ок) или отменить выход (Отмена) Если окно Диспетчера Программ не видно, то можно для выхода из Windows нажать клавиши <Alt+F4> и подтвердить выход из Windows (OK) или отменить выход (Отмена).

*Системная информация в Windows* (программа Setup установки Windows):

Компьютер: Система MS-DOS
Дисплей: VGA
Мышь: Мышь Mouse System
Клавиатура: Клавиатура США и не-США (101/102 клавиши)
Раскладка клавиатуры: Американская
Доп. раскладка клав.: Русская
Язык: Кириллица
Кодовая страница: США (437)
Сеть: Сеть не установлена

*Панель управления* служит для настройки режимов работы Windows. Панель управления позволяет также выбрать необходимый драйвер Принтера, установить параметры Оформления и Цвета экрана, установить параметры Расширенного режима, выбрать параметры работы Клавиатуры и Мыши, установить текущие Дату и Время, выбрать Шрифты, Порты и режимы мультимедийного оборудования.

Панель управления Windows находится в Программной группе Главная.

*Стандарты (Панель управления):*
Страна: Россия
Язык: Кириллица
Раскладка клавиатуры: Американская
Дополнительная раскладка клавиатуры: Русская
Тема
Система мер: Метрическая

Панели управления, в этих параметрах, которые устанавливаются с помощью Панели управления, в Windows и WinWord должна переключаться клавиатура (рус./лат.) нажатием двух <Shift+Shift> или <Alt+Tab>.

*Windows•for Workgroups 3.11.* Следующая версия Microsoft Windows 3.11 была названа Windows for Workgroups 3.11. Основным отличием ее от версии Windows 3.1 является то, что в программный пакет интегрированы сетевые драйверы, позволяющие использовать его не только на отдельно стоящем ПК, но и в сети. Кроме того, в операционную среду включены несколько новых программ, значительно изменен Диспетчер Файлов — одно из слабых мест предыдущих версий, расширен Диспетчер Печати и встроен факс. Разработан новый офисный пакет, включающий в себя: текстовый редактор Word, электронные таблицы Excel, редактор формул Equation и ряд других возможностей. Внешний вид интерфейса Windowsfor Workgroups 3.11 представлен на рис. 2.10 и практически не отличается от Windows 3.1.

*Сетевые возможности.* Само название системы «Windows для рабочих групп» означает то, что программа предназначена для работы на нескольких ПК, причем равноправных пользователей, объединенных в рабочие группы.

В Windows for Workgroups дополнительно введен ряд сетевых функций. Наиболее заметные из них — средства организации одноранговых сетей, которые позволяют вам пользоваться диском или каталогом совместно с другими пользователями либо с помощью

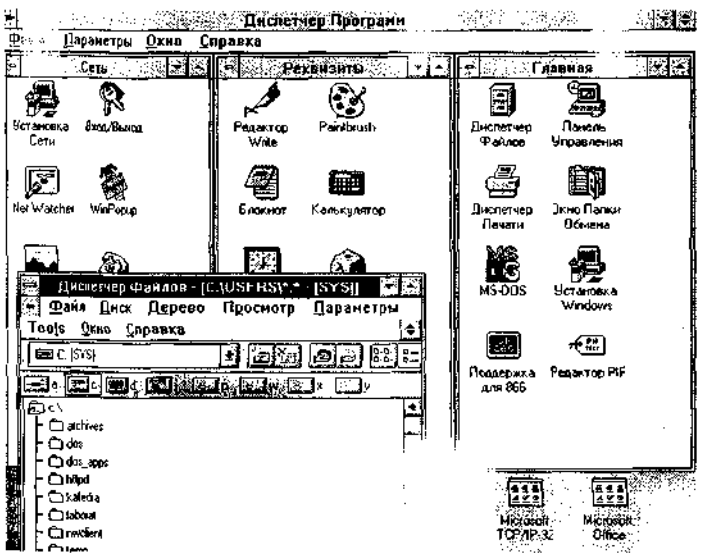



Рис. 2.10. Операционная система Win... for Workgroups 3.11

встроенных сетевых функций Windows, либо через более крупную сеть на основе сервера,такую, как NetWareфирмы Novellили Windows NT.

Кроме того, в Windows for Workgroups добавлен клиент электронной почты Microsoft Mail и планировщик Schedule+. Для пользователей не подключенных к сети, большое преимущество Windows for Workgroups состояло в применении 32-разрядного виртуального драйвера устройств (VxD) для файловой системы. Он может значительно повысить производительность многих накопителей с интерфейсом IDE.

Сетевая версия Windows for Workgroups 3.11 позволяет легко связываться с другими участниками сети и пользоваться информацией общего доступа, а также легко изменять конфигурацию сети, приспосабливая ее для нужд рабочей группы. Работая с сетевой версией Windows For Workgroups 3.11, можно обмениваться информацией по сети, где используются совместимые сетевые программы, такие, как Novell NetWare. Можно работать и на отдельно стоящем компьютере, а позже при необходимости легко создать сеть. Создаваемая с помощью пакета Windows For Workgroups 3.11 сеть обладает всеми необходимыми качествами для обеспечения успешной работы с основными прикладными программами MS-DOS.

*Процесс инсталляции и рекомендации:* Инсталляция системы происходит частично в текстовом режиме и требует установки некоторых начальных параметров, а далее инсталляция продолжается в графическом режиме и при инсталляции на процессоре 386DX-40 МГц занимает около 30 мин.

Эта операционная система, в связи с ее расширенными возможностями, может рекомендоваться для организации небольшой сети в офисах и учреждениях.

*Краткие сведения об архитектуре Windows 3.x.* В соответствии с архитектурой Windows все прикладные программы и системный код размещаются в едином адресном пространстве. Это означает, что недоработанная прикладная программа, содержащая ошибки, может испортить области памяти, которые используются операционной средой или другой прикладной программой. Результат будет весьма неприятная ошибка общего нарушения защиты (General Protection Fault). Иногда Windows с честью выходит из положения, восстанавливая свою работоспособность, но чаще всего это ей не удается.

В своей основе Windows 3.x — 16-разрядная операционная система, поэтому для программ память представляется состоящей из 64-Кбайт сегментов, а все данные в свой основе 16-разрядные. Та-

кая система может оказаться менее эффективной по сравнению с 32-разрядной адресацией при работе с большими массивами данных. Еще одно следствие 16-разрядной базы этой ОС — ограниченность системных ресурсов. В Windows 3.x для хранения таких структур, как дескрипторы файлов прикладных программ, выделяется лишь небольшой блок памяти в других адресах. После того как эти области памяти заполнятся, Windows не может загрузить новые прикладные программы, даже если в ее распоряжении остается вполне достаточно памяти в других адресах.

В основе организации Windows 3.x лежит 16-разрядная архитектура. Ее ядро, большинство важнейших компонентов и собственные

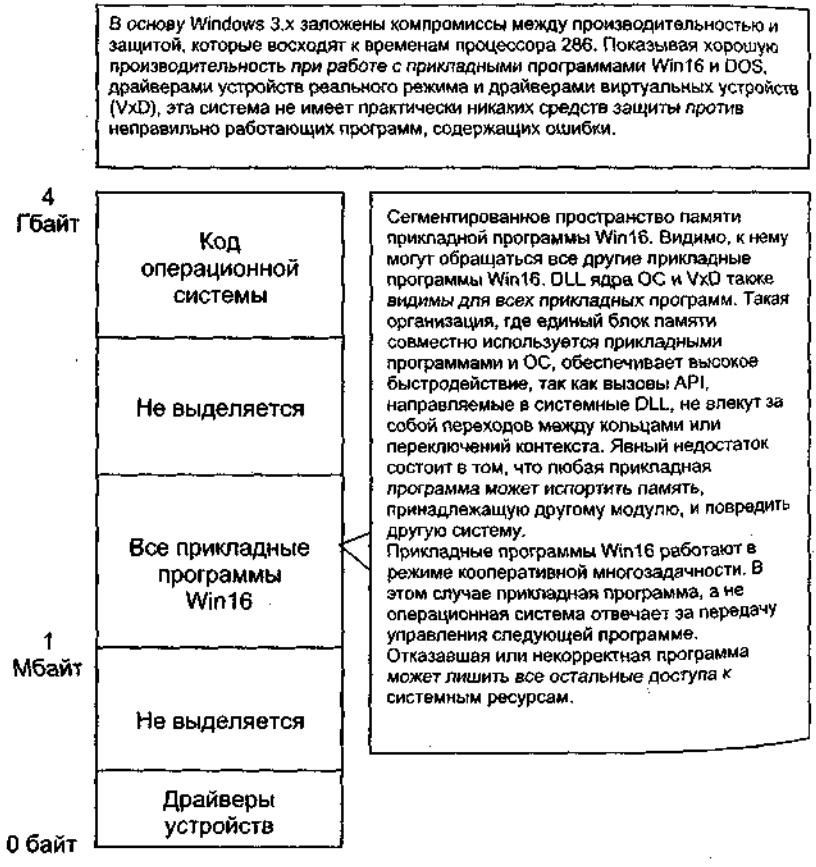



Рис. 2.11. Модель памяти Windows 3.x

прикладные программы представляют собой 16-разрядные коды. (Ее основной используемый интерфейс Win32 API дает возможность выполнять 32-разрядные прикладные программы, но не позволяет работать с несколькими потоками.)

Все собственные прикладные программы Windows 3.x и все ее системные библиотеки DLL отображаются в общее сегментированное виртуальное адресное пространство размером 4 Гбайт. Все эти компоненты видимы (и часто доступны на уровне записи) друг для друга. В нижней части этого адресного пространства, обычно ниже метки 1 Мбайт, размещаются драйверы устройств реального режима, обеспечивающие взаимодействие с периферийными подсистемами, такими, как видеоплаты или принтеры. В Windows 3.11 драйверы VxD файловой системы используются для отыскания маршрута доступа к диску в защищенном режиме.

Упрощенная организация системы позволяет получить очень малое рабочее множество (working set — прикладной и системный код, который необходимо загрузить в память для любой данной задачи), поэтому Windows 3.1x может успешно выполняться на компьютерах с ОЗУ ограниченного размера. Такая архитектура также способствует повышению эффективности исполнения кода, так как программы могут вызывать функции API из собственного пространства памяти. Недостаток архитектуры состоит в слабой защите от сбоев при неправильной работе программ. Программы и системные компоненты видимы друг для друга, модуль, содержащий ошибки, может легко испортить содержимое памяти, принадлежащей другому процессу. Хотя Windows 3.1x способна восстанавливать свою работоспособность после некоторых нарушений защиты общего характера (General Protection Fault), зачастую результатом становится крах всей системы.

Windows 3.1x одновременно выполняет несколько прикладных программ с помощью простого механизма планирования, называемого кооперативной многозадачностью. В этой системе каждая прикладная программа должна добровольно уступить управление, когда, проверив свою очередь сообщений, она обнаруживает, что та пуста. Но если прикладная программа не проверит свою очередь сообщений либо по причине занятости, либо вследствие зависания, то все прикладные программы лишатся доступа к совместно используемым ресурсам.

Другой недостаток — ограниченность ресурсов модулей GDI и USER. Эти ограничения возникают в связи с тем, что системные библиотеки GDI и USER используют несколько 64-Кбайт динамических областей (хипов) для хранения разнообразных скрытых

структур данных, создаваемых выполняющимися в данный момент прикладными программами. Когда эти небольшие хипы переполняются, вы получаете сообщение о нехватке памяти, даже если в системе остается много свободной памяти.

## 2.3. Операционные системы Windows 95/98/ME

Первые версии Windows 3.x нельзя назвать полноценными операционными системами, так как для их работы обязательно было наличие активной копии MS-DOS. Таким образом, Windows являлась как бы посредником пользователя и операционной системы, облегчая процесс общения между ними.

В качестве развития серии Windows были выпущены две параллельные ветви операционных систем с графической оболочкой:

- Windows 95/98/ME;
- Windows NT/2000.

Системы, внешне весьма схожие по интерфейсам и предоставляемым пользователям возможностям, коренным образом различаются по назначению и администрированию. Основные отличия заключаются в том, что предназначенная для использования в быту и малых офисах серия 95/98 базируется на принципе Plug&Play подключения новых устройств (автоматический поиск и установка драйвера вновь появившегося устройства) и не требует серьезного администрирования, в то время как серия NT ориентирована на сетевое использование в больших организациях и требует точного конфигурирования и постоянного администрирования.

***Объектно-ориентированный подход.*** При создании Windows 95 в полной мере реализован объектно-ориентированный подход. Понятие «объектно-ориентированный» возникло в программировании сравнительно недавно. Когда вычислительная мощность машин была невысока, о создании объектно-ориентированных систем не могло быть и речи. Основой всего был программный код. Программисты записывали последовательности команд для выполнения тех или иных действий над данными, которые оформлялись в модули и процедуры. Для работы с каждым объектом создавалась своя процедура.

*Объекты, их свойства и методы.* Постепенно с увеличением производительности вычислительных систем процедурный подход начал заменяться объектным. На первое место выдвинулся объект, а не код, который его обрабатывает. На уровне пользователя объект-

ный подход выражается в том, что интерфейс представляет собой подобие реального мира, а работа с машиной сводится к действиям с привычными объектами. Так, папки можно открыть, убрать в портфель, документы — просмотреть, исправить, переложить с одного места на другое, выбросить в корзину, факс или письмо — отправить адресату и т. д. Понятие объекта оказалось настолько широким, что до сих пор не получило строгого определения.

*Объект,* как и в реальном мире, обладает различными свойствами. Программист или пользователь может изменять не все свойства объектов, а только некоторые из них. Можно изменить имя объекта, но нельзя изменить объем свободного места на диске, который также является его свойством. Свойства первого типа в языках программирования носят название read/write (для чтения и записи), а свойства второго — read only (только для чтения).

*Метод* — это способ воздействия на объект. Методы позволяют создавать и удалять объекты, а также изменять их свойства. Например, для того чтобы нарисовать на экране точку, линию или плоскую фигуру, составляются разные последовательности кодов или программы. Пользователь, однако, применяет для отображения этих объектов один метод Draw (), который содержит коды для отображения всех объектов, с которыми он работает. За такое удобство приходится платить тем, что объектно-ориентированные системы могут работать только на достаточно мощных вычислительных установках.

***Windows 95, основные особенности.*** С точки зрения базовой архитектуры Windows 95 — 32-разрядная, многопотоковая операционная система с вытесняющей многозадачностью, что ставит ее в один ряд с такими соперниками, как OS/2, UNIX и Windows NT. В ее среде могут выполняться собственные 32-разрядные прикладные программы, написанные в соответствии со спецификацией Win32 API (почти идентичный вариант этого интерфейса реализован в Windows NT). Собственные прикладные программы Windows 95 используют неструктурированное 32-разрядное адресное пространство, что делает их потенциально более быстродействующими при обработке больших массивов данных [11, 13].

*Компоненты ядра Windows 95.* Ядро Windows 95 состоит из трех компонент:

- *User* управляет вводом с клавиатуры, от мыши и других координатных устройств, а также выводом через интерфейс пользователя. В Windows 95 используется модель асинхронного ввода;

- *Kernel* обеспечивает базовые функциональные возможности операционной системы (поддержку файлового ввода/вывода, управление виртуальной памятью, планирование задач), загружает exe- и dll-файлы при запуске программы, обрабатывает исключения, обеспечивает взаимодействие 16-разрядного и 32-разрядного кодов;
- *GDI* — графическая система, управляющая всем, что появляется на экране дисплея, и поддерживающая графический вывод на принтер и другие устройства.

В Windows 95 каждая 32-разрядная прикладная программа выполняется в собственном адресном пространстве, но все они совместно используют один и тот же 32-разрядный системный код. Неправильно написанная 32-разрядная программа все еще может привести к аварийному сбою всей системы. Все 16-разрядные программы Windows разделяют общее адресное пространство, поэтому они столь уязвимы друг для друга, как и в среде Windows 3.1. В практической работе Windows 95 производит впечатление более устойчивой среды, чем предшествующие версии Windows. Очевидны крупные изменения в пользовательском интерфейсе Windows 95. Вы используете кнопку Start для запуска прикладных программ самих по себе или через документы, с которыми программы связаны. После запуска программ их пиктограммы появляются на линейке заданий, обычно размещаемой в нижней части экрана. Щелчок на любой кнопке линейки заданий вызывает переключение на соответствующую программу. Это интуитивно самый понятный способ переключения задач из всех когда-либо существовавших.

Модули Рrодгат Мападег (Диспетчер Программ) и РПе Manager (Диспетчер Файлов) уступили место *образу рабочего стола,* на котором файлы пользователя показаны в виде *пиктограмм,* помещенных в так называемые *папки.* Более сложные функции по управлению файлами Windows 95 поручены утилите Проводник (Explorer), по существу заменившей РПе Мападег, которая показывает древовидную диаграмму файловой структуры компьютера и его сетевого окружения. Благодаря расширению файловой системы РАТ имена файлов не ограничены, как раньше, восемью символами плюс состоящим из трех букв расширением, а можно использовать имена длиной до 255 символов.

Среди прочих благоприятных изменений в пользовательском интерфейсе — анимационные пиктограммы и диалоговые окна с закладками. В целом новый интерфейс представляет собой существенное улучшение по сравнению с Windows 3.1, хотя прежним пользователям потребуется некоторое время, чтобы привыкнуть к нему.

Конечно, метафора «рабочего стола», основанная на применении папок и длинные имена файлов не изобретены создателями Windows 95, а в течение длительного времени они были составной частью различных пользовательских интерфейсов, начиная с Macintosh и кончая Workplace Shell операционной системы OS/2.

Увеличилось число и повысилось качество поставляемых вместе с Windows 95 стандартных вспомогательных программ — от традиционного калькулятора и игр до мощных инструментальных средств контроля состояния системы. Также очевидны значительные усовершенствования средств связи.

*Важнейшими приложениями* Windows 95 являются:

1. Блокнот (NotePad) — простейший текстовый редактор (notepad.exe).
2. Калькулятор (вид: обычный и научный, calc.exe).
3. Paint — графический редактор, аналог PaintBrush (pbrush.exe).
4. WordPad — текстовый редактор (улучшенный аналог Write, write.exe).
5. Часы (clock.exe).
6. Медиаплеер (универсальный проигрыватель, mplayer.exe) - служит для проигрывания видео- и аудиоклипов на мультимедийных компьютерах.
7. Проводник (Explorer, explorer.exe) — играет роль оболочки или менеджера файлов.
8. Буфер обмена (ClipBoard, clipbrd.exe).
9. ScanDisk для Windows (scandskw.exe) — основной инструмент исправления ошибок в файловой системе Windows95 при сбоях.
10. Defrag (defrag.exe) — используется для дефрагментации жесткого диска.
11. Лазерный проигрыватель (cdplayer.exe) — используется для проигрывания аудио CD.

Есть еще ряд других приложений Windows 95 (набиратель телефонного номера календарь, фонограф для записи и воспроизведения звука, эмуляция терминала, игры и др.).

Расширились сетевые функциональные возможности. В состав Windows 95 включен встроенный клиент для сетей NetWare 3.x, 4.x и для серверов Windows NT. Предусмотрены также средства для работы с протоколами IPX/SPX, NetBEUI, TCP/IP. Последний из перечисленных протоколов позволяет выполнять подключение к Internet, хотя лучшая программа для соединения с Internet, содержащая утилиту просмотра Web, входит в состав пакета Microsoft Plus. Windows95 позволяет непосредственно подсоединяться к другому компьютеру через кабель и располагает базовыми средствами для

установления коммутируемых соединений через телефонные линии с сервером удаленного доступа Remote Accesz Server системы Windows NT, NetWare Connect или с коммутируемыми серверами компании Shiva. В состав Windows95 также входит интерфейс прикладного программирования для телефонии (TAPI) фирмы Microsoft, обеспечивающий совместную работу вашей машины с телефоном, регистрируя телефонные вызовы и выполняя функции автоответчика (прикладные программы для телефонии будут поставляться независимыми фирмами).

Резюмируем основные достоинства Windows 95:

1) практически полная 32-битная операционная система, что ускоряет работу многих программ по сравнению с 16-битной операционной системой MS-DOS 6.22 и операционной оболочкой Windows 3.1;

2) удобный графический многооконный интерфейс для пользователя (Рабочий стол — Desktop);

3) возможность создания на Рабочем столе Ярлыков и Папок важнейших программ для их быстрого запуска. Внутри Папок могут находиться другие Папки и Ярлыки, что создает удобство в работе;

4) запуск прикладных программ и возможность создания Меню при нажатии кнопки Пуск (Start) в Панели задач;

5) самонастраивающаяся система драйверов поддержки аппаратной части компьютера (технология «Plug and Play» — «Подключай и Работай»);

6) настоящая многозадачность (по сравнению с Windows 3.1);

7) развитые сетевые функции, включая Internet;

8) большой выбор прикладного программного обеспечения и поддержка большинства 16-битных приложений;

9) хорошая поддержка Multimedia;

10) ориентация большинства современных производителей компьютерной техники и программного обеспечения на Windows 95.

Недостатки ОС Windows 95:

1) высокие требования к аппаратной части компьютера (процессору, ОЗУ, жесткому диску);

2) недостаточная устойчивость в работе, особенно у русскоязычных версий. При сбоях в работе Windows 95 автоматически запускается программа Scandisk, которая, как правило, устраняет сбои и восстанавливает нормальную работу Windows 95;

3) при выключении или перезагрузке компьютера с ОС Windows 95 надо обязательно использовать кнопку Пуск, Завершение работы, Выключить (Перезагрузить) компьютер или клавиши <Alt+F4>, <Enter>. Это по сути дела есть парковка компьютера в

W ... 95 При этом сохраняется текущая конфигурация и очища-
ются ...утренние буфера. Только после этого пользователь имеет
право ...ключить компьютер (на экране появится соответствующее
сообщ... ). В противном случае возможны сбои и потеря важной
...формации!

41 для обслуживания диска на компьютере с ОС Windows 95
надо использовать только собственные средства Windows 95 (Scan-
Disk и Detrag для Windows 95) или специальные утилиты NU-9 для
W dows-95 и ни в коем случае нельзя использовать старые утилиты
(NU-6 7,8) SpeeDisk, Norton Disk Doctor, особенно для русскоязыч-
ной версии Windows 95. Иначе можно повредить файловую систему
Windows 95.

*Основные отличия Windows 98.* Новые средства Windows 98 облегчают работу с компьютером и расширяют возможности его использования.

*Поддержка нескольких мониторов.* Поддержка нескольких мониторов делает возможной использование нескольких мониторов для расширения рабочего стола, выполнения разных программ на разных мониторах, а также выполнение многоэкранных программ или игр. Например, студенты получают возможность открыть энциклопедию Microsoft Encarta на одном мониторе и печатать собственный отчет в Microsoft Word на втором мониторе.

*Управление питанием.* Выбор режима Всегда Вкл сокращает время запуска компьютера. При использовании средств управления питанием в режиме Всегда Вкл для запуска компьютера достаточно нескольких секунд. При этом все программы восстанавливаются в том состоянии, которое они имели на момент отключения. Кроме того, этот режим позволяет компьютеру работать даже тогда, когда он кажется выключенным. Пользователь получает возможность оставить все программы выполняющимися, загружать нужные Web-страницы, отправлять и получать электронную почту, архивировать жесткий диск или выполнять настройку операционной системы без необходимости находиться у компьютера.

...еобходимо иметь компьютер со средствами автоматического у...ления питанием, которые особенно хорошо работают на новых ...омпьютерах с интерфейсом автоматического управления конфигурацией и питанием (ACPI). Кроме того, средства управления питанием позволяютперевести компьютер в режим ожидания (спящий р...жим) для сохранения ресурсов питания.

*Ша* Универсальная последовательная шина (USB)облегчает использование компьютера за счетрасширенных возможностей самонастраивающихся (plug-and-play) устройств. Новый универ-

сальный стандартный разъем позволяет добавлять устройства без необходимости перезагружать компьютер.

*Повышение надежности.* В Windows 98 надежность компьютера повышается за счет применения новых мастеров, служебных программ и ресурсов, обеспечивающих бесперебойную работу системы.

*Проверка системных файлов.* Проверка системных файлов позволяет отслеживать наиболее важные файлы, обеспечивающие работу компьютера. Если эти файлы повреждены или перемещены, программа проверки системных файлов их восстанавливает.

*Проверка реестра.* Проверка реестра является системной программой, позволяющей обнаруживать и устранять ошибки в реестре. При каждом запуске компьютера программа проверки реестра автоматически проверяет реестр на наличие несогласованности структуры данных.

Кроме того, программа проверки реестра каждый день выполняет резервирование реестра. Если обнаруживаются серьезные ошибки в реестре, реестр можно восстановить по резервной копии. Программа проверки реестра поддерживает до пяти сжатых архивных копий реестра, при которых компьютер успешно запускался. Если архив не удается обнаружить, программа проверки реестра исправляет ошибки реестра.

Программа установки автоматически запускает проверку реестра при каждом обновлении операционной системы компьютера. При установке Windows 98 программа проверки реестра исправляет большинство ошибок в реестре, даже те, о которых было неизвестно пользователю.

*Архивация данных.* Программа архивации предоставляет расширенные возможности архивации и восстановления данных, в том числе поддержку большего числа накопителей на магнитной ленте и самого современного оборудования. Пользователям становится легче сохранять важные данные. Файлы с жесткого диска можно резервировать на гибких дисках, магнитной ленте или другом компьютере в сети. Если исходные файлы повреждены или потеряны, их можно восстановить из архива.

*Быстрая операционная система.* Windows 98 включает средства, позволяющие компьютеру работать быстрее по сравнению с Windows 95 без добавления нового оборудования. В состав Windows 98 входит ряд программ, совместное применение которых повышает производительность компьютера.

*Мастера Windows.* Разработанные Microsoft Мастера стали популярным средством для проведения пользователя через последовательные этапы сложных процедур. С помощью серии доста[illegible]чно

простые вопросов пакету удается досконально выяснить, чего конкретно хочет пользователь, и выполнить соответствующую операцию.

*Мастер обслуживания.* Мастер обслуживания помогает повысить производительность системы. Мастер обслуживания позволяет быстрее выполнять программы, проверять жесткий диск на наличие ошибок и освобождать место на диске. Создав расписание для регулярного выполнения этих служебных программ, вы сможете добиться максимальной производительности компьютера. Например, оставляя компьютер включенным на ночь, можно составить расписание для выполнения этих задач в указанное время каждую ночь, раз в неделю или с любым другим интервалом.

*Мастер подключения к Интернету.* Новый мастер подключения к Интернету поможет вам зарегистрироваться для доступа к Интернету и автоматически выполняет шаги по настройке программного обеспечения, необходимые для доступа к Интернету.

*Проверка диска.* Проверка диска запускается автоматически после неверного выключения операционной системы. Программа проверки диска обнаруживает наиболее вероятные повреждения файлов и папок и выполняет исправление ошибок. Кроме того, пользователь имеет возможность выполнить проверку диска в любое время.

*Преобразование диска.* Преобразование диска в систему FAT32, которая является расширенной версией системы FAT (File Allocation Table), позволяет форматировать как один диек большие диски с емкостью более 2 Гигабайт. Преобразованные диски используют кластеры меньших размеров, чем на дисках FAT, в результате чего повышается эффективность использования объема диска. В состав Windows 98 включена служебная программа преобразования диска с графическим интерфейсом, которая позволяет быстро и безопасно преобразовать диск из исходной системы FAT в систему FAT32.

*Дефрагментация диска.* Дефрагментация диска повышает скорость загрузки и выполнения программ. Быстрый запуск и выключение позволяют быстрее и эффективнее работать, играть и путешествовать по Интернету.

*Объединение с Web.* Проводник Windows 98 и Internet Explorer позволяют объединить ресурсы Web в едином представлении.

*Расширенные средства Web.* Windows 98 делает наиболее продуктивным использование Web за счет применения всех возможностей компьютера к интерактивному содержимому Интернета:

- автоматическое дополнение ранее вызывавшихся адресов Web по мере их ввода;
- улучшенные списки часто посещаемых Web-узлов;

- улучшенный журнал и возможности отслеживания посе... мых Web-узлов;
- поддержка всех основных стандартов Интернета, в том чи... ActiveX, Java и др.;
- повышенная производительность динамического языка HTML, что позволяет сделать Web-страницы более богатыми и интересными.

*Общий доступ к подключению Интернета.* Windows 98, второй выпуск, предоставляет пользователям возможность общего доступа к подключению Интернета для нескольких компьютеров домашней сети. При этом один компьютер имеет непосредственный доступ к подключению Интернета, а запросы от остальных компьютеров домашней сети направляются в Интернет через этот компьютер. Кроме того, общий доступ к подключению Интернета позволяет организовать доступ пользователей Интернета к Web-серверам, а также почтовым и игровым серверам, размещенным в домашней сети.

*Рабочий стол «Active Desktop».* Active Desktop делает возможной настройку рабочего стола, запуск программ, переключение между файлами и отслеживание последних мировых новостей за счет объединения Web и рабочего стола пользователя. Active Desktop позволяет преобразовывать элементы Web в элементы рабочего стола и обновлять их в любое время.

*Электронная почта.* Панель управления Windows 98 включает программу Outlook Express, которая предоставляет защищенные средства для личной электронной почты и подключения к группам новостей. Для запуска Outlook Exprezz нажать кнопку Пуск и выбрать команды Программы и Outlook Express.

*NetMeeting.* Программа NetMeeting позволяет вести разговоры по цифровым каналам связи с родственниками, друзьями и деловыми партнерами по всему миру без больших расходов. Кроме того, NetMeeting делает возможной совместную работу группы пользователей любых приложений для Windows с помощью общей доски, текстовых сообщений и передачи файлов. Если на компьютере установлено необходимое оборудование, становятся возможными телеконференции в режиме реального времени.

### *Функции и состав ОС Windows 95*

*Управление процессами.* В Windows 95 процесс — это либо виртуальная машина MS-DOS, либо любое работающее приложение Windows. Каждый процесс может порождать множество потоков. *Поток* — это последовательность команд в пределах процесса. С... ный планировщик процессов (часть диспетчера виртуальной маш... ны) управляет именно потоками.

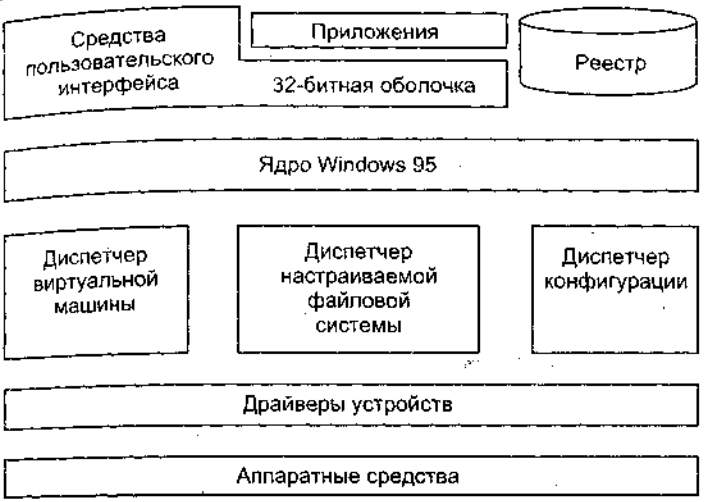



Рис. 2.12. Основные компоненты Windows 95

*Организация файловой системы.* Основной файловой системой является VFAT. Пересмотренная версия Windows 95 OSR2 (OEM Service Release 2) поддерживает файловую систему FAT32, обеспечивающую поддержку жестких дисков более 2 Гбайт и более эффективное распределение дисковой памяти благодаря тому, что размер кластера в ней всего 4 Кбайт.

*Поддержка технологии Plug-&-Play (PNP).* Эта технология была разработана целой группой фирм — разработчиков аппаратного и программного обеспечения. Она предназначается для упрощения Установки и конфигурирования новых устройств. Устройства, соответствующие данной технологии, обязаны «уметь» сообщать ОС о своем наличии и о требуемых для работы ресурсах. С другой стороны, ОС обязана уметь распознавать такие устройства и автоматически выделять требуемые ресурсы.

В Windows 95 основным средством поддержки PNP является диспетчер конфигурации. Он осуществляет идентификацию всех устройств, загружает необходимые драйверы и, с целью выделения ресурсов, обращается к арбитрам ресурсов.

Управление ресурсами, которое включает в себя распределение памяти, процессорного времени, управление доступом к устройствам ввода-вывода и т. д. Эти функции выполняются диспетчером виртуальной машины и ядром системы.

*Реестр.* Важную роль в управлении ресурсами играет реестр.

Реестр — это иерархическая база данных, в которой централизованно хранится вся информация об аппаратных средствах, конкретных приложениях Windows 95 и о настройках пользователя интерфейсной части ОС.

*Драйверы устройств.* В Windows 95 применяется архитектура «универсальный драйвер — минидрайвер».

Универсальный драйвер содержит основную часть кода, необходимого для общения целого класса устройств (например, для принтеров или модемов) с соответствующими компонентами операционной системы (скажем, с подсистемами печати или связи).

Минидрайвер содержит небольшую часть кода, который обеспечивает работу конкретного устройства, принадлежащего данному классу.

*Диспетчер конфигурации.* Он включен в архитектуру Windows 95 для поддержки функциональных возможностей технологии Plug-&-Play.

*Диспетчер виртуальной машины.* VMM (Virtual Machine Manager) выделяет ресурсы каждому приложению и системному процессу, выполняемому на компьютере.

Виртуальная машина представляет собой некую среду в памяти, которая кажется приложению отдельным компьютером с теми же ресурсами, что и у физического компьютера.

*Настраиваемые файловые системы.* Файловая система Windows 95 характеризуется многоуровневой архитектурой, поддерживающей несколько файловых систем (ФС на основе PAT, файловая система CD ROM, файловые системы сторонних разработчиков).

Особенностью файловой системы Windows 95 является поддержка длинных имен файлов (каталогов). В именах файлов можно использовать до 255 символов, включая пробелы и знаки препинания. Запрещенными в именах являются только следующие знаки:

\ — обратный слэш (используется для обозначения пути);

/ — прямой слэш (используется для подстановки ключей);

> — знак «больше» (используется для указания направления вывода);

< — знак «меньше» (используется для указания направления вывода);

: — двоеточие (используется для обозначения имени диска);

? — вопросительный знак (используется в масках поиска),

* — знак «звездочка» (используется в масках поиска);

" — кавычки (в них заключаются полные имена файлов и папок с длинными именами).

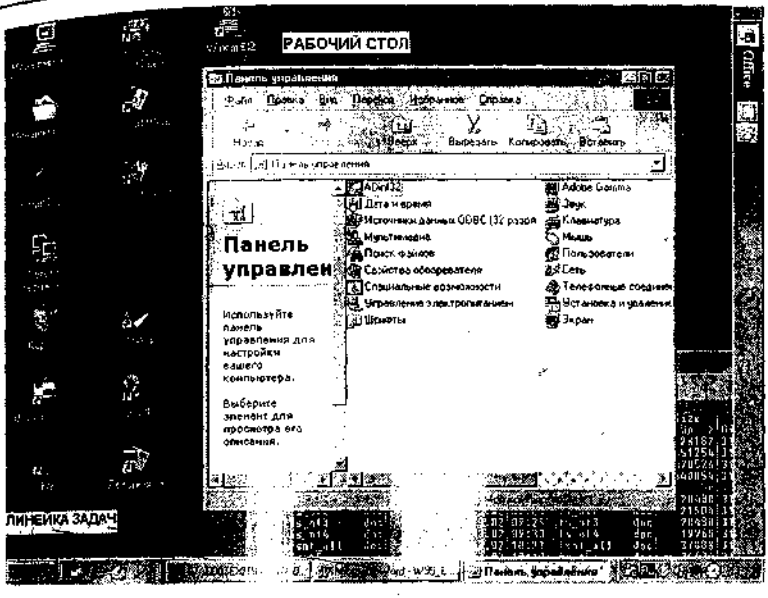



Рис. 2.13. Основные элементы экрана Windows 95/98

Другая особенность — это динамическое кэширование, поддерживаемое файловой системой CD KOM. Это обеспечивает оптимальный баланс между памятью, необходимой приложению, и памятью, выделяемой под дисковый кэш.

***Интерфейс Windows 95.*** Интерфейс — самая важная часть операционной системы (как, впрочем, и любой программы), предопределяющая эффективность решения поставленных задач и способы работы в программе. В понятие «интерфейс» входят следующие компоненты:

- внешний вид — как выглядит на экране оболочка программы;
- набор команд, используемый программой;
- способ подачи команд и реакция программ на них.

*кран Windows95.* Сразу после установки Windows 95 вы видите ран, на котором находится несколько графических объектов. Осн ные элементы нового интерфейса: Рабочий стол и Линейка задач

Рабочий стол — занимает все пространство экрана и вполне соответствует своему названию. Здесь вы можете располагать документы, файлы, ярлыки и прочие объекты для удобной работы с ними ервоначально на Рабочем столе расположены значки Мой компьют- Р. Сетевое окружение, Корзина и т. п.

Линейка Задач — один из основных элементов пользовательского интерфейса Windows 95. Задуманная как инструмент для запуска приложений и переключения между ними, Линейка Задач, в конце концов, приобрела гораздо больше функций. Основные элементы линейки — это кнопка Пуск для вызова Главного меню Windows 95 и кнопки самих приложений для переключения между ними.

Во время установки Windows 95 запрашивает пользователя, какие программы будут использоваться чаще всего. Эти программы помещаются в меню Программы кнопки Пуск и могут запускаться прямо оттуда. По желанию пользователя список наиболее часто используемых программ легко изменяется в любое время из меню Линейки Задач кнопки Пуск. При установке Windows 95 поверх предыдущих версий Windows все программные группы автоматически преобразуются в соответствующие папки, содержимое которых доступно с помощью той же кнопки Пуск.

*Переключение между задачами.* В Windows 95 специально введен механизм (реализуемый с помощью Линейки Задач), позволяющий переключаться между различными запущенными задачами так же просто, как между телевизионными программами. При открытии любого окна на Линейке Задач автоматически возникает кнопка приложения, находящегося в этом окне. Все, что нужно сделать для перехода к новой задаче, — это нажать соответствующую кнопку на Линейке Задач.

Размер кнопок на Линейке Задач автоматически изменяется в зависимости от количества запущенных задач. Если кнопки, по мнению пользователя, становятся очень маленькими, он может настроить Линейку Задач по своему усмотрению. Опции конфигурирования Линейки Задач позволяют:

- поместить Линейку Задач в любое место по периметру экрана;
- изменить размер Линейки Задач, просто перетащив мышкой его внутренний край;
- спрятать Линейку Задач с тем, чтобы он возникал на экране только в том случае, когда курсор мыши попадает на края экрана.

Следует отметить также специально разработанный видеоэффект, включающийся при минимизации задачи и помещении ее в виде кнопки на Линейку Задач и при обратном процессе восстановления задачи.

***Работа с окнами.*** Основным элементом Windows является окно. Все программы, запущенные из-под Windows, работают в окне. Окна, по правилам работы с ними, разделяются на рабочие и диалоговые.

*Рабочее окно* — это, с одной стороны прямоугольная часть экрана, имеющая стандартные элементы управления, а с другой — представитель программы (именно в окне происходит весь обмен информацией между пользователем и программой).

рабочие окна имеют стандартные элементы:

. пиктограмму системного меню;

. название окна;

. кнопки: Свернуть: , Развернуть—восстановить: , Закрыть: ;

. строку меню;

. панель инструментов (может быть включена или выключена);

. рабочее поле окна;

. рамку;

« строку состояния.

Эти управляющие элементы предназначены для реализации *свойств рабочего окна:* изменяемые в размерах; перемещаемые; перекрывающиеся.

*Изменяемый размер* — окно может быть:

- развернуто на весь экран;
- свернуто до значка;
- иметь произвольный размер.

Изменить размер окна можно следующими способами:

- воспользоваться командами системным меню Развернуть, Свернуть, Восстановить;

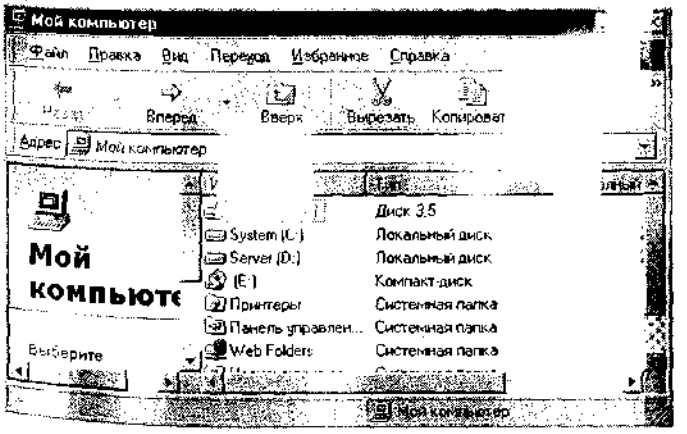



Рис. 2.14. Рабочее окно

• воспользоваться кнопками Свернуть, Развернуть — восстановить, которые дублируют команды из системного меню. При этом команда Свернуть свертывает окно до кнопки на панели задач. Кнопка Развернуть разворачивает окно на весь экран. После выполнения этой операции кнопка превращается в Восстановить: — восстанавливает тот размер окна, который был у него до операций Свернуть или Развернуть. Тот же результат можно получить, если произвести двойной щелчок мышью по названию окна.

В режиме произвольного размера очень важным элементом является рамка окна. Точно указав на нее мышью, можно получить новый вид указателя мыши: ⇔ или ⇳, ⇘, если в этот момент нажать левую кнопку мыши и перемещать ее, то можно изменять размер окна.

*Перемещаемые* — рабочие окна (в режиме произвольного размера) можно произвольно перемещать по экрану. С помощью мыши это делается так: указать на заголовок, зажать левую кнопку мыши. переместить в нужное место и отпустить кнопку. С помощью команды «переместить» из системного меню: подать команду и затем клавишами управления курсором переместить окно.

*Перекрывающиеся* — активное окно перекрывает собой все другие, развернутые на экране, окна (рис. 2.15). Для переключения между окнами можно:

• щелкнуть мышью по любому элементу нужного окна;

• воспользоваться кнопками на панели задач, а для перехода в нужное окно нажать соответствующую кнопку на панели;

• воспользоваться комбинацией клавиш <Alt+Tab>. Нажав и удерживая клавишу <Alt>, нужно нажимать клавишу <Tab>. В центре экрана появляется окно, в котором при каждом нажатии клавиши <Tab> будут переключаться значки всех открытых окон. Если отпустить клавишу <Alt>, активизируется (выйдет на передний план) указанное окно.

*Диалоговые окна.* Диалоговое окно — это окно, предназначенное для ввода пользователем различной управляющей информации. Диалоговые окна выводятся программами для диалога с пользователем.

Свойства таких окон следующие:

• это перемещаемые, но неизменяемые (как правило) по размерам окна;

• как правило, это окна-поплавки, т. е. они всегда находятся на переднем плане и до тех пор, пока пользователь не закончит работу с таким окном, невозможно вернуться назад к работе с программой, выведшей это окно;

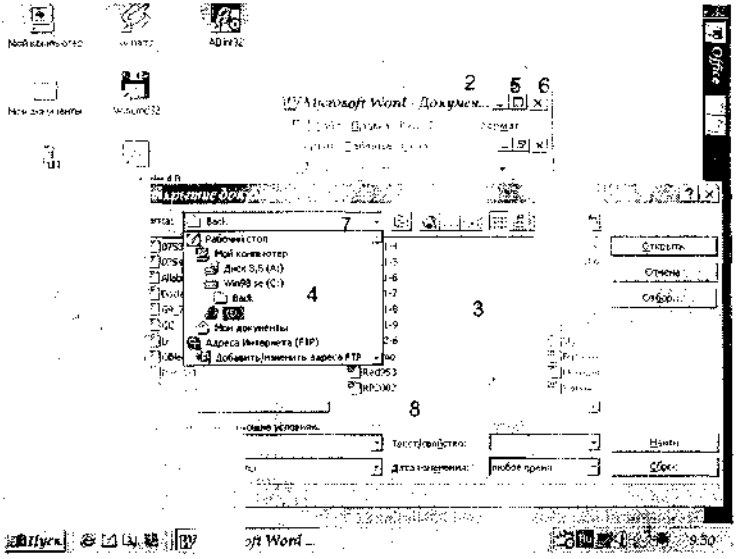



Рис. 2.15. Интерфейс Windows 95/98 с основными элементами: 1 — рабочий стол; 2 — окно приложения; 3 — список; 4, 7 — раскрывающийся список; 5 — кнопка развертывания окна; 6 — кнопка закрытия приложения; 8 — линия прокрутки окна

- такое окно состоит из набора стандартных управляющих элементов (виджетов).

***Работа с файлами.*** В Windows 95 для просмотра «содержимого» компьютера используются совершенно новые механизмы. Например, существует иконка Мой Компьютер, которая запускает приложение для работы с файлами, ориентированное на неопытных пользователей. Правда, простота работы с этой программой совсем не означает, что ее возможности ограничены. Так, файлы и «папки» очень логично ведут себя при операциях Drag&Drop и, кроме того, к файлам и «папкам», как и к традиционным объектам — кускам текста, картинкам и т. д., — могут применяться команды Вырезать, Копировать и Вставить.

*Drag&Drop* — «Перетащи и оставь». Это технология работы с объектами Windows с помощью мыши. Выделив объект (например, Файл или папку) и зажав левую кнопку мыши, вы можете перетаскивать их на другое место. При этом действия Drag&Drop *по умолчанию* приведут к следующим результатам:

- на одном и том же диске произойдет перемещение объекта;
- с диска на диск — копирование;

• если объектом является исполняемый файл, произойдет создание ярлыка на этот файл.

Однако этими процессами можно управлять, используя одновременно с перетаскиванием клавиши:

• <Ctrl> — всегда будет производиться копирование;

• <Shift> — всегда будет производиться перемещение.

Выполняя Drag&Drop правой кнопкой мыши, вы получите по окончании операции контекстное меню, которое позволит выбрать тип производимой операции (Копировать, Переместить, Создать ярлык).

***Проводник.*** Новинкой в Windows 95 является возможность переименования файлов «на месте» простым щелчком мыши на имени файла с последующим введением нового имени. При этом скрытые расширения остаются неизменными. Файлы могут переименовываться в обычных диалоговых окнах приложений, таких, как Открыть файл или Сохранить файл.

Столь же просто и удобно производится работа с *сетевыми ресурсами* — папками и принтерами, расположенными на других ПК (если они выделены в совместное использование на последних). Для этого можно использовать приложение Проводник или Сетевое окружение. Другие ПК, которые в настоящий момент работают в сети, просматриваются точно так же, как и дисковые устройства на собственном ПК. При желании вы можете подключить любой из доступных вам ресурсов на другом ПК в качестве своего сетевого диска и в дальнейшем работать с ним, как с обычным дисковым устройством.

Предоставление сетевым пользователям доступа к вашей «папке»:

• в Проводнике выберите одну из «папок» щелчком правой кнопки мыши;

• выберите пункт Доступ;

• выберите закладку Доступ и ответьте на вопросы в появившемся диалоговом окне.

Подключение «чужого» сетевого ресурса в качестве собственного сетевого диска:

• в Проводнике или в окне Сетевое окружение откройте нужный ПК;

• выделите соответствующий ресурс (ресурсы выглядят точно так же, как и папки на вашем ПК);

• щелкните на нем правой кнопкой мыши и выберите команду Подключить сетевой диск...;

• при желании измените имя диска, установите флажок Автоматически подключать при входе в систему и нажмите OK.

***Ярлыки*** — это средство для повышения эффективности работы, особенно полезное в сетевой среде. Пользователь может создать ярлык на любой объект Windows 95 (файл, программу, диск, утилиту Панель управления, сетевую папку) и поместить его в любом месте интерфейса или внутри документа. При активизации указателя открывается объект, на который этот указатель ссылается. Например, пользователь может создать ярлык на папку Моя сеть и разместить его на экране Windows 95. После этого при активизации ярлыка открывается сетевая папка, находящаяся где-то на сетевом сервере.

Ярлыки выглядят точно так же, как и обычные иконки, за исключением небольшой стрелочки в нижнем левам углу. При ярлыке указателя сам объект, на который он ссылался, остается неизменным. Для создания указателя нужно выбрать объект и выполнить команду Создать ярлык из меню Файл или из контекстного меню, появляющегося при нажатии правой кнопки мыши. Windows 95 следит за переименованием и перемещением файлов, оставляя ярлык действительным даже при изменении характеристик объекта, на которые он ссылается (например, при перемещении файла). Ниже приводится лишь несколько вариантов возможного использования ярлыков.

Ярлыки могут стать развитием концепции иконок из Диспетчера Программ Windows 3.1, ссылаясь на EXE-файл в некотором месте файловой системы. В Windows 95 иконки, появляющиеся в меню Пуск Программы, содержатся в виде ярлыков и в папке Программы. Таким образом, пользователь может сгруппировать ссылки на все свои любимые программы в одном месте, независимо от того, где эти программы установлены на самом деле. Когда ярлык устанавливается или удаляется в «папке» Программы, аналогичный пункт вносится в меню Пуск Программы или удаляется из него.

Помещение наиболее часто используемых ярлыков прямо на рабочее поле экрана особенно популярно среди опытных пользователей. Таким способом они получают быстрый доступ к наиболее часто используемым ресурсам компьютера; особенно упрощается при этом работа с сетью. Ярлыки, вставляемые в приложения, помогают улучшить организацию данных. Так, например, если пользователь вставит в email-сообщение ярлык на большой файл, находящийся где-нибудь в сети, то получатель может открыть этот файл в любой момент двойным щелчком мыши на ярлыке. Это эффективнее, чем включать в послание сам файл, поскольку указатель занимает намного меньше места.

*Смена пиктограммы ярлыка.* Все папки имеют одинаковый вид и отличаются только подписями, а ярлыки имеют, как правило, оригинальную пиктограмму (икону, значок). Можно сменить пиктограмму ярлыка, если щелкнуть по нему правой кнопкой мыши и указать левой кнопкой пункт Свойства в появившемся окне. Затем надо указать пункт Ярлык, нажать кнопку Сменить значок, затем кнопку Обзор, выбрать на диске файл, содержащий пиктограммы, выбрать подходящую пиктограмму и нажать 2 кнопки OK в верхней и нижней частях окна диалога Свойства.

Какие же файлы содержат пиктограммы?

1. Программы, работающие только в Windows (иногда содержат не один, а несколько значков).
2. Библиотеки пиктограмм. Их в Windows 95 как минимум 4:
   1) moricons.dll (c:\windows\);
   2) progman.exe (c:\windows\);
   3) shell32.dll (c:\windows\system\);
   4) pifmgr.dll (c:\windows\system\).
3. Файлы-иконки *.ico. Их можно при необходимости самому создать в графическом редакторе пиктограмм IconEdit из комплекта Norton Desk Top. В этом редакторе можно создавать целые библиотеки пиктограмм (файлы *.nil).

***Окна свойств.*** С любым объектом пользовательского интерфейса ассоциируется контекстно-зависимый набор свойств, который может быть просмотрен и изменен с помощью меню Файл/Свойства или же с помощью контекстного меню, которое появляется при нажатии правой кнопки мыши. Легкодоступные окна свойств с логично подобранными параметрами получили очень много одобрительных откликов среди опытных пользователей. Наилучшим образом пояснят работу окон свойств несколько примеров.

Изменение метки тома:

- в Проводнике или Мой компьютер щелчком правой кнопкой мыши выберите один из жестких дисков;
- выберите Свойства;
- введите новое имя в окошке Метка тома и нажмите OK;
- нажмите F5 (команда Обновить показания в окне).

Изменение значка ярлыка:

- щелкните правой кнопкой мыши на любом ярлыке и выберите Свойства;
- выберите закладку Ярлык (для Windows-приложений) или Программа (для DOS-приложений);
- нажмите кнопку Сменить значок;
- выберите новую иконку и нажмите Ок.

*Функции правой кнопки мыши.* Наряду с окнами свойств, меню, которые вызываются щелчком правой кнопки мыши, также являются вездесущими контекстно-зависимыми элементами Windows 95 (поскольку при желании пользователи легко могут поменять местами кнопки мыши, везде в тексте под правой кнопкой понимается «второстепенная» кнопка). Правда, новички неохотно пользуются этим мощным механизмом, однако среди опытных пользователей он оказался очень популярным. Ввиду контекстной зависимости меню, вызываемых нажатием правой кнопки мыши, лучше всего пояснить работу описываемого механизма на примерах:

Изменение характеристик рабочего поля экрана Windows 95:

- щелкните правой кнопкой мыши на свободном месте экрана; выберите Свойства;
- сделайте необходимые изменения и нажмите OK.

Минимизация и реорганизация окон:

- щелкните правой кнопкой мыши на свободное месте Линейки Задач;
- выберите Свернуть все для сворачивания всех окон или Сверху вниз для расположения окон последовательно друг под другом.»

Создание нового ярлыка:

- щелкните правой кнопкой мыши на объекте, на котором вы хотите создать ярлык, — выберите Создать ярлык.

*Меню* ***Start*** *(****Главное*** *меню).* Вызывается кнопкой Пуск (Start) на линейке задач. Оно служит стартовой площадкой не только для поставляемых с Windows 95 бесплатных приложений, но и для инсталлированных отдельно программ. После появления меню из него можно выбрать элемент, применяя один из двух способов:

- щелкнуть на элементе мышью;
- при работе с клавиатурой найти подчеркнутую букву в имени элемента, например Р в Ргодгатз, и нажать клавишу с этой буквой. Если подчеркнутой буквы нет, с помощью клавиш управления курсором подсветить элемент и нажать клавишу <Enter>.

Меню Start содержит *команды, принадлежности и приложения.*

*Команда* — это средство, встроенное в саму систему Windows 95.

*Принадлежности* — это небольшие программы, которые поставляются с Windows 95.

*Приложения* — это программы, которые продаются отдельно и должны быть инсталлированы в компьютере. В большинстве случаев процедура инсталлирования просто добавляет имя приложения в меню Start.

Меню Start состоит из неизменяемой части и меню пользователя.

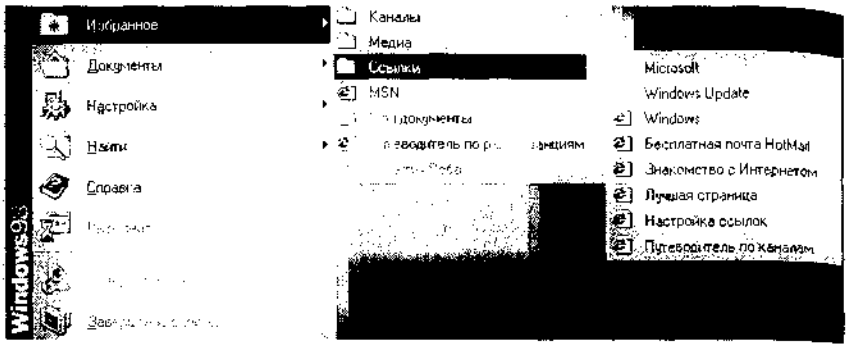



Рис. 2.16. Главное меню (Меню Пользователя)

Элементы главного меню:

Избранное (Меню Пользователя). В Меню Пользователя пользователь может расположить значки приложений, документов, папок и файлов, наиболее необходимых пользователю при работе, для быстрого их вызова. Заполняется при помощи: Пуск — Настройка — Панель Задач, закладка Настройка меню или простым перетаскиванием значка нужного объекта на кнопку Пуск.

Неизменяемая часть Главного меню включает:

Programs (Программы). Этот пункт Главного меню содержит сложное иерархическое подменю, состоящее из групп программ и самих программ. Предназначено оно для запуска приложений, проинсталлированных в Windows 95 (рис. 2.17).

Document (Документы). Содержит меню из 15 последних документов, с которыми пользователь работал в любых приложениях. (Под документом подразумевается любой файл, с которым пользователь работал в программе. Им может быть электронная таблица, рисунок, письмо и т. д.) При выборе документа из этой папки Windows запускает соответствующую программу и автоматически загружает документ.

Setting (Настройка). Содержит несколько программ, с помощью которых можно задать различные параметры Windows 95. Включает в себя подменю Панель Управления, Принтеры, Панель Задач.

Панель Управления. Задает параметры системной даты, времени, позволяет производить настройку клавиатуры, мыши, а также установку принтера и модема. Кроме того, позволяет настроить звуковые сигналы на соответствующие события Windows 95. Также предоставляет возможность установить конфигурацию сети, идентифицировать компьютер, определить уровень доступа к ресурсам в сети.

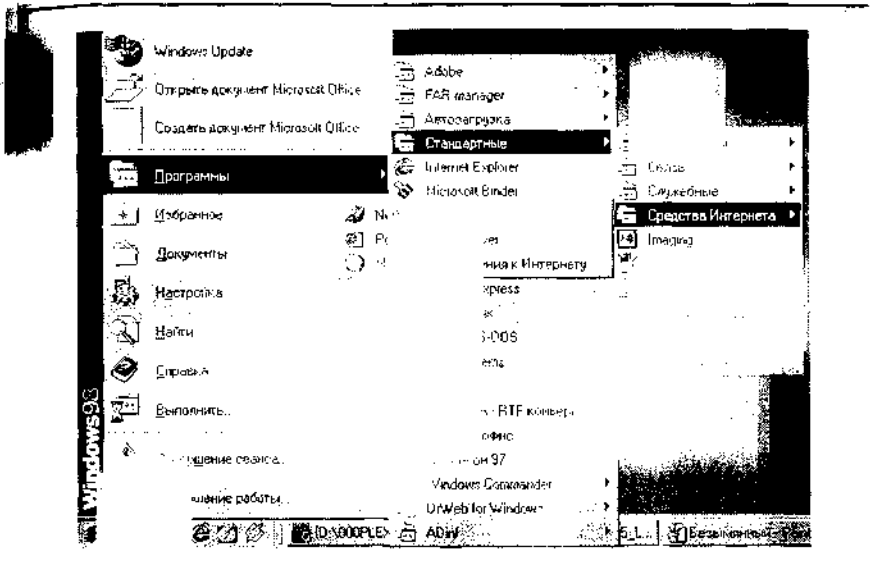



Рис. 2.17. Главное меню (подменю Программы)

Принтеры. Это подменю вызывает один из элементов Панели Управления — модуль для установки и настройки принтеров.

Панель Задач включает в себя две закладки:

1. Параметры Панели Задач. Состоит из четырех пунктов:
   - расположить поверх всех окон;
   - автоматически убрать с экрана;
   - мелкие значки в главном меню;
   - отображать часы.
2. Настройка меню.

Включает в себя настройку Главного меню, позволяющую добавлять и удалять пункты меню и управление параметрами панели задач.

Find (Поиск). Содержит инструменты, помогающие искать объекты в компьютере. Состоит из подменю Файлы и Папки и подменю Компьютер. Позволяет разыскать нужный файл по имени, размещению, дате изменения, размеру. Все данные выводятся в специальное окно, характерное для Windows 95, позволяющее двигать файлы методом «переместить-и-отпустить» и переименовывать их (рис. 2.18).

Help (Справка). Вызывает справочную систему Windows 95 (рис. 2.19). По сравнению с Windows 3.x справочная система Windows 95 была полностью переписана. Это вызвано тем, что система помощи в Windows 3.x оказалась сложной для освоения и использования. Она имела три основные функции — Contents, Search и Glossary, и пользователям не всегда было ясно, какую из них следует ис-

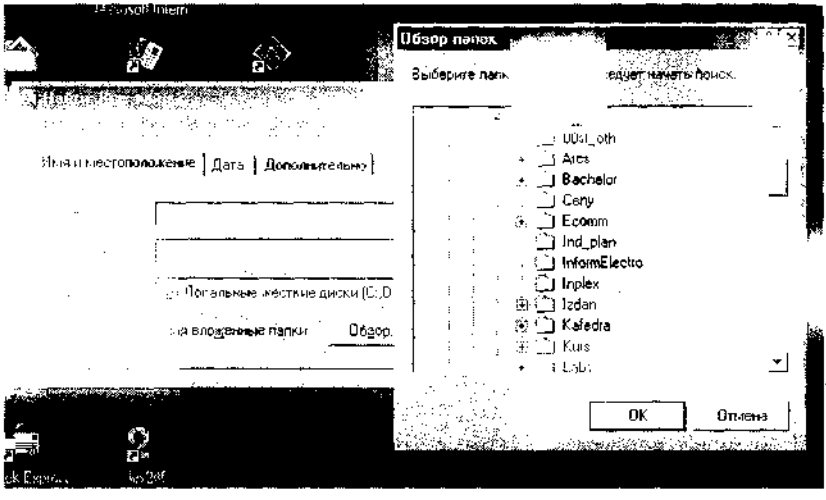



Рис. 2.18. Окно поиска файлов

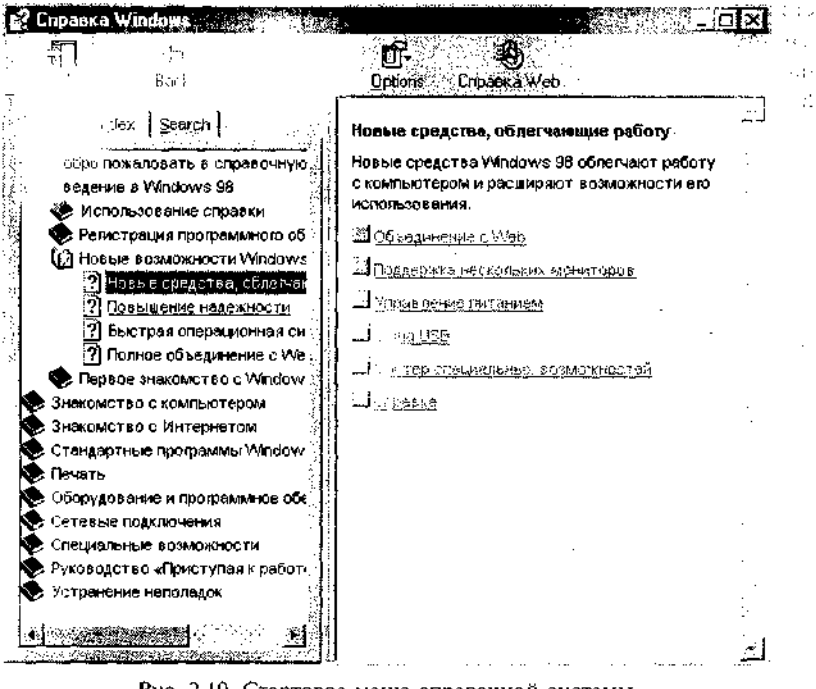



Рис. 2.19. Стартовое меню справочной системы

пользовать в том или ином конкретном случае. Поэтому в Windows 95 функции Search больше не существует, и теперь справочная система содержит только команды Contents и Index.

При вызове функции Contents появляется окно, содержимое которого похоже на оглавление книги. Темы верхнего уровня сопровождаются иконками с рисунком книги и могут раскрываться для рассмотрения подтем, которые отображаются иконками с изображением листов. Многие главы помощи снабжены специальными разделами Подсказки и Хитрости. Кроме того, все главы укорочены настолько, чтобы они помещались на одном экране и пользователям не приходилось пользоваться прокруткой для рассмотрения больших и сложных тем.

В справочной системе Windows 95 появились специальные кнопки-указатели, которые переносят пользователя прямо в то место операционной системы, о котором рассказывается в справке. Так, например, пользователь, читающий в разделе Помощь о том, как переставить часы компьютера, может нажатием одной кнопки попасть в соответствующее место Панели Управления прямо из справочной системы. Кроме того, в Windows 95 добавлен новый механизм, уже опробованный в последних версиях Word и Excel компании Microsoft. При нажатии специальной кнопки с изображением вопросительного знака (или выборе соответствующего пункта из контекстного меню, которое вызывается правой кнопкой мыши) курсор принимает форму «?». Если после этого навести его на какой-нибудь объект, справочная система выдаст краткое описание выбранного объекта.

Кип (Выполнить). Этой командой пользуются, если нужная программа отсутствует в папках меню Start. После ее выбора нужно ввести дисковый накопитель, где находится программа, ее папку и имя файла, запускающего программу.

Shut Down (Завершение работы). Эта команда позволяет произвести корректный выход из среды Windows 95.

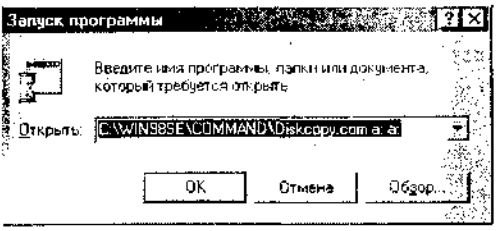



Рис. 2.20. Окно команды запуска приложений

***Панель Управления*** (рис. 2.13, 2.21) — специальная системная папка, в которой сосредоточены все средства для настройки Windows *95.*

Именно здесь устанавливаются *общие ресурсы Windows*, т. е. такие ресурсы, которые используются *всеми* приложениями, работающими в данной ОС (операционной системе). Это означает, что данные установки сохраняются самой операционной системой и в дальнейшем все приложения используют эти установки для отображения информации, например цветовая гамма (цвета заголовка окна, рабочего стола, объемных объектов, рабочих областей приложений и т. д.), национальные установки и многое другое. Кроме того, здесь же производятся и все остальные настройки, такие, как установка нового оборудования, настройка сетевых параметров, добавление и удаление компонентов Windows и пр. Панель управления содержит значки (иконки) для запуска специальных модулей, позволяющих производить перечисленные настройки. Число включенных в нее модулей зависит от полноты инсталляции системы Windows 95. Каждая иконка контрольной панели запускает соответствующий модуль, меняющий установки системы. При этом появляется то или иное диалоговое окно.

Ниже перечислены основные пути запуска Панели Управления.

- Пуск — Настройка — Панель Управления;
- Мой компьютер — Панель Управления;
- Проводник — в левом окне выбрать Панель Управления;
- в любой папке (например, Рабочий стол) можно создать ярлык на Панель Управления.

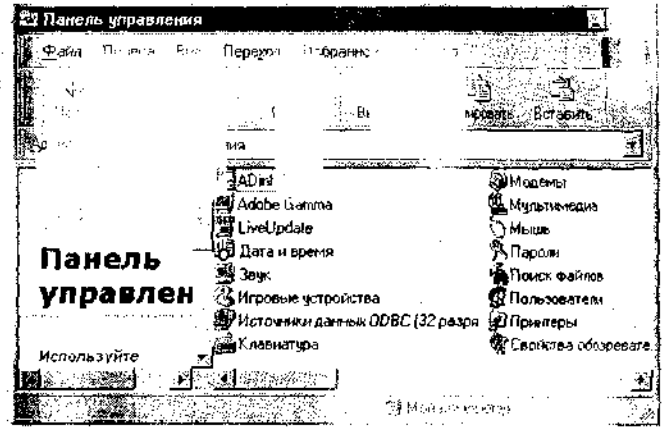



Рис. 2.21. Окно Панели Управления

Рассмотрим наиболее важные модули из Панели Управления.

Языки и стандарты. Данный модуль предназначен для установки *общего ресурса Windows*«Регион пребывания», что включает в себя: национальные форматы валюты, представления даты и времени, символа, который используется в качестве разделителя между элементами списка, символа-разделителя между дробной и целой частью числа. *Например*, если установить регион пребывания — Россия, то названия месяцев (во всех приложениях, где вы будете выводить календарную дату) будут: «январь», «февраль» и т. д., если установить — Украина, то: «січень», «лютий» и т. д. Если установить ресурс «*разделитель целой и дробной частей числа — запятая*», то запись: «1.5» уже не будет восприниматься приложением (например, Excel) как число, а будет интерпретироваться, как текст или календарная дата.

Системный модуль «Принтеры». В среде Windows, в отличие от DOS, ни одно приложение не выполняет печать самостоятельно. Приложения только формируют задания на печать и отправляют его специальному приложению — Диспетчеру Печати. Windows 95 представляет нам диспетчера как установленный принтер конкретной модели. В пакет установки Windows (инсталляционный пакет) входят драйверы, обеспечивающие работу более 800 типов принтеров самого различного типа — от простых 9- и 24-игольчатых матричных принтеров до самых современных цветных струйных и лазерных принтеров с высоким разрешением.

В системе должен быть установлен хотя бы один принтер для того, чтобы приложения Windows могли производить печать документов или показывать предварительный просмотр документа перед печатью. Из любой программы доступна печать как принтерами, установленными на вашем ПК (локально), так и сетевыми принтерами, подключенными к другому ПК или серверу в компьютерной сети.

Один из установленных принтеров назначается *текущим,* т. е. используемым всеми приложениями по умолчанию (если пользователь не указал другой). Этот принтер обеспечивает все приложения Windows информацией о состоянии принтера, о размерах используемой бумаги, ее ориентации и т. д.

Системный модуль Принтеры выглядит на первый взгляд как окно обычной папки и позволяет:

- устанавливать драйверы принтеров (т. е. добавлять в систему принтер конкретной модели, например Epson FX-1050);
- настраивать параметры принтеров: команда Свойства из контекстного меню на значке соответствующего принтера.

Шрифты. Windows выгодно отличается от MS-DOS возможностью применения множества масштабируемых шрифтов (фонтов). Масштабируемость означает возможность изменения размера того или иного шрифта при сохранении достаточно гладкого вида символов. Основным видом фонтов в Windows 95 являются хорошо известные пользователям прежних версий Windows фонты True-type, сохраняющие свое очертание при изменении размеров в широких пределах.

Системная папка Шрифты предназначена для добавления/удаления шрифтов в среде Windows 95 (рис. 2.22). Она запускает специальный модуль для работы с реально существующим каталогом «%WINDIR%/Fonts», с помощью которого можно вывести окно с файлами всех наборов фонтов, инсталлированных в вашей системе, — это наборы, поставляемые с Windows 95, наборы, инсталлированные отдельно, и наборы, инсталлированные некоторыми приложениями. Открыв любой файл, можно получить диалоговое окно с подробной характеристикой заданного набора шрифтов.

Нетрудно заметить, что помимо информации о наборе фонтов в окне просмотра даны примеры вывода фонтов с различным размером. Опция Done позволяет продолжить просмотр, а очень удобная опция Print — тут же распечатать набор принтером. Это полезно для суждения о том, стоит ли применять данный набор для определен-

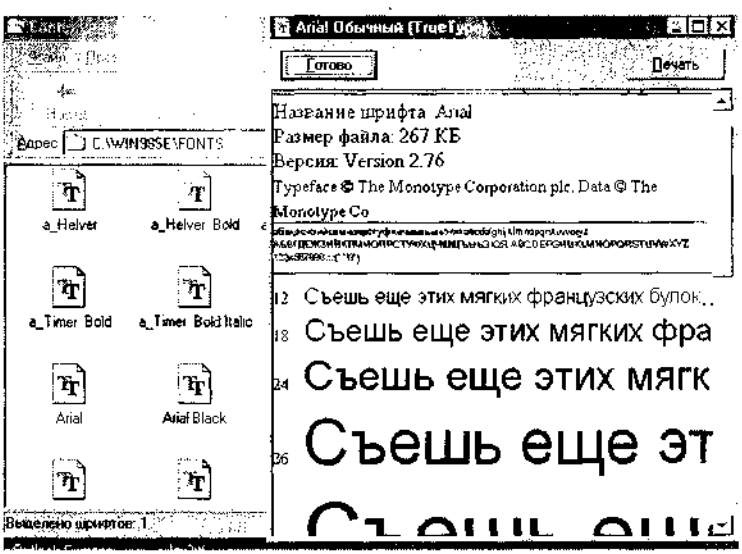



Рис. 2.22. Окно управления шрифтами

ных целей: например печати визитной карточки, письма или приглашения на день рождения.

Установка/удаление программ. Одна из часто используемых функций — установка в Windows новых приложений. Она была возможна и в прежних версиях Windows, но с одним серьезным недостатком — от установленных и затем стертых приложений все же оставались файлы, попусту расходующие память жесткого диска и захламляющие файловую систему.

В Windows 95 установка новых программ и переустановка самой системы могут происходить разными способами. К примеру, вы можете задать новый ярлык и указать, какой пусковой файл он запускает. Используя команду «Выполнить...» Главного меню и запустив файлы Setup или Install приложений, можно выполнить их установку или инсталляцию, пополнив таким образом набор приложений, с которыми будет работать Windows 95.

Вместе с тем наиболее правильно использовать специальный модуль ПУ Установка/удаление программ, который применяется для установки новых программ, а также их корректного удаления из операционной системы.

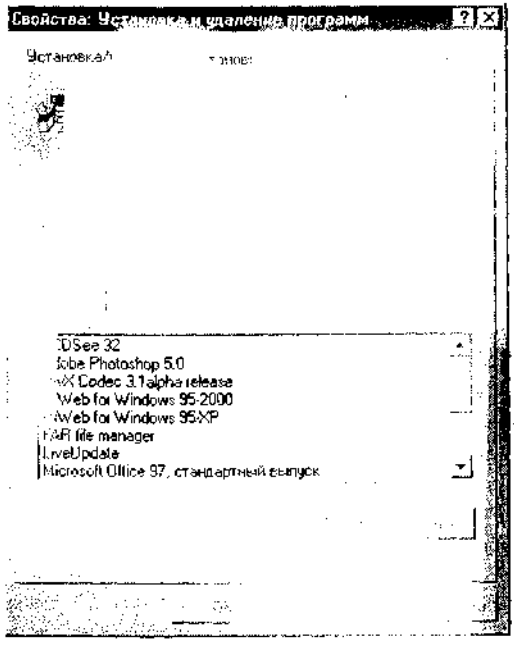



Рис. 2.23. Установка и удаление программ

Диалоговое окно этого модуля на первой закладке отображает список всех установленных в ОС приложений. Выбрав нужное приложение в списке и нажав кнопку Добавить/удалить, пользователь может запустить окно Мастера, появляющееся удалить данное приложение или изменить состав его компонентов (ели это предусмотрено самим приложением, как, например, MS Office).

При нажатии кнопки Установить запускается Мастер по установке нового приложения. В этом случае можно проводить установку программ с гибкого диска или CD-ROM. Поскольку инсталляция различных программ имеет свои отличительные особенности, для ее проведения и используется Мастер. Нажимая кнопку Next в окне Мастера, можно пройти все шаги по инсталляции той или иной программы. Инсталляция крупных программ занимает много времени — подчас до десятков минут и требует слежения за ней. Тем не менее при применении Мастеров она довольно проста и редко сопровождается сбоями

Компоненты самой Windows 95 добавляются и удаляются через закладку Установка Windows. На ней отображается список тех программ, которые включены в инсталляционный пакет Windows и которые пользователь может по своему усмотрению устанавливать или нет. Установленные компоненты отмечены «галочкой», а не установленные — нет. Чтобы добавить нужный компонент (например, Специальные возможности), просто поставьте в списке против него отметку и нажмите OK. При выполнении установки Windows попросит у вас указать путь к инсталляционному пакету. (Его необходимо просто знать.)

Система. Модуль Система позволяет просматривать и изменять аппаратную конфигурацию системы. Окно свойств системы обеспечивает доступ ко всем параметрам компьютера в целом и отдельных устройств (рис. 2.24).

Закладка Общие — чисто информационная, отображает общие сведения о системе.

Закладка Устройства отображает дерево устройств (физического оборудования), установленных на вашем ПК. В корне дерева находятся названия типов устройств, а ниже — модель конкретного оборудования. Выделив какое-нибудь устройство, можно получить окно с его свойствами, для чего нужно воспользоваться кнопкой Свойства. Кнопка Обновить обновляет данные дерева устройств.

Кнопка Удалить удаляет выделенное устройство из текущей конфигурации оборудования.

Закладка Конфигурации используется для создания нескольких альтернативных конфигураций оборудования на одном ПК. (*Испо-*

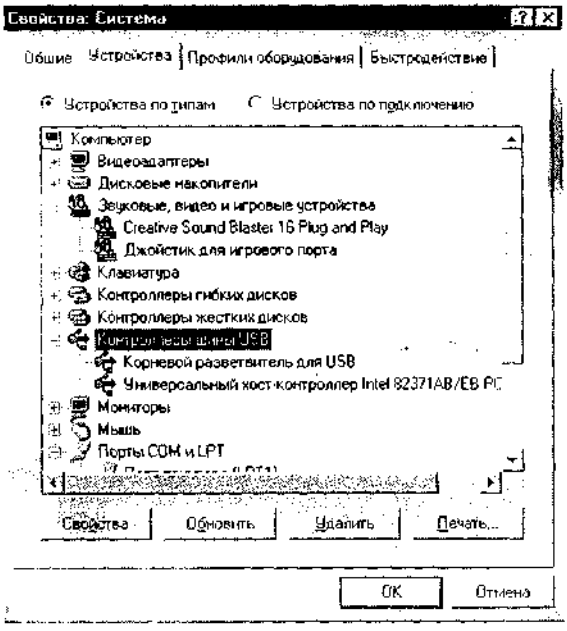



Рис. 2.24. Закладка Устройства

льзуется крайне редко.) Использование нескольких конфигураций оборудования позволяет всякий раз при изменении состава оборудования загружать правильный набор драйверов. Это необходимо в первую очередь при работе с переносным компьютером, который может, например, работать как *в доке* (системе стационарного подключения), так и *вне дока.*

Закладка Быстродействие отображает сведения о памяти (ОЗУ), системных ресурсах (имеется в виду, сколько свободно памяти ОЗУ+виртуальная память), файловой системе, а также на этой закладке появляется список причин, замедляющих работу системы.

Если работа системы ничем не замедляется, вместо перечня проблем в окне присутствует сообщение «Система настроена на оптимальное быстродействие».

Если причины существуют, то выводится список драйверов, работа с которыми замедлена. Чтобы получить более подробные разъяснения по ним, выберите одно из сообщений и нажмите кнопку Сведения. (Появляется только при наличии списка причин.)

Кнопка Файловая система позволяет управлять параметрами кэширования при выполнения операций ввода/вывода на накопители.

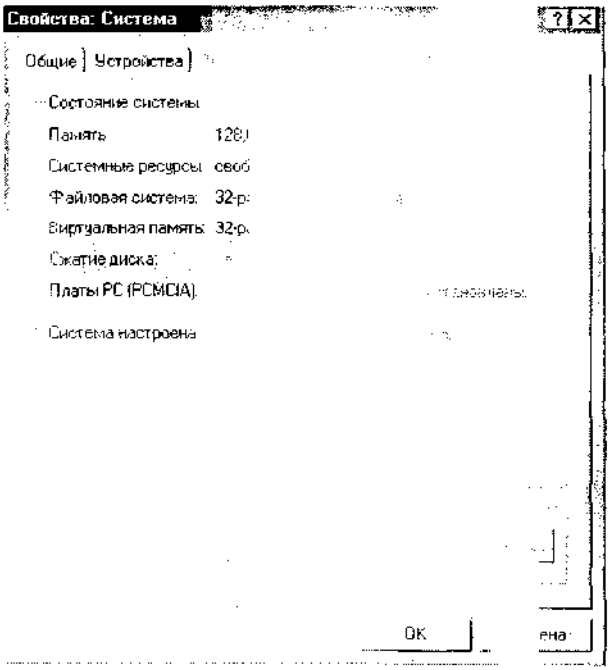



Рис. 2.25. Закладка Быстродействие

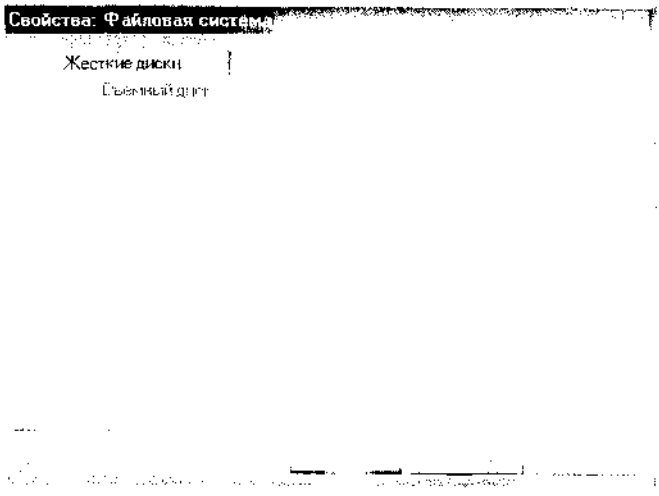



**Рис.** 2.26. Закладка Файловая система — устранение неполадок

Кнопка Графика — настройка параметров ускорения видеоадаптера.

Кнопка Виртуальная память — применяется для назначения диска, где Windows располагает свой *«файл подкачки»* и его размера (используется ОС для виртуального расширения объема используемой памяти). Этот файл для Windows очень важен. Он имеет большой размер (до 2х ОЗУ), и если на системном диске осталось мало дискового пространства, может возникнуть необходимость воспользоваться этой закладкой. Хотя, как правило, ОС сама регулирует эти вопросы.

*Запуск Windows 95.* При установке на компьютер Windows 95 ОС конфигурируется таким образом, чтобы после включения питания происходила автоматическая загрузка всех необходимых модулей вплоть до графической оболочки.

Операционная система Windows 95, как и любая DOS, загружается при включении компьютера. После загрузки ОС загружается оболочка Windows 95 и на экране монитора появляется Рабочий стол (Desk-top). Это первоначальное (главное) окно Windows 95, в котором расположены все другие окна и значки.

Для автоматического запуска Windows 95 при загрузке компьютера необходимо в конце файла autoexec.bat поместить команду win. Этой же командой win можно всегда запустить оболочку Windows 95, в том числе, когда загрузка оканчивается появлением Norton Commander или приглашения DOS C:\>. Если компьютер загружается в Norton Commander (в конце файла autoexec.bat стоит команда NC), то для запуска оболочки Windows 95 достаточно просто выйти из Norton Commander — P10, Enter. Выйти из ОС Windows 95 полностью при включенной машине нельзя, можно лишь выйти из ее графической оболочки в режиме эмуляции MS-DOS.

Если при загрузке ОС Windows 95 при появлении на мониторе слов «Starting Windows 95» быстро нажать клавишы Shift-F5, то файлы config.sys и autoexec.bat исполняться не будут и загрузка сразу закончится появлением приглашения DOS C:\>. Это следует делать, если в файлах config.sys и autoexec.bat есть грубые ошибки и загрузка прерывается. Тогда надо вручную загрузить Norton Commander и исправить ошибки, отредактировав указанные файлы. Можно также при ошибках в загрузке нажать клавишы Shift-F8 и просматривать директивы файлов config.sys и autoexec.bat, давая команду на их исполнение (Yes) или пропуск (N0). Если при загрузке нажать клавишу P5, то Windows 95 будет загружаться в защищенном режиме (Safe Mode) и можно исправить ошибки в настройке Windows 95 при помощи Панели управления. При нажатии

клавиши F8 можно самому выбрать режим загрузки Windows 95 из предложенного меню:

```
Microsoft Windows 95 Startup Menu
1. Normal.
2. Logged (\BOOTLOG.TXT).
3. Safe mode.
4. Step-by-step confirmation.
5. Command prompt only.
6. Safe mode command prompt only.
Previous version of MS-DOS
Enter a choice: F5=Safe mode Shift+F5=Command prompt
Shift+F8=Step-by-step confirmation
```

Например, можно выбрать пункт меню Step-by-step confirmation и в пошаговом режиме проследить процесс загрузки Windows 95.

*Связь Windows 95 с Windows 3.1.* В составе Windows 95 имеются Диспетчер программ (progman.exe) и Диспетчер файлов (winfile.exe), по своим свойствам аналогичные Windows 3.1. Запустив указанный Диспетчер программ из-под Windows-95, можно работать практически также, как в Windows 3.1, например создавать Программные группы (Окна) и внутри них Программные элементы, запускать прикладные программы. То же самое касается и Диспетчера файлов. Пиктограмму Диспетчера программ можно разместить на Рабочем столе Windows 95 по аналогии с экраном Windows 3.1. Панель задач Windows 95 в этом режиме сохраняется. При выходе из Диспетчера программ вы попадаете опять на Рабочий стол Windows 95.

*Сеанс связи с MS-DOS.* В режиме эмуляции MS-DOS (кнопка Пуск, Завершение работы) открывается прямой доступ к диску C:. В этом режиме запускают некоторые программы DOS, которые из-под Windows 95 не работают. Возвращение назад в Windows-команда Exit. При этом происходит перезагрузка Windows 95. Кроме того, возможен Сеанс связи с MS-DOS без перезагрузки Windows 95 (кнопка Пуск, Программы, Сеанс MS-DOS). В этом режиме тоже можно запускать ряд программ DOS. Для возвращения в Windows 95 надо ввести команду Exit.

***Краткие сведения об архитектуре Windows 95/98.*** Windows 95 представляет собой продукт эволюционного развития системы Windows 3.1x. Хотя она и несет в себе много важных изменений по сравнению с 16-разрядной архитектурой Windows, в ней сохранены некоторые важнейшие свойства ее предшественницы. Результатом стало появление гибридной ОС, способной работать с 16-разрядными прикладными программами Windows, программами, уна-

следованными от DOS, и старыми драйверами устройств реального режима и в то же время совместимой с истинными 32-разрядными прикладными программами и 32-разрядными драйверами виртуальныхустройств.

Среди наиболее важных усовершенствований, явившихся в Windows 95, — изначально заложенная в ней способность работать с 32-разрядными многопотоковыми прикладными программами, защищенные адресные пространства, вытесняющая многозадачность, намного более широкое и эффективное использование драйверов виртуальных устройств и возросшее применение 32-разрядных хипов для хранения структур данных системных ресурсов. Ее наиболее существенный недостаток состоит в относительно слабой защищенности от плохо работающих программ, содержащих ошибки.

Каждая собственная прикладная программа Windows 95 видит неструктурированное 4-Гбайт адресное пространство, в котором размещается она сама плюс системный код и драйверы Windows 95. Каждая 32-разрядная прикладная программа выполняется так, как будто она монопольно использует весь ПК. Код прикладной программы загружается в это адресное пространство между отметками 2 и 4 Гбайт. Хотя 32-разрядные прикладные программы «не видят» друг друга, они могут обмениваться данными через буфер обмена (Clipboard), механизмы DDE и OLE. Все 32-разрядные прикладные программы выполняются в соответствии с моделью вытесняющей многозадачности, основанной на управлении отдельными потоками. Планировщик потоков, представляющий собой составную часть системы управления виртуальной памятью (VMM), распределяет время среди группы одновременно выполняемых потоков на основе оценки текущего приоритета каждого потока и его готовности к выполнению. Вытесняющее планирование позволяет реализовать намного более плавный и надежный механизм многозадачности, чем кооперативный метод, используемый в Windows 3.1x.

Системный код Windows 95 размещается выше границы 2 Гбайт. В пространстве между отметками 2 и 3 Гбайт находятся системные библиотеки DLL кольца 3 и любые DLL, используемые несколькими программами. (В 32-разрядных процессорах фирмы Intel предоставляются четыре уровня аппаратной защиты, поименованные, начиная с кольца 0 до кольца 3. Кольцо 0 наиболее привилегировано.) Компоненты кольца 0 в системе Windows 95 отображаются в пространство между 3 и 4 Гбайт. Эти важные участки кода с максимальным уровнем привилегий содержат подсистему управления виртуальными машинами (VMM), файловую систему и драйверы VxD.

В Windows 95 достигнут баланс между производительностью, совместимостью и надежностью. Она обеспечивает быстрое исполнение прикладных программ Win16, Win32 и Dos и совместима с драйверами устройств реального режима. Несмотря на лучшую, чем у Windows 3.1, защищенность, она остается уязвимой с нескольких сторон.

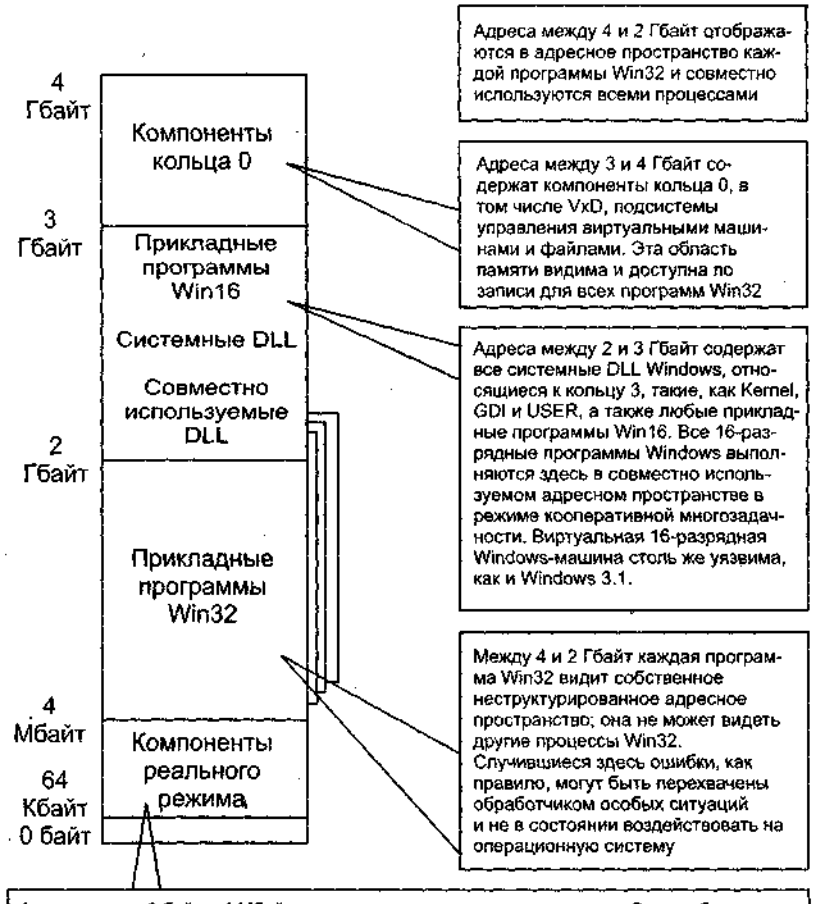



Рис. 2.27. Модель памяти Windows 95

Область памяти между 2 и 4 Гбайт отображается в адресное пространство каждой 32-разрядной прикладной программы, т. е. оно совместно используется всеми 32-разрядными прикладными программами в вашем ПК. Такая организация позволяет обслуживать вызовы API непосредственно в адресном пространстве прикладной программы и ограничивает размер рабочего множества. Однако за это приходится расплачиваться снижением надежности. Ничто не может помешать программе, содержащей ошибку, произвести запись в адреса, принадлежащие системным DLL, и вызвать крах всей системы.

В области между 2 и 3 Гбайт также находятся все запускаемые вами 16-разрядные прикладные программы Windows. С целью обеспечения совместимости эти программы выполняются в совместно используемом адресном пространстве, где они могут испортить друг друга так же, как и в Windows 3.1x.

Адреса памяти ниже 4 Мбайт также отображаются в адресное пространство каждой прикладной программы и совместно используются всеми процессами. Благодаря этому становится возможной совместимость с существующими драйверами реального режима, которым необходим доступ к этим адресам. Это делает еще одну область памяти незащищенной от случайной записи. К самым нижним 64 Кбайт этого адресного пространства 32-разрядные прикладные программы обращаться не могут, что дает возможность перехватывать неверные указатели, но 16-разрядные программы, которые, возможно, содержат ошибки, могут записывать туда данные.

## 2.4. Операционные системы Windows NT/2000

Операционная система Windows NT или New Technology была создана группой разработчиков под руководством Дэйва Катлера, ранее работавшего в DEC над проектом VMS (кстати, довольно часто используемая аббревиатура WNT получается из VMS сдвигом букв V, M, 3 по алфавиту на одну: V → W, M → N, S → T). Дэйв Катлер пришел в Microsoft в 1988 году специально для работы над проектом NT. NT, в отличие от остальных ОС Microsoft, в некотором смысле проект одного человека, она наиболее законченная ОС из всего того, что они выпустили.

Windows N7 является 32-разрядной операционной системой с приоритетной многозадачностью. В качестве фундаментальных компонент в состав операционной системы входят средства обеспечения безопасности и развитый сетевой сервис. Windows N7 также

обеспечивает совместимость со многими другими операционными и файловыми системами, а также с 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 мультипроцессорной конфигурацией.

Знакомым в Windows NT является только внешний облик. За графическим пользовательским интерфейсом скрываются новые мощные возможности.

*Задачи, поставленные при создании Windows NT.* Система Windows NT не является дальнейшим развитием ранее существовавших продуктов. Ее архитектура создавалась заново с учетом предъявляемых к современной операционной системе требований. Особенности системы, разработанной на основе этих требований, следующие.

Стремясь обеспечить *совместимость* новой операционной системы, разработчики Windows NT сохранили привычный интерфейс Windows и реализовали поддержку существующих файловых систем (таких, как PAT) и различных приложений (написанных для M8-DOS, O8/2 1.x, Windows 3.x и POSIX). Разработчики также включили в состав Windows NT средства работы с различными сетевыми средствами.

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 M1P8 R4000, Digital Alpha AXP и Pentium серии P54 и выше.

*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.

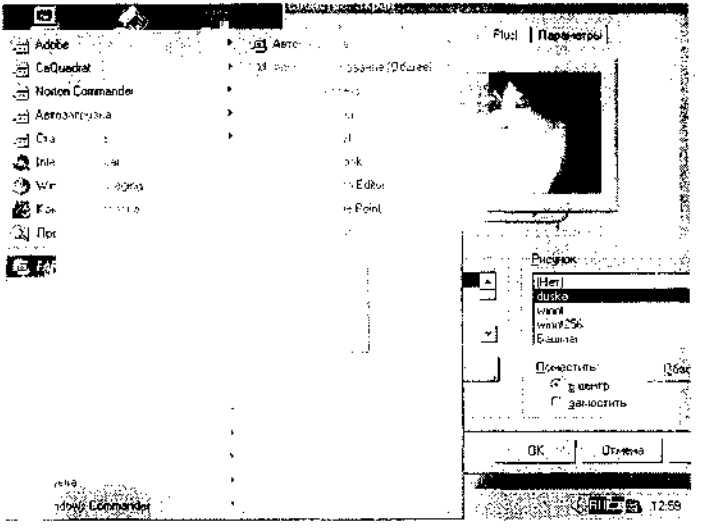



Рис. 2.28. Главное меню Windows NT, подменю Программы. Окно установки свойств экрана

Windows NT имеет однородную *систему безопасности* (security), удовлетворяющую спецификациям правительства США и соответствующую стандарту безопасности B2. 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 и обеспечивает защиту с помощью встроенной системы безопасности и усовершенствованных методов управления памятью.

Возможности *локализации* (allocation) представляют средства для работы во многих странах мира на национальных языках, что дости-

гается применением стандарта Unicod (разработан международной организацией по стандартизации — ISO).

Благодаря модульному построению системы обеспечивается *расширяемость* Windows NT, что позволяет гибко осуществлять добавление новых модулей на различных уровнях операционной системы.

***Интерфейс Windows NT.*** При входе в систему пользователь видит образы, практически совпадающие с привычной картиной для Windows 95/98, - Рабочий Стол, Главное Меню, Линейку Приложений (именуется здесь Панель задач).

На рис. 2.28—2.31 приведены типичные экраны и меню, которые наблюдает пользователь Windows NT:

- меню программ, запускаемое из Главного меню;
- экран помощи, также включаемой из Главного меню;
- режим поиска файлов;
- окно настройки линейки приложений (панели задач) из рубрики «Настройки» Главного меню.

Внешне Windows NT 4.0 аналогична Windows 95. Единственный признак, позволяющий с первого взгляда различить эти две системы, — стартовое меню, где указано, в какой среде вы работаете. Достоинства и недостатки Windows 95 уже хорошо известны, однако необходимо отметить, что новый облик, как ни странно, не слишком облегчил работу. Например, если раньше многие функции

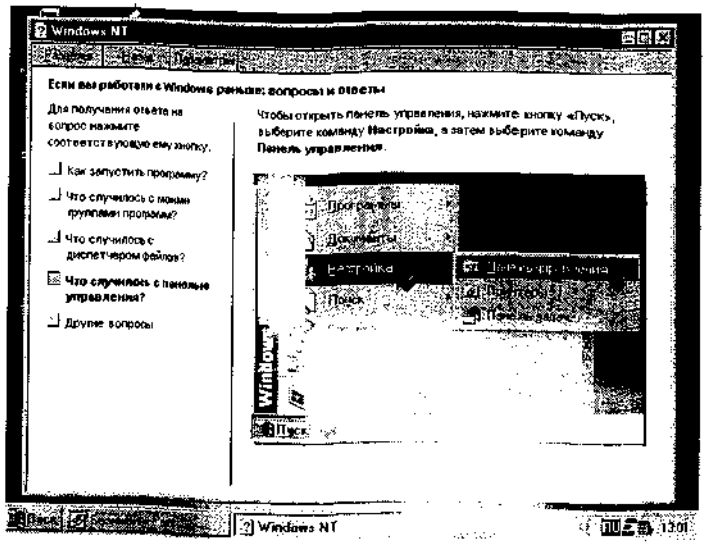



Рис. 2.29. Окно подменю Справка

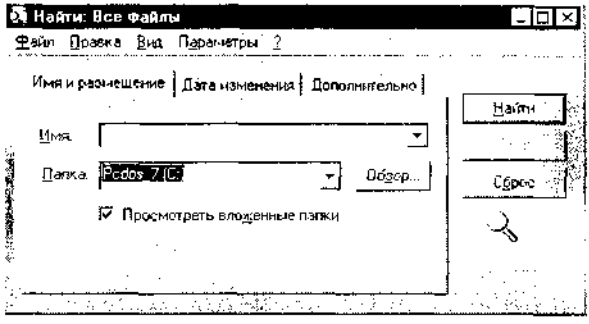



Рис. 2.30. Окно поиска файлов из Главного меню

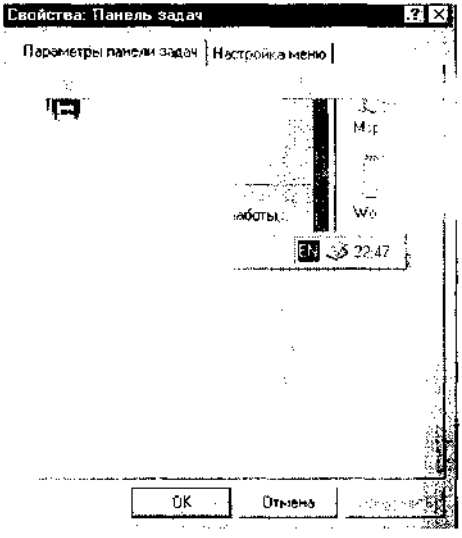



Рис. 2.31. Окно свойства из Настройка панели задач Главного меню

управления файлами и дисками были объединены в программе File Мападег, то теперь их приходится разыскивать по многочисленным меню объектов.

В пакет входит ряд прикладных программ: Internet Information Server 2.0, Index Server, FrontPage, Internet Explorer, Domain Name System (DNS) Server, Proxy Server и Internet Resource Center, все пакеты Service Pack, Plus! и ряд дополнительных утилит, среди которых имеются как новые, например Administrative Wizards или Imager, так и усовершенствованные версии старых программ, например Task Мападег.

Administrative Wizard позволяет автоматизировать типичные задачи, возникающие при управлении сетью, а обновленные версии программ Windows NT Diagnostic, Perfomance Monitor служат для оперативного контроля за состоянием системы. Диалоговое окно Task Manager трансформировалось в мощную программу, которая предоставляет массу полезной информации — от степени загруженности процессора до имен всех активных системных процессоров. При этом пользователь не остается пассивным наблюдателем: при желании с помощью Task Manager возможно, например, завершение любой задачи.

Один из ключевых компонентов Windows NT 4.0 — Internet Information Server 2.0. Это гибкое и многофункциональное решение как для подключения к сети Internet, так и для создания собственной частной сети intranet. От пользователя требуется Только настроить параметры протокола TCP/IP (при установленном сервисе DHCP IP-адрес присваивается автоматически), запустить IIS и создать одну или несколько собственных Web-страниц. После этого Web-документы доступны для всех пользователей вашей сети, у которых установлено ПО, обеспечивающее функционирование протокола TCP/IP и стандартный браузер World-Wide Web.

Появились некоторые изменения в подсистеме дистанционного доступа, Remote Access Service (RAS). 7еперь имеется возможность использовать защищенные каналы связи, новый протокол Point-To-Point Tunneling Protocol (PP7P), возможность использовать несколько модемов для организации каналов связи с удаленными сетями.

Особенности сетевой архитектуры прежних версий Windows N7 (многоуровневая модель защиты от несанкционированного доступа, специфика модульного построения системы и пр.) ограничивали ее пропускную способность при работе в сетях Fast Internet. В версии 4.0 были улучшены алгоритмы кэширования сетевых запросов, оптимизированы модули подсистемы разделения ресурсов, изменен механизм генерации прерываний (при переходе к высокоскоростным сетям эта функция неожиданно стала источником проблем для сетевых ОС). Второе изменение, на которое указывает Microsoft, — увеличенная производительность ОС при выполнении графических операций. Разработчики, которые «переодевали» Windows N7, перенесли часть кода модулей USER и GDI в ядро системы, что позволило ускорить выполнение графических операций на 15—20 %. Однако реальную выгоду от этого улучшения оценить трудно — операции вывода на экран представляют собой лишь малую часть работы, которую выполняют типичные программы для Windows N7. Выводы от более быстрой графики получат преимущественно САПР и

ПО для мультимедиа, но даже в этом случае преимущества далеко не очевидны — быстро выполнив запросы на вывод изображения, операционная система, как правило, отдает освободившееся время процессам с более высокими приоритетами.

Сочетание мощной сетевой ОС и графического интерфейса, созданного для неквалифицированных пользователей, выглядит довольно непривычно. Windows NT 4.0 — это не просто очередная версия популярной операционной системы. Она Представляет собой основу для нового поколения программных продуктов, ориентированных на работу в сети Internet. Возможность создания инфраструктуры intranet, простота в обращении и хорошая репутация прошлых версий Windows NT в сочетании с усиливающейся тенденцией к созданию однородных сетей делают ее привлекательной для пользователей из сферы бизнеса.

Поскольку практически все эти экраны знакомы из содержания предшествующего параграфа, перейдем к рассмотрению архитектурных особенностей ОС.

***Архитектурные модули Windows NT.*** Как показано на рис. 2.32, Windows NT представляет собой модульную (более совершенную, чем монолитная) операционную систему, которая состоит из отдельных взаимосвязанных относительно простых модулей.

Основными модулями Windows NT являются (перечислены в 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), представляющих базовый сервис операционной системы подсистемам среды. Исполняющая система состоит из нескольких компонент,

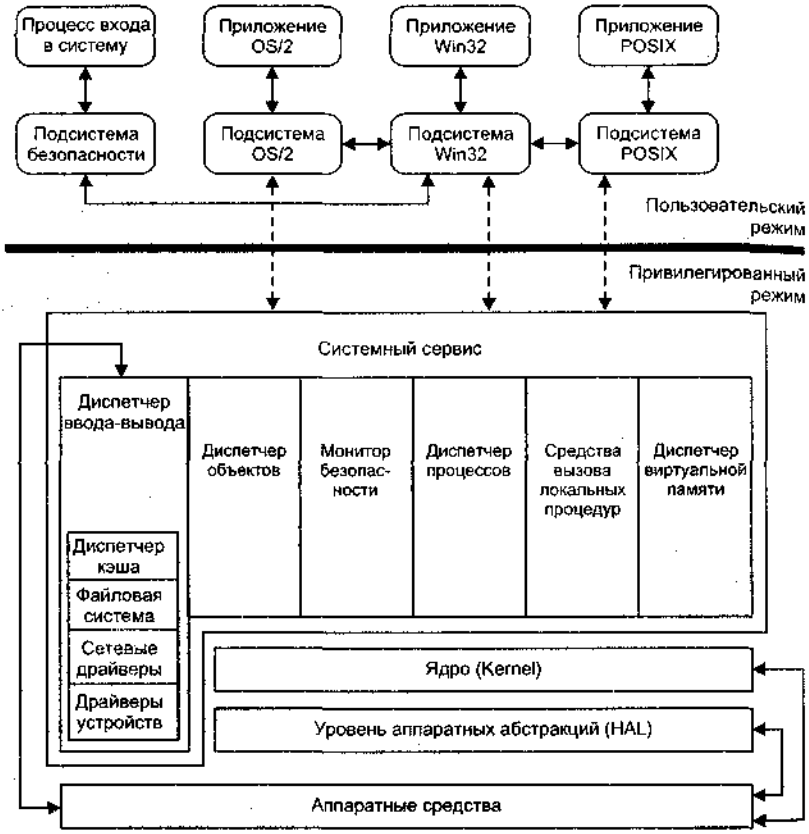



Рис. 2.32. Модульная структура Windows NT

каждая из них предназначена для поддержки определенного системного сервиса. Так, один из компонент — 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.

*Уровень аппаратных абстракций* (HAL) представляет собой создаваемый производителями аппаратных средств слой программно-

го обеспечения, который скрывает (или абстрагирует) особенности и различия аппаратуры от верхних уровней операционной системы. Таким образом, благодаря обеспечиваемому HAL фильтру, различные аппаратные средства выглядят аналогично с точки зрения операционной системы; снимается необходимость специальной настройки операционной системы под используемое оборудование.

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 называются 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.

*Ядро* (Kernel) работает в тесном контакте с уровнем, аппаратных абстракций. Этот модуль в первую очередь занимается планированием действий процессора. В случае, если компьютер содержит несколько процессоров, ядро синхронизирует их работу с целью достижения максимальной производительности системы.

Ядро осуществляет диспетчеризацию *потоков* (threads — нитей Управления, которые иногда называются подзадачами, ответвлениями), которые являются основными объектами в планируемой системе. Потоки определяются в контексте процесса; процесс включает адресное пространство, набор доступных процессу объектов и совокупность выполняемых в контексте процесса потоков управления. Объектами являются управляемые операционной системой ресурсы.

Ядро производит диспетчеризацию потоков управления таким образом, чтобы максимально загрузить процессоры системы и обеспечить первоочередную обработку потоков с более высоким приоритетом. Всего существует 32 значения приоритетов, которые сгруппированы в два класса: 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.

*Объекты диспетчеризации* (dispatcher objects) характеризуются сигнальным состоянием (signaled или nonsignaled) и управляют диспетчеризацией и синхронизацией системных операций. Эти объекты включают события, мутанты, мутэксы, семафоры, потоки управления и таймеры (events, mutants, mutexes, semapnores, threads, timers).

*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 (asynhcronous procedure calls, interupts, power notifies, power statuses, processes, profiles).

*Исполняющая система* (Executive), в состав которой входит 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);
- средства вызова локальных процедур (Local Procedure Call Facility);
- диспетчера ввода-вывода (I/O Manager);
- монитора безопасности (Security Reference Monitor).

Монитор безопасности совместно с процессором входа в систему (Logon) и защищенными подсистемами реализует *модель безопасности Windows NT.*

Верхний уровень исполняющей системы называется системным сервисом (System Services). Показанный на рис. 2.33 системный сервис представляет собой интерфейс между подсистемами среды пользовательского режима и привилегированным режимом.

*Диспетчер кэша.* 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 эффек-

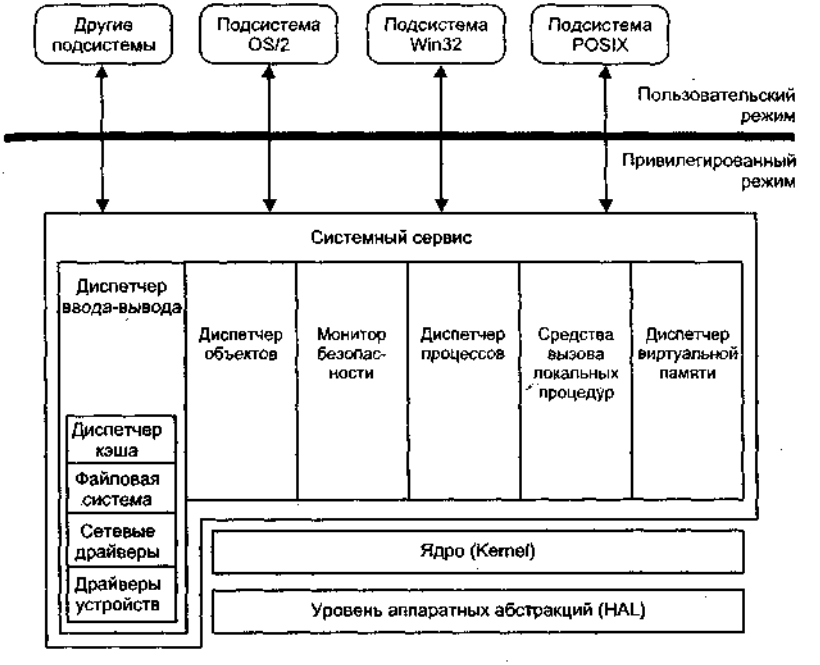



Рис. 2.33. Системный сервис

тивности. 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. Диспетчер кэша обеспечивает службу кэширования для всех файловых систем и сетевых компонентов, функционирующих под управлением диспетчера ввода-вывода. В зависимости от объема доступной оперативной памяти диспетчер кэша может динамически увеличивать или уменьшать размер кэш-памяти. Когда процесс открывает файл, который уже находился в кэше, диспетчер кэша просто копирует данные из кэша в виртуальное адресное пространство.

Диспетчер кэша поддерживает службы типа замедленной записи (lazy write) и замедленной фиксации (lazy commit), которые могут значительно увеличить эффективность файловой системы. В процессе замедленной записи изменения регистрируются в кэше файловой структуры, обеспечивающем более быстрый доступ. Позднее, когда

загрузка центрального процессора снижена, диспетчер кэша заносит изменения на диск. Замедленная фиксация подобна замедленной записи. Вместо немедленной маркировки транзакции, как успешно завершившейся, переданная информация кэшируется и позднее в фоновом режиме записывается в журнал файловой системы.

*Драйверы файловой системы.* В архитектуре ввода-вывода Windows NT управление драйверами файловой системы осуществляет диспетчер ввода-вывода. Windows NT допускает использование множества файловых систем, включая существующие файловые системы типа PAT. Для обеспечения совместимости снизу вверх с операционными системами MS-DOS, Windows 3.x и O8/2 Windows NT поддерживает файловые системы PAT и HPP5.

Кроме того, Windows NT также поддерживает NTFS — новую файловую систему, разработанную специально для использования с Windows NT. NTFS обеспечивает ряд возможностей, включая средства восстановления файловой системы, поддержку Unicode, длинных имен файлов и поддержку для POSIX.

Архитектура ввода-вывода Windows NT не только поддерживает традиционные файловые системы, но и обеспечивает функционирование сетевого редактора и сервера в качестве драйверов файловой системы. С точки зрения диспетчера ввода-вывода, нет разницы между работой с файлом, размещенным на удаленном компьютере сети, и работой с файлом на локальном жестком диске. Редиректоры и серверы могут быть загружены и выгружены динамически так же, как и любые другие драйверы; на одном компьютере может одновременно находиться большое число редиректоров и серверов.

*Сетевые драйверы.* Следующим типом драйверов, присутствующих в качестве компонентов в архитектуре ввода-вывода, являются сетевые драйверы. Windows NT включает интегрированные возможности работы с сетями и поддержку для распределенных приложений. Редиректоры и серверы функционируют как драйверы файловой системы и выполняются на уровне интерфейса поставщика или ниже, где находятся NetBIOS и Windows-сокет.

Драйверы транспортного протокола общаются с редиректорами и серверами через уровень, называемый интерфейсом транспортного драйвера (TDI — Transport Driver Interface). Windows NT включает следующие транспортные средства:

- протокол управления передачей/межсетевой протокол TCP/IP, который обеспечивает возможность работы с широким диапазоном существующих сетей;
- NBF — потомок расширенного интерфейса пользователя NetBIO8 (NetBEUI), который обеспечивает совместимость с су-

шествующими локальными вычислительными сетями на базе LAN Manager, LAN Server и MS-Net;

- управление передачей данных (DLC — Data Link Control), которое обеспечивает интерфейс для доступа к мэйнфрэймам и подключенным к сети принтерам;
- NWLink — реализация IPX/SPX, обеспечивающая связь с Nowell 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 Inerface 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 сетевых адаптеров. В свою очередь, каждая плата сетевого адаптера может поддерживать несколько транспортных протоколов для доступа к различным типам сетевых станций.

*Модель безопасности Windows NT* — представлена монитором безопасности (Security Reference Monitor), а также двумя другими компонентами: процессором входа в систему (Logon Process) и безопасными защищенными подсистемами.

В многозадачной операционной системе, каковой является Windows NT, приложения совместно используют ряд ресурсов системы, включая память компьютера, устройства ввода-вывода, файлы и процессор (ы) системы. Windows NT включает набор компонентов безопасности, которые гарантируют, что приложения не смогут обратиться к этим ресурсам без соответствующего разрешения.

Монитор безопасности отвечает за проведение в жизнь политики проверки правильности доступа и контроля определенной локальной подсистемой безопасности. Монитор безопасности обеспечивает услуги по подтверждению доступа к объектам, проверке привилегий пользователя и генерации сообщений как для привилегированного режима, так и для режима пользователя. Монитор безопасности, подобно другим частям операционной системы, выполняется в привилегированном режиме.

Процесс входа в систему в Windows NT предусматривает обязательный вход в систему безопасности для идентификации пользователя. Каждый пользователь должен иметь бюджет и должен использовать пароль для обращения к этому бюджету.

Прежде чем пользователь сможет обратиться к любому ресурсу компьютера из Windows NT, он должен войти в систему через процесс входа в систему для того, чтобы подсистема безопасности мог-

ла распознать имя пользователя и пароль. Только после успешного установления подлинности монитор безопасности выполняет процедуру проверки правильности доступа для определения права пользователя на обращение к этому объекту.

Защищенность ресурсов — одна из особенностей, предоставляемая моделью безопасности. Задачи не могут обращаться к чужим ресурсам (типа памяти) иначе, чем через применение специальных механизмов совместного использования.

Windows NT также предоставляет средства контроля, которые позволяют администратору фиксировать действия пользователей.

*Управление памятью Windows NT.* Windows NT Workstation 3.51 по существу представляет собой операционную систему сервера, приспособленную для использования на рабочей станции. Этим обусловлена архитектура, в которой абсолютная защита прикладных программ и данных берет верх над соображениями скорости и совместимости. Чрезвычайная надежность Windows NT обеспечивается ценой высоких системных затрат, поэтому для получения приемлемой производительности необходимы быстродействующий ЦП и по меньшей мере 16 Мбайт ОЗУ. В системе Windows NT безопасность нижней памяти достигается за счет отказа от совместимости с драйверами устройств реального режима. В среде Windows NT работают собственные 32-разрядные NT-прикладные программы, а также большинство прикладных программ Windows 95. Так же как к Windows 95, система Windows NT позволяет выполнять в своей среде 16-разрядные Windows- и DOS-программы.

Схема распределения памяти Windows NT отличается от распределения памяти системы Windows 95. Собственным прикладным программам выделяется 2 Гбайт особого адресного пространства, от границы 64 Кбайт до 2 Гбайт (первые 64 Кбайт полностью недоступны). Прикладные программы изолированы друг от друга, хотя могут общаться через буфер обмена Clipboard, механизмы DDE и OLE.

В верхней части каждого 2-Гбайт блока прикладной программы размещен код, воспринимаемый прикладной программой как системные библиотеки DLL кольца 3. На самом деле это просто заглушки, выполняющие перенаправление вызовов, называемые DLL клиентской стороны (client-side DLLs). При вызове большинства функций API из прикладной программы библиотеки DLL клиентской стороны обращаются к локальным процедурам (Local Process Communication — LPC), которые передают вызов и связанные с ним параметры в совершенно изолированное адресное пространство, где содержится собственно системный код. Этот сервер-процесс (server process) проверяет значение параметров, исполняет запрощенную

Архитектура клиент-сервер Windows NT обеспечивает существенно более надежную защиту, чем Windows 95. Но для получения приемлемой производительности требуется мощный ПК с обширной памятью. Windows NT работает с программами, ориентированными на Win16 и Dos, но драйверы устройств реального режима не могут функционировать в среде Windows NT.

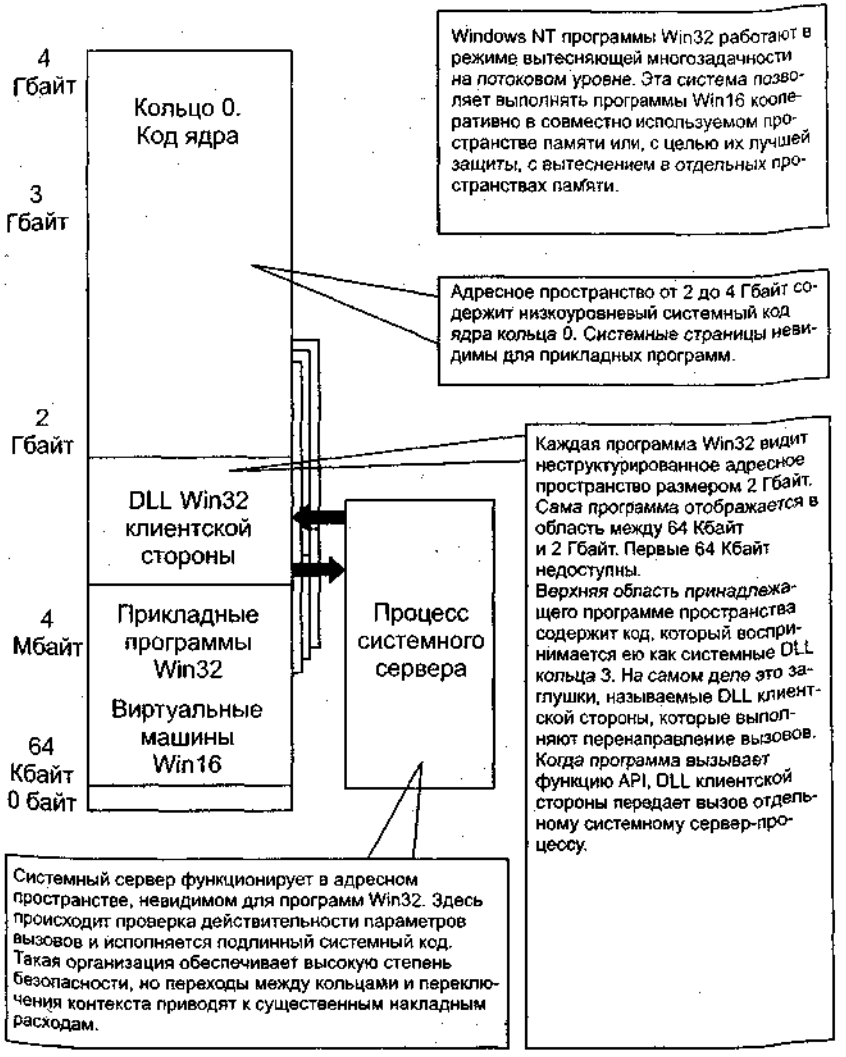



Рис. 2.34. Модель памяти Windows NT

функцию и пересылает результаты назад в адресное пространство прикладной программы. Хотя сервер-процесс сам по себе остается процессом прикладного уровня, он полностью защищен от вызывающей его программы и изолирован от нее.

Между отметками 2 и 4 Гбайт расположены низкоуровневые системные компоненты Windows NT кольца 0, в том числе ядро, планировщик потоков и диспетчер виртуальной памяти. Системные страницы в этой области наделены привилегиями супервизора, которые задаются физическими схемами кольцевой защиты процессора. Это делает низкоуровневый системный код невидимым и недоступным по записи для программ прикладного уровня, но приводит к падению производительности во время переходов между кольцами.

Для 16-разрядных прикладных Windows-программ Windows NT реализует сеансы Windows оп Windows (WOW). Windows NT дает возможность выполнять 16-разрядные программы Windows индивидуально *в* собственных пространствах памяти или совместно в разделяемом адресном пространстве. Почти во всех случаях 16- и 32-разрядные прикладные программы Windows могут свободно взаимодействовать, используя OLE (при необходимости через особые процедуры thunk) независимо от того, выполняются они в отдельной или общей памяти. Собственные прикладные программы и сеансы WOW выполняются в режиме вытесняющей многозадачности, основанной на управлении отдельными потоками. Множественные 16-разрядные прикладные программы Windows в одном сеансе WOW выполняются в соответствии с кооперативной моделью многозадачности. Windows NT может также выполнять в многозадачном режиме несколько сеансов DOS. Поскольку Windows NT имеет полностью 32-разрядную архитектуру, не существует теоретических ограничений на ресурсы GDI и USER.

***Основные отличия Windows2000.*** Windows 2000 или W2k — операционная система (ОС) Microsoft, основанная на технологии Windows NT, что было отражено в первоначальном названии проекта W2k — Windows NT 5.0. Windows 2000 — полностью 32-разрядная ОС с приоритетной многозадачностью и улучшенной реализацией работы с памятью. В основе проекта W2k лежат те же принципы, которые когда-то обеспечили успех NT.

*Интерфейс W2k* подобен интерфейсу Windows 98 с установленным IE 5.0. Однако некоторые детали мы все-таки отметим.

Первое, что бросается в глаза, это то, что изменилась цветовая гамма. Теперь она напоминает одну из схем, используемых в рабочем столе KDE для Linux. Еще одной заметной деталью является тень под курсором мыши, которая снимается/выставляется в Control Pa-

nel → Mouse → Pointers, галочкой на Enable pointer shadow. Кроме этого, добавлен новый эффект при появлении меню, теперь они постепенно проявляются из воздуха. Управляется из Desktop Properties, на закладке Effects галочкой Use transition effects for menu and tooltips.

В Start Мепи введена функция, знакомая по Office 2000, когда при открытии показываются только наиболее часто употребляемые пункты, остальные открываются, если нажать стрелочку вниз. Управлять этим эффектом можно в Taskbar Properties, в закладке General галочкой Use Personalized Мепиз (аналогично, в IE5 данная опция отключается в Tools → Internet Options → Advanced → Enable Personalized Favorites Мепи). В Desktop Properties есть еще несколько пунктов, в том числе и Hide keyboard navigation indicators until I use the АИ key. Если он выбран, то убирается подчеркивание под буквами, которые означают Keyboard shortcut в программах Windows, до тех пор, пока не нажат <Alt>.

На второй закладке ТазКЬаг Properties, Advanced, находится окошко Start Мепи Settings, которое позволяет добавить/удалить строки, входящие в Start Мепи, и расширить некоторые пункты. Например, если отметить галочкой Expand Control Panel, то при наведении курсора мыши на Control Panel в Start Мепи от него вправо откроется еще одно меню, в котором будут все элементы, входящие в нее. Полезной функцией на этой закладке является кнопка Re-sort. W2k, по умолчанию, ставит папки с последними инсталлированными программами в самом низу Start Мепи, папки могут быть даже ниже линков на файлы. Re-sort устраняет эту несправедливость и расставляет все папки сверху вниз по алфавиту. Впрочем, этого же эффекта можно добиться, нажав правую кнопку мыши в Start Мепи → Ргодгатз и выбрав Sort Ьу пате. Кроме этого, правой кнопкой можно «перетащить» (drag and drop) оттуда любые элементы в любое место.

Еще одним отличием, часто подводящим людей, ранее работавших с NT и W9x, как ни странно, является широкое применение Checkbox. Особенно тех, которые представляют из себя просто квадратик на белом фоне. Так что если вы обнаружите, что не можете чего-то сделать, то просмотрите еще раз все окна, возможно, вы просто не обратили внимания на такой Checkbox.

*Task Manager* — это один из самых мощных и удобных инструментов в NT, предназначенных для управления процессами. Вызывается он либо <Ctrl+Shift+Esc>, либо выбором в меню, появляющимся после нажатия правой кнопкой на Taskbar. Можно его выбрать и после <Ctrl+Alt+Del>.

Task тападег состоит из трех закладок — Performance, Procesеs, Applications. Начнем с Performance. На этой закладке показывается информация о загрузке процессора(ов) в реальном времени. показывается загрузка физической памяти, причем показано, сколько занято/свободно оперативной памяти и сколько занято системного Swap'a. Кроме этого, там же дается другая дополнительная информация, например Threads и Processes — количество нитей и процессов, исполняемых сейчас на машине, Peak — пиковый размер Swap'a в течение сессии, Nonpaged — количество памяти, отведенное под ядро. Эта информация может использоваться, когда надо будет ответить на вопрос, какой фактор в системе является «бутылочным горлышком», замедляющим работу (хотя для этих целей лучше использовать Performance Monitor).

Вторая закладка, Processes, содержит список процессов, активных в данный момент. Для каждого процесса можно узнать некоторую дополнительную информацию, как то: PID (Process ID), количество используемой оперативной памяти, количество нитей, сгенерированных процессом, и многое другое. Добавить/удалить показываемые параметры можно через View → Select Columns. Кроме этого, с любым из этих процессов можно произвести вполне определенные действия. Для этого надо просто нажать на нем правой кнопкой мыши, появится контекстное меню, через которое можно закончить процесс, End Процесз, можно «убить» сам процесс и все остальные, которые он «породил», End Process Tree. Можно выставить приоритет процессу, от высшего RealTime до самого низкого, Low. Если на машине установлено два процессора и многопроцессорное ядро, то в этом меню появляется еще один пункт, Set Affinity, который позволяет перевести процесс на другой процессор, Cpи 0, Cpu1, и так далее до Cpu31.

Последняя закладка Task Мападег — Applications, позволяет просмотреть список работающих приложений и завершить любое из них. Task Мападег позволяет не только завершать приложения, он может также запускать новые приложения. File → New task (Run).

*Active Directory* — это новое средство управления пользователями и сетевыми ресурсами. Оно призвано облегчить работу администраторам больших сетей на базе W2k и вокруг него строится вся система управления сетью и ее безопасности. Для установки Active Directory необходимо иметь W2k Server. W2kPro может работать в среде Active Directory, но не может создавать ее. Active Directory строится на следующих принципах:

1. Единая регистрация в сети. Благодаря технологии IntelliMirror можно подойти к любому компьютеру в офисе, ввести свой пароль

и перед вами будет ваш рабочий стол, ваши документы и ваши настройки.

2. Безопасность информации. В службу Active Directory встроены средства идентификации пользователя. Для каждого объекта в сети можно централизованно выставлять права доступа, в зависимости от групп и конкретных пользователей. Благодаря системе безопасности Kerberos можно осуществлять защищенную связь даже по открытым сетям, таким, как Интернет. При этом данные, передаваемые по сети, шифруются, а пароли не передаются и не хранятся на клиентских машинах. Система безопасности Kerberos (называется по имени мифического трехголового пса, который, согласно греческой мифологии, охранял адские врата) известна довольно давно, но в ОС от Microsoft она используется впервые. Если не вдаваться в подробности, то работает эта система так:

- клиент посылает запрос серверу аутентификации на разрешения доступа к нужной информации;
- сервер проверяет права клиента и отсылает ему разрешение на получение требуемой информации, зашифрованное с помощью известного клиенту ключа, и заодно отсылает временный ключ шифрования. С помощью этого ключа шифруется вся передаваемая информация, причем время жизни ключа ограничено, поэтому сервер аутентификации время от времени присылает новый ключ (естественно, новый ключ зашифрован с помощью текущего ключа), который неизвестен никому, кроме сервера и клиента. Регулярная смена ключей шифрования сильно затрудняет жизнь злоумышленникам, охотящимся за вашими данными.

Однако, как мы все помним, в греческом мифе Kerberos не смог противостоять могучему Гераклу. Так и в нашем случае, несмотря на все свои преимущества, система безопасности Kerberos не может противостоять всем видам атак. Например, можно засыпать приложение ложными запросами, так называемая атака «Deny of service», что может привести к тому, что приложение не будет использовать протокол Kerberos.

3. Централизованное управление. При использовании службы Active Directory у администратора отпадает необходимость вручную конфигурировать каждую машину, если, к примеру, необходимо поменять права доступа к какому-либо одному объекту или установить новый сетевой принтер. Такие изменения можно производить сразу Для всей сети.

4. Гибкий интерфейс. Структуры каталогов меняются быстро и легко. Например, можно создать каталог своей фирмы, выделить в

отдельные подкаталоги бухгалтерию, отделы маркетинга, секретариат и представить все это в виде древовидной структуры. Или, например, создать несколько деревьев, представляющих различные офисы в разных зданиях или регионах и с легкостью задать связь и права доступа между ними. Подключить сетевой принтер к директории бухгалтеров одним движением мыши. (При этом драйверы установятся на эти компьютеры автоматически.) Или мышью перетащить весь бухгалтерский отдел с одного сервера на другой, со всеми их правами, папками и документами.

5. Интеграция с DNS. Благодаря тесной интеграции с DNS в Active Directory в локальной сети используются те же имена ресурсов, что и в Интернет, что приводит к меньшей путанице и способствует более тесному взаимодействию локальной и глобальной сети.

6. Масштабируемость. Несколько доменов Active Directory могут быть объединены вместе под одним управлением.

7. Простота поиска. В домене Active Directory различные объекты можно находить по самым различным признакам, таким, как имя пользователя или компьютера, адрес электронной почты пользователя и т. д.

*DFS (Distributed File System)* — один из инструментов Active Directory. Он позволяет создавать сетевые ресурсы, в которые может входить множество файловых систем на различных машинах. Для пользователя Active Directory это абсолютно прозрачно и не имеет никакого значения, где и на каких машинах физически расположены те файлы, с которыми он работает, — для него они все расположены в одном месте. Кроме этого, при использовании DFS и Active Directory упрощается управление такими ресурсами. Оно централизовано, можно просто и безболезненно добавлять новые ресурсы или удалять старые, менять физическое месторасположение файлов, входящих в DFS, и т. д.

*Файловые системы NTFS4 (Windows NT) и NTFS5 (Windows 2000).* NTFS выросла из файловой системы HPFS, разрабатываемой совместно IBM и Microsoft для проекта OS/2. Она начала использоваться вместе с Windows NT 3.1 в 1993 году. Windows NT 3.1 должна была составить конкуренцию серверам на базе NetWare и Unix, поэтому NTFS вобрала в себя все технологические достижения того времени. Вот основные из них:

1. *Работа с большими дисками.* NTFS имеет размер кластера 512 байт, что в принципе оптимально, но его можно менять до 64 Кбайт. Более важно то, что NTFS способна теоретически работать с томами размером в 16,777,216 терабайт. Теоретически, потому что таких жестких дисков пока не существует.

2. *Устойчивость.* NTFS содержит две копии аналога PAT, которые называются MFT (Master File Table). В отличие от PAT MSDOS, MPT больше напоминает таблицу базы данных. Если оригинал MPT поврежден в случае аппаратной ошибки (например, появления bad-сектора), то система при следующей загрузке использует копию MPT и автоматически создает новый оригинал, уже с учетом повреждений. Но это не самое главное. Главное, что NTFS использует систему транзакций при записи файлов на диск. Эта система пришла из СУБД, где зашита целостности данных — жизненно важное дело. Уже это говорит о ее эффективности. В упрощенном виде она работает так:

- драйвер ввода/вывода NTFS инициирует процесс записи, одновременно сообщая сервису Log РПе Service вести регистрацию всего происходящего;
- данные пишутся в кэш, под управлением сервиса CacBe Manager;
- Cache Manager посылает данные Virtual Memory Manager (менеджеру виртуальной памяти) для записи на диск в фоновом режиме;
- Virtual Метогу Manager посылает данные драйверу диска, пропустив их через Fault Tolerant Driver;
- драйвер диска шлет их контроллеру, который уже пишет их либо в кэш, либо прямо на диск;
- если эта операция проходит без ошибок; запись регистрации удаляется;
- если происходит сбой, запись остается в таблице транзакций, и при следующем доступе к диску Log РПе Service обнаруживает эту запись и просто восстанавливает все, как было до этой операции.

Такая система гарантирует абсолютную сохранность данных в случае копирования, перемещения и удаления файлов или директорий. При внесении изменений в файл вы теряете те изменения, которые находились в момент сбоя в памяти или в кэше контроллера, и не успели записаться на диск.

3. *Защищенность.* NTFS рассматривает файлы как объекты. Каждый файловый объект обладает свойствами, такими, как его имя, дата создания, дата последнего обновления, архивный статус и Дескриптор безопасности. Файловый объект также содержит набор Методов, которые позволяют с ним работать, такие, как open, close, read и write. Пользователи, включая сетевые, для обращения к файлу вызывают эти методы, а Security Reference Monitor определяет, имеет ли пользователь необходимые права для вызова какого-либо из этих методов. Кроме этого, файлы можно шифровать.

4. *Компрессия данных.* NTFS позволяет сжимать отдельные каталоги и файлы, в отличие от DriveSpace, который позволял сжимать только диски целиком. Это очень удобно, для экономии пространства на диске, например, можно сжимать «на лету» большие графические файлы формата BMP или текстовые файлы, причем для пользователя все это будет прозрачно.

5. *Поддержка формата 150 Unicode.* Формат Unicode использует 16bit для кодировки каждого символа, в отличие от ASCII, который использовал 8bit или еще хуже — 7bit. Для простого пользователя это означает то, что теперь он может называть файлы на любом языке, хоть на китайском — система это будет поддерживать, не требуя изменить кодовую страницу, как это делал DOS и W9x.

. *Файловая система NTFS5.* Основное, в чем NT4 уступала NetWare, это отсутствие квотирования (или ограничение максимального объема дискового пространства для пользователя, которое он сможет использовать). При этом вовсе не обязательно, чтобы все файлы пользователя хранились в одном месте, они вполне могут быть разбросаны по всем дискам. Устанавливаются квоты через Properties NTFS раздела, закладка Quota. Через Quota Entries... можно выставлять квоты для каждого отдельного пользователя.

Второе, достаточно важное отличие NTFS5 от старой версии — возможность поиска файла по имени его владельца. Если нужно найти все файлы, созданные в Word'е каким-то одним пользователем на диске, где этих файлов тысячи. В NT4 это было проблемой. С помощью Access Control List (Список управления доступом) можно легко проверить, какие файлы доступны пользователю, и установить права доступа к отдельным файлам или каталогам.

Кроме непосредственного изменения самой структуры NTFS, в W2k добавлен Microsoft Index Server, который значительно ускоряет поиск файлов, особенно по их содержимому, за счет индексации содержимого дисков. Управляется эта служба через раздел Indexing Service окна Computer Management. В этом разделе можно просмотреть, какие директории индексируются, и, при желании, добавить новые или удалить старые. Это возможно и для разделов других файловых систем, а не только NTFS.

В NTFS5 добавлена функция точки монтирования или, по-другому, точки соединения (junction point). Функция эта знакома пользователям различных версий Unix/Linux, но в продуктах Microsoft она появилась впервые. С помощью этой технологии можно присоединить любой дисковый ресурс в любое место файловой системы. Например, можно присоединить жесткий диск D:\ в любой из каталогов на диске C:\, например в C:\games. 7еперь зайдя в директо-

рию C:\games, можно будет видеть содержимое корневого каталога диска D:\. Все изменения, которые будут произведены в этой директории, будут произведены на диске D:\. После этого можно в окне Computer management → Disk Management убрать букву, присвоенную этому диску (Change disk letter and path), и пользователь даже не будет знать, что на компьютере установлено два диска! Он будет работать с одним диском C:\ и директория C:\games для него ничем не будет отличаться от других. Смонтировать диск или раздел в директорию на NTFS разделе или диске можно из уже знакомого нам меню Change disk letter and path выбором пункта Add.. → Mount in this NTFS folder → Browse. Управлять этой функцией можно и через командную строку командой mountvol.

*Динамические диски (Dynamic Disk).* Это физический диск, на котором могут быть созданы динамические разделы. Такой диск может быть доступен только из W2k. Динамические разделы могут быть следующих видов:

1. Простые (simple). Простые разделы практически ничем не отличаются от тех, к которым мы привыкли.

2. Составные (spanned). Состоят из нескольких динамических дисков, которые представлены как один диск. Данные пишутся и читаются последовательно.

3. Чередующиеся (stripped). Несколько динамических дисков, которые представлены как один диск. Данные пишутся и читаются одновременно на несколько дисков. Это теоретически должно обеспечивать вдвое большую скорость на дисковых операциях. На практике прирост хотя и значительный, но меньше, чем в два раза. Мы бы рекомендовали использовать этот режим только в том случае, если уже имеются два диска. В противном случае, гораздо выгоднее купить один винчестер вдвое большего объема, с отличными скоростными характеристиками (например, IBM DJNA Janus или IBM DPTA Pilo), чем два маленьких и более медленных, в расчете на то, что они будут много быстрее. Конечно, если взять два IBM и объединить их в Stripped Volume, то они будут быстрее, чем один. Однако системный раздел не может быть Stripped Volume. В этом случае разумнее приобрести аппаратный IDE-RAID контроллер, например Promise FastTrack66, который обеспечивает возможность работы с RAID даже из-под ДОС; таким образом, можно сделать системный раздел чередующимся.

4. Зеркальные (mirrored). Эти разделы состоят из двух физических дисков. Данные, записываемые на один из дисков, автоматически дублируются на другом. Это не дает никаких преимуществ в плане скорости, но зато обеспечивает вдвое большую степень надежностисохранностиданных.

5. RAID5. Состоит из трех или более дисков. Представляет из себя stripped volume с контролем ошибок. То есть данные пишутся на два диска, в два блока, а на третий диск и в третий блок записывается ECC, код коррекции ошибок, с помощью которого по информации любого из блоков можно восстановить содержимое второго блока. Причем код ECC записывается попеременно, на каждый из входящих в массив дисков. Эта технология позволяет более экономно использовать дисковое пространство, чем mirrored volumes, но работает медленнее. Любой из этих разделов может быть отформатирован как под PAT32, так и под NTFS. Управление Dynamic disk осуществляется через раздел Disk Management окна Compu1er Management.

***Примечание:*** *все эти разделы, кроме simple, можно создавать только на динамических дисках.*

Обычный диск может быть конвертирован в динамический из окна Disk Management, однако обратный процесс (конвертировать динамический диск в простой) не всегда возможен. Например, если диск с самого начала создавался как динамический, то на нем отсутствует привычная таблица разделов, и чтобы создать ее, его придется заново разбивать с помощью fdisk и форматировать. Кроме этого, если удалить на динамическом диске несколько разделов, то свободное место не объединяется и новый раздел, равный по размеру удаленным, будет состоять из нескольких мелких разделов, объединенных в volume set под одной буквой.

## Вопросы и упражнения к главе 2

1. **Что такое дисковая операционная система?**
2. **Какие операционные системы вам известны?**
3. **Какие версии DOS вам известны?**
4. **Какие компоненты входят в состав DOS? Что такое ядро DOS?**
5. **Что такое командный процессор и какие функции он выполняет?**
6. **Что такое драйвер?**
7. **Как происходит начальная загрузка DOS?**
8. **Что представляет из себя файловая система MS-DOS?**
9. **Какие существуют правила для задания на диске имени файла и каталога? Что такое задание файла по маске?**
10. **Как строится полное имя файла и каталога?**
11. **Какие функции выполняют файлы autoexec.bat и config.sys и какова их типичная структура?**
12. **Какие основные внутренние команды DOS вы знаете?**

13. Какие вам известны основные внешние команды DOS?
14. Как создать каталог и файл в MS-DOS?
15. Что такое Windows и какие бывают версии Windows?
16. Какие преимущества имеет Windows?
17. Какие требования предъявляются к компьютеру при установке на нем Windows?
18. Какие функции выполняет Windows? Как запускаются программы в Windows?
19. Как производится запуск Windows и выход из Windows и какие бывают режимы работы у Windows?
20. Как изменить размеры и положение окна в Windows?
21. Что такое Диспетчер Программ Windows и какие функции он выполняет? Меню Диспетчера Программ.
22. Что такое Диспетчер Файлов Windows и какие функции он выполняет? Меню Диспетчера Файлов.
23. Как создать программную группу и программный элемент Windows?
24. Какие основные приложения Windows вы знаете? В какой программной группе они находятся и как их запустить?
25. Что такое Панель Управления Windows? Где она расположена и какие функции выполняет?
26. Что такое pif-файл и как производится его создание и редактирование?
27. Что такое пиктограмма Windows? Как она создается? Укажите библиотеки пиктограмм в Windows.
28. Как упорядочить пиктограммы в Windows?
29. Как переключиться от одного приложения к другому, затем завершить задачу в Диспетчере Программ Windows?
30. Что такое буфер обмена Windows?
31. Что такое ОС Windows 95? Какие функции она выполняет?
32. Как запустить оболочку Windows 95 и как из нее выйти?
33. Какие бывают версии Windows 95? Какие преимущества и недостатки имеет ОС Windows 95? Какие требования предъявляются к компьютеру при установке Windows 95?
34. Что такое Рабочий СТОЛ и какие его элементы вы знаете?
35. Что такое Панель Задач и Панель управления Windows 95? Каково их назначение? Что такое Проводник и каковы его функции?
36. Что такое Папки и Ярлыки и как они создаются и удаляются?
37. Что такое окна Windows 95 и каковы их основные элементы?
38. В чем состоит документо-ориентированный принцип Windows 95?
39. Каковы особенности файловой системы в Windows 95?
40. Какие важнейшие приложения содержит Windows 95?
41. Какие программы можно использовать для обслуживания диска и восстановления файловой системы в Windows 95, а какие нет?
42. Какие программы позволяют работать в Windows 95 так же, как в Windows 3.1?

## Задания

1. Определите версию DOS на своей машине.
2. Проверьте, правильно ли установлены дата и время на вашей машине. При необходимости произведите их коррекцию.
3. Создайте каталог PRIMER на диске C: и создайте в нем файл proba.txt. Затем удалите этот файл и каталог.
4. Вставьте дискету, выведите ее оглавление и затем выполните предыдущее задание.
5. Прочитайте и проанализируйте содержимое файлов autoexec.bat и config.sys. Скопируйте файл config.sys с жесткого диска на дискету.
6. Выйдите из Norton Commander и выведите оглавление диска C:, затем перейдите в каталог, где находится Norton Commander, и запустите его.
7. Проверьте свой диск программой chkdsk на наличие потерянных кластеров. Затем запустите программу ScanDisk и проверьте свой винчестер.
8. Выведите карту ОЗУ вашего компьютера и проанализируйте ее. Затем введите команду очистки экрана.
9. Проведите диагностику своей машины программой MSD.
10. Создайте тестовую программную группу TES_GROUP и в ней тестовый программный элемент Windows. Удалите тестовый программный элемент и программную группу.
11. Измените описание программной группы и программного элемента.
12. Измените вид пиктограммы Windows. Создайте свою пиктограмму в редакторе IconEdit.
13. Измените положение и размеры окна в Windows. Увеличьте окно на весь экран и сверните его обратно в пиктограмму.
14. Запустите приложение Windows и вернитесь обратно в Windows.
15. Откройте одновременно несколько окон в Windows и скопируйте программный элемент из одного окна в другое, затем переместите программный элемент из одного окна в другое.
16. Создайте PIF-файл в редакторе PIF-Editor для любой программы.
17. Запустите программу с помощью PIF-файла.
18. Создайте тестовый каталог на диске C: в диспетчере файлов Windows. Скопируйте в этот каталог группу файлов. Удалите эти пробные файлы, а затем и пробный каталог.
19. Свяжите в Диспетчере Файлов текстовый файл с соответствующим текстовым редактором, графический файл — с графическим редактором.
20. Измените цвета и оформление в Windows, установите дату и время в панели управления. Установите хранитель экрана и время задержки его появления в Windows.

21. Проверьте переключение клавиатуры РУС/ЛАТ. Проверьте, правильно ли установлены стандарты и драйвер принтера в панели управления Windows.
22. Запустите справочную систему Windows, найдите гипертекстовые ссылки и воспользуйтесь ими. Запустите учебник Windows. Выйдите из учебника и из справочной системы Windows.
73 Запустите Windows 95 и осуществите выход из оболочки Windows 95.
24 Создайте ярлыки основных программ (Word, Excel, Works) и положите их на Рабочий стол. Проверьте, запускаются ли эти программы с помощью ярлыков. Измените внешний вид и название ярлыка.
25 Создайте Папку, измените ее название. Включите в Папку документы.
26. Поместите основные программы (Word, Excel, Works) в меню кнопки Пуск. Запустите эти программы. Произведите поиск файла и папки на диске С: с помощью меню кнопки Пуск.
27. Используйте Корзину для удаления пробных Ярлыка и Папки. Восстановите из Корзины удаленные объекты. Очистите Корзину.
28. Запустите Панель управления и измените параметры оформления Рабочего стола.
29. Запустите основные программы на исполнение с помощью инструмента Мой компьютер. Скопируйте файлы и папки на дискету.
30. Запустите основные программы на исполнение с помощью Проводника.
31. Откройте окно, измените его размеры и место положения. Сверните и разверните окно, закройте окно. Перетащите пробный документ из окна на Рабочий СТОЛ. Измените вид содержимого окна.
32. Введите прикладные программы Disk Commander и Windows Соттапder в пункт Меню Приложение кнопки Пуск в Панели Задач.
33. Определите параметры файла в инструменте Мой компьютер и с помощью Проводника.
34. Произведите поиск заданного файла или папки на диске (дискете) с помощью меню кнопки Пуск.

# Глава 3. Операционные системы коллективного пользования — многопользовательские многозадачные

Многопользовательские многозадачные ОС в связи с необходимостью обеспечения мультипрограммирования и обеспечения многопользовательского режима обработки данных впервые были разработаны для *больших ЭВМ (mainframes).*

Первая функционально полноценная ОС — О8/360 была предложена фирмой IBM в качестве оболочки ЭВМ IBM/360. Разработка и внедрение ОС позволили разграничить функции операторов, администраторов, программистов, пользователей, а также существенно (в десятки и сотни раз) повысить производительность ЭВМ и степень загрузки технических средств. Версии О5/360/370/375 — MPT (мультипрограммирование с фиксированным количеством задач), MVT (с переменным количеством задач), SVS (система с виртуальной памятью), SVM (система виртуальных машин) — последовательно сменяли друг друга и во многом определили современные представления о роли ОС в общей иерархии *систем управления данными и задачами при обработке данных на ЭВМ.*

Ранние версии О8/360 были ориентированы на пакетную (batch processing) обработку информации — входной поток заданий (на МЛ, МД или перфокартах) подготавливался заранее и поступал на обработку в непрерывном режиме. В дальнейшем возникли расширения О8/360/375, допускающие диалоговую обработку данных с терминалов пользователя, последняя из версий (О8 SVM) фактически предоставляла в распоряжение пользователя «виртуальную персональную ЭВМ» с полной мощностью вычислительной установки IBM/360/375. ОС других семейств (поколений), например RSX (для PDP/11 DEC) или Unix, с самого начала ориентировались на интерактивное взаимодействие с пользователями.

Относительно Unix следует отметить, что она в настоящее время является самой популярной ОС (если судить *не по количеству, а по качеству ЭВМ,* на которых она установлена, а также учесть, что М8-

DOS функционально является подмножеством Unix). Unix-подобные системы известны для:

- больших машин (Digital/PDP, Digital/VAX, Hewlett-Packard 4000/9000, Honeywell 6070, IBM/370, Amdahl 470);
- персональных ЭВМ и рабочих станций (SUN SparcStation, IBM PowerStation, DECStation, систем на процессорах Zilog/Z8000, систем на процессорах Motorola/68000/68040, в том числе Apple Macintosh, систем на процессорах Intel/80286/386/486/Pentium).

Наиболее распространенные версии Unix:

- BellLabs AT&T — версииSystem 6, 7, III, V;
- FreeBSD (версии 4.2, 4.3) — реализация UNIX для персональных компьютеров, выполненная исследовательской группой вычислительных систем Калифорнийского университета в Беркли. Версии, выпускаемые этой группой, обозначаются BSD (от Berkeley Software Distribution);
- фирменные версии — SCO (Santa Cruz Operation), ISC (Interactive 8узlет Corporation), A/OX, SunOs, AIX, Xenix, Ultrics, HP-UX и пр.

Широко распространены также системы, разработанные под влиянием концепций UNIX, — MS-DOS, Windows 95/NT, O8/2.

В настоящей главе будут кратко рассмотрены операционные системы семейства O8/360, RSX, UNIX, LINUX.

## 3.1. Операционные системы O5/360/370/375

***Вычислительные машины ряда ЭВМ IBM/360(ЕС ЭВМ).*** Вычислительные машины ряда ЭВМ IBM/360 (в СССР была разработана аналогичная серия — ЕС ЭВМ) представляли собой семейство программно-совместимых машин третьего поколения. Архитектурно машины включают:

- главный процессор (CPU — Central Processing Unit);
- оперативную память (Мат storage, Core storage);
- каналы устройств, обеспечивающие операции обмена данными между памятью и внешними устройствами независимо от процессора (channel proce88or);
- набор внешних устройств ввода-вывода (devices), выполняющих обмен информацией между внешними носителями и каналами.

Для ряда ЭВМ IBM/360 (ЕС ЭВМ) характерно наличие каналов — специализированных процессоров, позволяющих освободить процессор от выполнения операций ввода-вывода и тем самым повысить скорость обмена с внешними устройствами. В машинах ряда ЭВМ IBM/360 (ЕС ЭВМ) с помощью каналов обеспечивается параллельная работа процессора и внешних устройств, а также параллельное выполнение операций ввода-вывода с несколькими внешними устройствами.

Система программного обеспечения ЕС ЭВМ включала в себя операционные системы, комплексы программ технического обслуживания, пакеты прикладных программ. Эта система открытая, т. е. ее состав может пополняться, обеспечивая развитие технических средств, совершенствование методов обработки информации, расширение сфер применения.

***Основные сведения о функционировании ОС.*** Главными функциями ОС являются: управление задачами и управление данными. Эти функции реализуются через формализованное средство описания данных и заданий — язык управления заданиями (JCL).

На рис. 3.1 схематически изображен порядок прохождения заданий в ОС ЕС. Первоначально задания, представляющие собой некоторые тексты (см. ниже), считываются и обрабатываются программой системного ввода, осуществляющей интерпретацию операторов JCL. При отсутствии ошибок очередное задание помещается во входную очередь, ожидая освобождения требуемой области оперативной памяти ЭВМ и других ресурсов. При наличии требуемых ресурсов программа инициатор-терминатор выделяет раздел памяти, размещает первую из программ задания, присоединяет необходимые входные-выходные наборы данных и передает ей управление. После завершения работы программы, при отсутствии других пунктов задания, осуществляется завершение *задания,* уничтожение временных наборов данных, закрытие выходных наборов, вывод информации на внешние носители, освобождение ресурсов и устройств ЭВМ.

Результирующие сообщения задания помещаются в выходную очередь на системную печать.

***Язык управления заданиями (JCL)*** ОС ЕС включает следующие основные операторы, приведенные в табл. 3.1.

Задание представляет собой совокупность операторов, помещенных между заголовком задания («JOB-карты») и пустым оператором (или следующей ГОВ-картой), и должно содержать хотя бы один оператор шага задания (EXEC-карту), соответствующий некоторой выполняемой в данном шаге программе. Каждый набор дан-

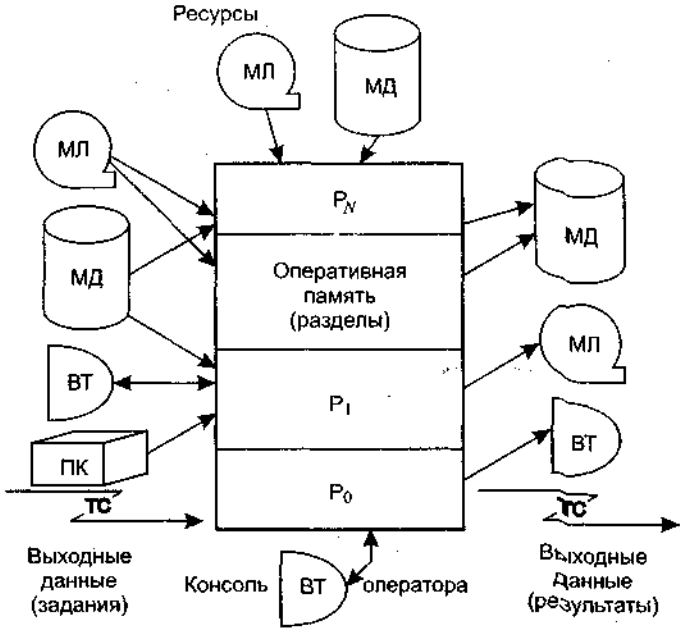



Рис. 3.1. Общая структура прохождения задач в ЭВМ: МД — накопители на магнитных дисках; МЛ — накопители на магнитных лентах; ТС — телекоммуникационный канал; ВТ — видеотерминалы; ПК — перфокарты; $P_i$ — разделы (сегменты оперативной памяти) ($i = 0, 1, ..., N$)

*Таблица 3.1.* **Основные операторы языка управления заданиями (JCL-операторы, карты)**

| Название оператора | Формат оператора |
|---|---|
| Заголовок задания | //ИМЯ JOB [ОПЕРАНДЫ][КОММЕНТАРИИ] |
| Заголовок шага задания | //[ИМЯ] EXEC [ОПЕРАНДЫ] |
| Определения (описания) данных | //[ИМЯ] DD [ОПЕРАНДЫ] |
| Пустой оператор | // |
| Комментарий | //*[ТЕКСТ] |
| Примечание: в квадратные скобки операторов включены необязательные компоненты. | |

ных (НД), вводимый или выводимый программой шага задания, задается картой описания данного (DD-оператором).

Для описания конструкций JCL-операторов используется *нотация IBM,* которая включает следующие конструкции:

- < > *угловые скобки* (или двойные кавычки""), обозначающие элементы программы, определяемые пользователем (<идентификатор>, <список параметров>, <условие> и пр. В соответствующих местах реальной программы будет находиться идентификатор переменной и т. д.);
- [] *квадратные скобки,* ограничивающие синтаксическую конструкцию, обозначают ее возможное отсутствие;
- { } *фигурные скобки,* содержащие вертикально расположенный список, означают обязательный элемент, одно из значений которого должно быть выбрано из этого списка;
- ... *горизонтальное многоточие*, следующее после некоторой синтаксической конструкции, обозначает последовательность конструкций той же самой формы, что и предшествующая многоточию конструкция. Например, ={<выражение> [,<выражение>]...} обозначает, что одно или более выражений, разделенных запятыми, может появиться между фигурными скобками;
- *вертикальное многоточие* означает пропуск некоторой части программы, представленной в примере.

Каждый оператор DD имеет имя (ddname), используемое для ссылок на этот оператор в программе и других целей. Иногда группа DD-операторов может иметь одно имя, находящееся в карте первого из операторов, а в остальных операторах группы поле имени не заполнено. Такие наборы данных называются сцепленными, рассматриваются ОС как один набор данных и должны иметь одинаковую организацию, формат и длину записей (см.ниже).

Имеется ряд стандартных и/или зарезервированных в системе dd-имен:

JOBLIB — библиотека задания; оператор размещается непосредственно после карты JOB и описывает одну или несколько (сцепленных) библиотек исполнительных программных модулей, один или несколько из которых вызываются в задании;

STEPLIB — библиотека шага задания. Оператор аналогичен JOBLIB, но действие его распространяется только на данный шаг;

SYSUDUMP — НД, в который при аварийном завершении выводится дамп памяти — содержимое основной памяти, в которой выполняется задание, а также ряд дополнительных данных о состоянии системы;

SYSIN — системный вход; содержит исходную программу транслятора, исходные данные пользовательской программы, управ-

ляющие карты утилит; весьма часто размещается во входном потоке (см.ниже);

SYSPRINT — системный вывод; содержит сообщения трансляторов, диагностические сообщения утилит и служебных программ; как правило, помещается в очередь, направленную на выходной принтер;

SYSUT1, SYSUT2,.. SYSUTN— рабочие НД трансляторов, утилит, редакторов, входные/выходные наборы данных утилит.

*Основные параметры операторов управления данными.* DD-операторы, как и другие JCL-операторы, могут содержать *ключевые* и *позиционные* параметры (подпараметры). Ключевые имеют вид: ИМЯ = ЗНАЧЕНИЕ, где ИМЯ — зарезервированное наименование параметра, а ЗНАЧЕНИЕ — выбираемое из зарезервированных (или произвольно), задаваемое программистом значение параметра.

Порядок ключевых параметров в операторах и их разрешенное отсутствие не играют роли. Позиционные параметры имен не имеют и их смысл определяется позицией в конструкции JCL. Они, как правило, разделяются запятыми и отсутствие параметра требует наличия соответствующего разделителя. Наиболее часто употребляются параметры DSNAME (DSN), DISP, UNIT, VOLUME (VOL), 8PACE, LABEL, DCB. Ниже приводятся краткие описания форматов и основных подпараметров данных конструкций.

*Параметр DSNAME*— имя НД и его вид (постоянный или временный).

Параметр задает имя набора, по которому осуществляется его поиск на устройствах ЭВМ или с которым он создается. Для временных наборов данных имя может не указываться, либо должно начинаться с символов &&.

*Параметр DISP* — текущее и будущее состояние набора данных, имеет следующую структуру. В то время, как сам параметр DISP является ключевым, подпараметры, указанные в скобках, — позиционные:

$$\text{DISP} = (\left\{\begin{matrix}\text{NEW}\\ \text{OLD}\\ \text{SHR}\end{matrix}\right\}, [\left\{\begin{matrix}\text{KEEP}\\ \text{DELETE}\\ \text{PASS}\end{matrix}\right\} [, \left\{\begin{matrix}\text{KEEP}\\ \text{DELETE}\\ \text{PASS}\end{matrix}\right\}]]).$$

Первый подпараметр — состояние, может иметь значения:

NEW — набор создается в данном пункте задания. При значении NEW необходимо указание также UNIT и, как правило, параметров DCB и SPACE (см. ниже);

OLD — НД существует и программа шага задания получает монопольный доступ к нему;

SHR — то же, но доступ происходит совместно с другими, активными в данный момент, заданиями.

Второй подпараметр (диспозиция):

KEEP — сохранить НД после завершения задания;

DELETE — удалить набор после завершения шага задания;

PA88 — сохранить в течение задания, но удалить после завершения последнего шага задания.

Третий подпараметр (условная диспозиция) может иметь значения KEEP, DELETE, PASS, но определяет, что должно происходить с файлом при аварийном завершении. Если параметр отсутствует, временные НД уничтожаются.

*Параметр UNIT* —указывается при создании новых НД или при использовании существующих сохраненных наборов. Имеет вид:

Групповое имя: UNIT = (тип устройства, [число устройств]) адрес.

Оба подпараметра — позиционные.

Групповое имя: TAPE — магнитные ленты; SYSDA — магнитные диски.

Тип устройства: 5010 или 2311 — любой из НМД с пакетами емкостью 7,25 Мбайт; 5061 или 2314 — НМД с пакетами емкостью 29 Мбайт; 5066 или 3330 — НМД емкостью ПО Мбайт; 5010 — накопитель на МЛ и т. п.

Адрес устройства — трехзначное шестнадцатеричное число, состоящее из номера канала ввода/вывода и номера устройства:

| Примеры типичных адресов устройств | |
|---|---|
| 00A, 00C | Устройства ввода с ПК |
| 00E, 00F | Алфавитно-цифровые построчные печатающие устройства (АЦПУ) |
| 190, 191... | Накопители на МД емкостью 7,25 Мбайт |
| 130, 131... | НМД 29 Мбайт |
| 151, 152... | НМД 100 Мбайт |
| 280, 281... | Накопители на МЛ |
| 0C0, 0C1... | Видеотерминалы ЕС 7920 |
| 4C0, 4C1... | Видеотерминалы ЕС 7970 |
| 01F - | Пишущая машинка (консоль оператора) |

Подпараметр «число устройств» — десятичное целое число.

*Параметр VOLUME* указывает имя тома на МЛ или МД, на котором Должен быть размещен или найден набор данных. Наиболее

употребительной является является структура параметра, указывающая на серийный номер, записанный в физической метке тома:

VOL= SEK = имя_тома.

*Параметр SPACE* используется при размещении нового НД на диске и характеризует выделяемое для этого пространство памяти. Чаше всего задается в виде:

$$\text{SPACE} = (\left\{\begin{matrix}\text{KEEP}\\ \text{DELETE}\\ \text{PASS}\end{matrix}\right\}, (\text{КОЛИЧЕСТВО } [,\text{ПРИРАЩЕНИЕ}])),$$

где первый, обязательный, параметр указывает, в каких единицах измеряется выделенная память:

— TRK — в дорожках;

— CYL — в цилиндрах;

— ЧИСЛО — размер блока (в которых измеряется объем выделенной памяти).

В зависимости от типа устройства единицы измерения имеют различные количественные характеристики:

— для ЕС 5061 — в дорожке 7294 байт, в цилиндре 20 дорожек, в пакете 200 цилиндров;

— для ЕС 5066 — 400 цилиндров по 19 дорожек, каждая длиной 13 Кбайт.

Размер блока не должен превышать длину дорожки. КОЛИЧЕСТВО и ПРИРАЩЕНИЕ определяют объем первичного экстента (непрерывной области) в указанных единицах и вторичного экстента, выделяемого при переполнении первичного. Всего может быть выделено 16 вторичных экстентов, после чего происходит аварийное завершение задания.

*Параметр DCB* содержит сведения, необходимые для обработки записей, из которых состоит НД; как правило, должен указываться для новых наборов данных. Имеет структуру:

DCB = (ключевые подпараметры).

Всего имеется 23 подпараметра, некоторые взаимоисключающие; ниже перечисляются наиболее употребимые:

$$\text{DSORG} = \left\{\begin{matrix}\text{PO}\\ \text{P3}\\ \text{DA}\end{matrix}\right\} \text{— организация НД,}$$

где PO — библиотечный НД или состоящий из именованных модулей, разделов; P5 — последовательный набор данных простейшей организации; DA — набор прямого доступа.

$$RECFM = \left\{ \begin{matrix} F \\ FB \\ V \\ VB \end{matrix} \right\} \text{ — формат записи,}$$

где:

P — записи фиксированной длины, не объединенные в блоки;

V — несблокированные записи переменной длины;

PB — сблокированные записи фиксированной длины;

VB — сблокированные записи переменной длины;

LRECL = ЧИСЛО, логическая длина записи, длина записи в байтах;

BLKSIZE = ЧИСЛО — длина блока записей в байтах (для форматов FB,VB). Блок записей, или физическая запись, есть массив данных, передаваемый за одно обращение из буфера оперативной памяти на внешний носитель, или наоборот. Длина блока, как правило, не превышает длины дорожки (для МД) и не ограничена для МЛ;

BUFNO = ЧИСЛО — количество буферов в оперативной памяти, отводимых под считывание блоков НД. Длина буфера равна длине блока. С увеличением BUFNO уменьшается количество обращений к внешним носителям при работе программ, однако увеличивается расход оперативной памяти.

Параметр LABEL=(ЧИСЛО, ТИП) — используется для указания порядкового номера файла, расположенного на МЛ, и типа меток МЛ (например, NL — лента без меток; SL — стандартные метки и т. п.).

Перечисленные параметры используются в общих случаях — при размещении наборов данных на конкретных устройствах и типах устройств. В частных случаях, когда НД размещены во входном или выходном потоке ОС (рис 3.6), применяются следующие параметры:

* — позиционный параметр, помещенный сразу после «DD» в DD-операторе и означающий, что во входном потоке далее находятся данные, входящие в данный набор;

SYSOUT=КЛАСС, означает, что набор данных является выходным и помещается в выходную очередь указанного КЛАССА (как правило, на соответствующее печатающее устройство).

Наконец, параметр DUMMY в DD-предложении символизирует «пустой НД» и является своеобразной «заглушкой» для входных/выходных НД.

***Утилиты ОС IBM/360.*** Основные операции по обработке файлов ОС ЕС осуществляются служебными программами-утилитами. Наиболее употребительными являются следующие: IEBGENER, IEHPTPCH, IEHMOVE, IEHDASDR, IEHINITT, хотя в состав ОС ЕС входит более 50 таких программ.

На рис. 3.2 приведена типовая структура информационных потоков в утилитах и общепринятые обозначения входных/выходных НД.

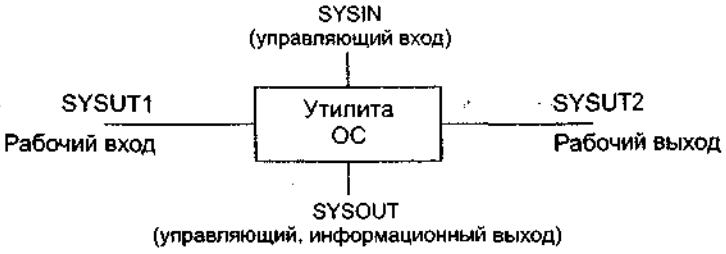



Рис. 3.2. Схема информационных потоков в утилитах

*Управляющий вход* представляет собой файл, обычно размещаемый во входном потоке и состоящий из одной или нескольких *управляющих карт,* которые содержат списки параметров утилиты, как *ключевых,* так и *позиционных.* Управляющая карта задает режим работы, адреса обрабатываемых файлов, количество преобразуемых записей файла, тип преобразования, формат представления выходного результата и т. д.

*Информационный выход* содержит сообщения об ошибках в управляющих картах и при обработке данных, протокол работы (число считанных и выведенных записей, содержимое обрабатываемых файлов, имена обрабатываемых файлов и т. д.).

*Рабочий вход* (может быть несколько) — это обычно входной файл утилиты с исходными данными.

*Рабочий выход* (также может быть несколько) — выходной файл (файлы) с результатами работы.

*Назначение и функции утилит:*

IEBGENER — создание НД на входных устройствах, копирование последовательных, прямых НД и разделов библиотечных наборов.

На рис. 3.3 приведен пример задания на копирование массива перфокарт, установленного на устройстве с адресом ООС на МД с именем MASTER, в качестве раздела NATEXEC ранее существовавшего библиотечного НДADA4.SOURCE.

```
ОС ЕС ЭВМ.
//GEN    JOB
//G EXEC PGM=IEBGENER
//SYSUT1 DD UNIT=00C
//SYSUT2 DD
UNIT=SYSDA,DISP=OLD,DSN=ADA4.SOURCE(NATEXEC),
//                          VOL=SER=MASTER
//SYSPRINT DD SYSOUT=A
//SYSIN DD DUMMY
//
```

Рис. 3.3. Пример JCL-операторов IEBGENER — копирование перфокарт на МД

IEHPTPCH — выдача на печать или перфорацию последовательных или библиотечных НД (рис. 3.4).

IEHPROGM — универсальная утилита по созданию, уничтожению НД, каталогов, оглавлений и других системных таблиц.

IEHMOVE — копирование библиотечных, последовательных, прямых НД с МД на МЛ и с МЛ на МД, а также вывод на перфокарты и ввод с перфокарт.

```
//PTP JOB MSGLEVEL=(1,2)
//PT EXEC PGM=IEBPTPCH
//SYSPRINT DD SYSOUT=A
//SYSUT2 DD SYSOUT=A
//SYSUT1 DD *
ПРИМЕР ПЕЧАТИ ПК                                          *)
//SYSIN DD *
PUNCH STOPAFT=2
ОС ЕС IEBPTPCH 63-09.81 PUNCH STOPAFT=2
IEB4231   USE STANDARD FORMAT
IEB4251   EOF ON SYSIN                                   **)
IEB4281   REQUESTED RECORDS WRITTEN FOR SDS
IEB4441   0000000002 INPUT DATA RECORDS READ
IEB4451   0000000002 OUTPUT DATA RECORDS WRITTEN
IEB4461   0000000000 TITLE RECORDS WRITTEN
IEB4421   END OF JOB IEBPTPCH
IEB4431   HIGHEST CONDITION CODE WAS 00
ПРИМЕР ПЕЧАТИ ПК                                        ***)
```

Рис. 3.4. Пример распечатки, иллюстрирующей работу IEBPTPCH: *) — входной рабочий набор SYSUT1, содержащий 3 перфокарты с текстовой информацией, из которых, согласно содержимому управляющего входа SYSIN, должно быть считано 2 и выведено на печать (согласно JCL-карты SYSUT2); **) — протокол работы, также выводимый на печать; ***) — выходной рабочий набор — копия двух перфокарт, выведенная также на печать

Выводимые утилитой данные оформляются в стандартном формате, одинаковом для МД, МЛ и перфокарт, что удобно для архивации данных.

На рис. 3.5 приведены карты задания на копирование библиотечного НД IZMIRAN с МД MASTER на МЛ с именем HELP01, имеющую стандартные метки, в качестве файла номер 5:

```
//CDT JOB
//C EXEC PGM=IEHMOVE
//SYSUT1 DD UNIT=SYSDA,SPACE=(TRK,10)
//TAPE DD
UNIT=5010,DISP=SHR,LABEL=(,SL),VOL=SER=HELP01
//DISK DD UNIT=5061,DISP=SHR,VOL=SER=MASTER
//SYSIN DD *
COPY
PDS=IZMIRAN,FROM=5061=MASTER,TO=5010=(HELP01,5)
//
```

Рис. 3.5. JCL — операторы утилиты IEHMOVE

IHEDASDR — утилита обслуживания пакетов магнитных дисков, форматирование (инициализация) тома, присвоение имени, выявление дефектных дорожек и размещение альтернативных; создание дампа (образа) диска на МЛ — сохранение МД; восстановление МД по образу на МЛ; прямое полное копирование одного диска на другой.

IEHINITT — инициализация МЛ со стандартными метками — запись в физическое начало ленты файла-метки, содержащего имя тома и необходимую информацию, а также признака конца МЛ, или закрытия ленты.

В ОС ЕС представлены средства *каталогизированных процедур*, предназначенных прежде всего для облегчения использования утилит системы. В специализированных системных библиотеках хранятся заранее подготовленные и соответственно оформленные «заготовки заданий» на выполнение той или иной утилиты или другой программы (совокупности программ), в которых предусмотрены *Формальные именуемые параметры,* значения которых задаются путем подстановки *фактических параметров,* указанных пользователем при запуске процедуры. Например, задание на IEHMOVE (см. выше) могло быть результатом следующего обращения к некоторой процедуре CDT (копирование с МД на МЛ) с формальными параметрами:

T — имя выходной МЛ;

V — имя входного МД;

D — имя копируемого набора данных;
L — тип меток МЛ;
P — номер файла на ленте.

```
START CDT V=MASTER,T=HELP01,D=IZMIRAN,L=SL,F=5.
```

Имена формальных параметров произвольны, определяются системным администратором и должны быть известны пользователям ОС.

## 3.2. Операционные системы RSX (ОС РВ)

Операционная система RSX (ОС РВ — операционная система с разделением времени) была разработана для машин среднего класса PDP-11 и VAX (фирма Digital Equipment Corporation — DEC). Основная особенность управляющих вычислительных машин типа PDP-11 заключается в том, что взаимодействие между всеми устройствами, входящими в состав комплексов, включая процессор, и оперативным запоминающим устройством (ОЗУ) осуществляется при помощи единого унифицированного интерфейса, получившего название «Общая шина» (ОШ).

В дальнейшем данная архитектура была заимствована для построения персональных компьютеров. Общая шина является каналом, чрез который передаются адреса, данные, управляющие сигналы на все устройства комплекса, включая процессор и память. Физически ОШ представляет собой высокочастотную магистраль передачи данных, состоящую из 56 линий.

Процессор использует установленный набор сигналов для связи с памятью и для связи с внешними устройствами, благодаря чему в системе отсутствуют специальные команды ввода-вывода.

Все устройства комплекса подключаются в ОШ по единому принципу. Некоторым регистрам процессора, регистрам внешних устройств, которые являются источниками или приемниками при передаче информации, на ОШ отводятся адреса. В программах адреса регистров устройств рассматриваются как адреса ячеек памяти, что позволяет обращаться к ним с помощью адресных инструкций. Так, программирование операций вывода данных на внешнее устройство практически сводится к пересылке этих данных по определенному адресу.

VAX-11 - более развитая машина, чем PDP-11. Это 32-битовая машина с адресным пространством свыше 4Гб. Она по архитектуре

похо... на PDP-11, но имеет 2 шинных адаптера — адаптер общей ши... и адаптер массовой шины. Все совместимые с общей шиной пер...ферийные устройства могут быть подключены к ней, тогда как вы...коскоростные устройства могут быть подключены к массовой ... не через собственные контроллеры. VAX — сокращение от анг...йских слов Virtual Adress eXtension «виртуальное адресное расши...рение» т. е. машина имеет виртуальную память и многозадачность.

В отечественной практике аналогами данных машин являлись системы СМ ЭВМ — Семейство Малых ЭВМ.

Средства управления вычислениями и манипулирования файлами в СМ ЭВМ (ОС РВ). Следует отметить, что в случае ЕС ЭВМ большинство используемых ОС представляют собой семейство систем с расширяющимися от поколения к поколению возможностями и в основном совместимых между собой по формату данных, программ, утилитам, компиляторам, имеющих стандартный язык управления заданиями.

Для СМ ЭВМ наряду с широко используемой операционной системой разделения времени (ОСРВ) распространены ОС ДИАМС, ТЗХ, UNIX и другие, несовместимые и разнородные по указанным аспектам.

***Некоторые основные понятия, связанные с функционированием ОСРВ (RSX).*** *Пользователь* — лицо, осуществляющее запуск, контроль, остановку некоторого вычислительного процесса, протекающего независимо от других, использующего как монопольно выделяемые, так и общие ресурсы. Обязательно выделяемым ресурсом пользователя является терминал ЭВМ, используемый им для выполнения перечисленных функций. Пользователь должен быть зарегистрирован в системе, с указанием имени, фамилии, пароля, идентификатора. Пользователи разделяются на *привилегированных и обычных.*

Первые имеют доступ ко всем ресурсам, типам программ, команд, операций, вторые — нет. Тип пользователя задается при его регистрации администратором системы.

*Пользовательский идентификатор (UIC — User Identification Co...e)* — код, состоящий из двух чисел С и N, относящий пользова...теля к некоторой группе и присваивающий ему номер в группе. При ор...пи...низации данных на внешних носителях (МД, МЛ) файлы групп...тал...ся по UIC и каждая образованная группа получает статус ка...а (словаря, оглавления, директории) файлов, находящегося в ра...оряжении данного пользователя.

*...адача* — соответственно оформленный и зарегистрированный в сис...еме исполнительный модуль.

*Физические устройства ЭВМ* — регистрируются в ОС, соответствуют фактической конфигурации ЭВМ и идентифицируются кодами: *групповой идентификатор, порядковый номер в группе, символ «двоеточие»*.

Вид идентификаторов основных устройств:

| | |
|---|---|
| DKx: | Накопитель на МДс емкостью 5000 блоков по 512 байт (2,5 Мбайт); x - порядковый номер |
| DMx: | Дисковый накопитель емкостью 20 000 блоков (10 Мб) |
| DPx: | МД емкостью 40000 блоков (20 Мб) |
| MTx: | Накопитель на МЛ емкостью 20 тыс. блоков |
| TTx: | Терминал ЭВМ |
| LPx: | Построчное печатающее устройство |
| Файл | Набор данных на внешнем носителе |

Типичная физическая организация последовательного файла на МД представляет собой размещение логических записей переменной длины, разделяющихся стандартными или специально оговоренными символами-терминаторами, в физических блоках стандартной длины (0,5 Кб). В общем случае одна запись может размещаться в нескольких блоках.

Обозначение файла — совокупность символов, идентифицирующих файл и используемых ОС для определения адреса на внешних носителях, состоящая из следующих компонент:

**устройство:[g,n] имя.расширение;версия**

где:

устройство — идентификатор устройства;

**[g,n]** — каталог (UIC);

имя — выбираемое пользователем наименование НД (не более 8 символов);

расширение — идентификатор файла (не более 3 символов), используемый для группирования файлов в типы.

Некоторые стандартные типы файлов, используемые в ОС и пользовательскими программами:

**.ftn** — текст программы на Фортране;

**.bas** — текст программы на ЯП Бейсик;

**.cmd** — командный файл;

**.tsk** — исполнительный модуль;

**.txt** — текстовый файл;

Пример: `DP0:[1,7]ADABAS.TSK;1` — это программный модуль с именем ADABAS, размещенный в директории [1,7] на МД с адресом DP0:.

Сокращенное наименование файла может состоять только из имени. При этом **устройство** и **[g,n]** берутся из системных умолчаний или пользовательских назначений; **расширение** — задается стандартным типом файла; **версия** — максимальная из существующих.

*Спецификация файлов* — соглашения о кратком групповом обозначении некоторой совокупности обрабатываемых, переименовываемых, удаляемых, копируемых и пр. файлов.

В спецификации файлов используются символы маскирования «*» и «?», вносимые в компоненты обозначения файла, причем «*» соответствует допустимой строке символов, а «?» — одному допустимому символу.

Примеры:

`[*,*]*.TSK;2` — все файлы задач во всех директориях, 2-й версии;

`[1, 5]ADA*.SYS` — файлы директории [1,5] с именем, начинающимся с ADA, расширением SYS, 1-й версии;

`[5,5]SYSTEM.??` — файлы с именем SYSTEM, имеющие 2-символьные расширения.

*Командные языки* — средства общения пользователя с системой, используются для построения команд (или командных строк), а также командных файлов. Основными являются языки **MCR** (монитор команд терминала) и **DCL** (командный язык DEC).

Команды МСК имеют следующую общую структуру:

имя **[/ключ 1/ключ** 2. . .], где имя — 3-символьный зарезервированный идентификатор. Ключи определяют особенности выполнения команды и могут иметь параметры.

Основные команды (вводятся пользователем с терминала на подсказку MCR> или >):

**uic/пароль**

**hel имя/пароль** — регистрация входа пользователя в систему с UIC именем многопользовательской защитой и закрепления за ним терминала;

**mou устройство/ключи** — монтирование устройства, закрепление за пользователем устройства;

**run имя задачи** — запуск программы на исполнение;

**abo имя задачи/кл** — прекратить работу задачи;

**set** — команда изменения системных параметров;

**ufd** — создание директории на устройстве;

**dmo устройство** — демонтирование устройства, отсоединение от пользователя (терминала);

Ьуе — регистрация выхода пользователя из системы с много-Пользовательской защитой.

*Утилиты ОС **RSX***— стандартные программы, предназначенные для обслуживания устройств, МД, МЛ, наборов данных и их групп. Активизация утилиты в любом из командных языков осуществляется путем набора и выполнения строки следующей структуры:

имя **[/ключи] [файл-1 ] [= файл-1]]устр-во-1 [/подключи] устр-во-2[/подключи]**

Здесь:

имя — 3-символьное наименование утилиты;

ключи (подключи) — управляющие параметры;

файл **(устройство)-1** — выходной поток информации (результаты);

файл **(устройство)-2** — входной поток информации.

Некоторые наиболее употребимые утилиты:

PMT — утилита форматирования тома на диске — создания структур данных для физического размещения файлов ОС РВ;

INI — присвоение метки тома на диске;

BAD — анализ и запоминание адресов физически поврежденных блоков на диске;

PIP — утилита работы с файлами.

Наиболее употребимый формат команды:

>P1P файл-1 = [файл-2] [/подключи] устр-1

Примеры:

>PIP LP1:=DP2:[*,*]/LI — вывод на печать списка всех файлов на DP2:

>PIP DP1:=DP0:[5,5] — копирование содержимого директории [5,5] диска DP0: в текущую директорию DP1:;

>PIP DP:/FR — показ числа свободных блоков на устройстве DP0;

>PIP DP2:[3,3]*.TSK;*/DE — удаление всех программных модулей из [3,3] устройства DP2.

BRU — утилита быстрого копирования МД на МЛ с целью архивизации и восстановления архива с МЛ на МД.

Пример:

>BRU /NOI/BAC:IVAN1 P1:=MT0: - копирование на устройство DP1: образа диска с именем файла на ленте IVAN1 с ленты на устройстве MT0:

BCD — анализ и распечатка содержимого ленты, подготовленной с помощью BRU, например:

>BCD LP:=MT: — вывод оглавления МЛ, установленной на MT: на печать.

DMP — вывод содержимого блоков информации некоторого файла на экран или печать, например:

>DMP LP:=HOUSES.D/AS/BL:1:2 — определяет распечатку пер... двух блоков файла HOUSES.LOD в коде ASCII.

ACNT — программа работы с учетным файлом пользователей, в ко...ром по каждому зарегистрированному пользователю представ... данные, имя, статус, UIC и уровень доступа к данным. ACNT ле... ...ляет вставлять, удалять, изменять эту информацию.

*Командный файл* — совокупность командных строк, оформлен...ная в виде файла с расширением имени .CMD. Файл обрабатывается *процессором командных файлов,* который распознает и отправляет на исполнение командные строки MCR и DCL, а также анализирует и выполняет операторы и директивы процессора командных файлов.

Некоторые из операторов и директив:

.METKA: — символьный идентификатор, позволяющий осуществлять передачу управления к определенной командной строке;

.ASKN, (.ASKS) — запрос к оператору терминала на ввод числового (символьного) значения, с проверкой на соответствие некоторым пределам;

.SETN (.SETS) — присвоить числовой (символьной) переменной значение;

.GOTO METKA — переход на метку;

.IF — проверить, удовлетворяет ли переменная одному или нескольким условиям;

.BEGIN -.END — блок командных строк, выполняемый полностью при передаче управления на строку.BEGIN;

.EXIT — выход из блока;

.ЗТОР — останов процессора командных файлов.

Запуск командных файлов осуществляется путем ввода пользователем команды: @ИМЯ_ФАЙЛА. Обозначение командного файла Должно иметь расширение .COM.

Возможности процессора командных файлов, следовательно, позволяют готовить на данном языке некоторые «программы», осуществляющие вычисления, диалог с пользователем, вызов команд DCL, MCR, утилит и пользовательских программ.

***Текстовые редакторы ОС РВ.*** Необходимо отметить, что средства редактирования могут быть разделены на 4 группы:

- построчные редакторы, допускающие обработку текущей строки;
- экранные редакторы, манипулирующие с группой строк, размещенных на текущем экране;
- Редакторы набора данных, позволяющие работать с текущим НД в целом;

• редакторы группы файлов, обеспечивающие доступ и обмен информацией внутри совокупности НД одновременно.

В ОС РВ СМ ЭВМ присутствует строчный редактор EDT, однако наиболее распространенным является редактор НД — TED.

Пользователю TED выделяется буфер, в который во время работы помещаются обрабатываемые файлы.

Функциональные возможности TED могут быть разделены на 2 группы:

• *командный режим работы* (манипулирование файлами и предварительное редактирование) — содержимое буфера визуально недоступно. Команда вводится по подсказке TED>;

• *экранный режим* — точное или окончательное визуально контролируемое редактирование содержимого буфера, с использованием функциональной и управляющей клавиатуры.

*Команда TED* состоит из кода и параметров (как обязательных, так и нет):

K файл [N1,N2] — прочитать файл от строки N1 до строки N2 в буфер (если параметры опущены, считывается весь файл);

W [P1,P2] [файл] [N] — содержимое буфера от строки P1 до P2 записать в файл после строки N; если опущены P1 и P2, переписывается весь файл; если опущен «файл», запись производится по месту последнего считанного файла; если опущен N, то запись происходит с начала файла;

PR P1 [P2] — распечатать строки буфера в интервале P1—P2, (или строку P1);

D P1 P2 — удалить строки с P1 до P2 включительно;

SU/текст1/текст2/ — контекстная замена цепочек символов в в буфере текста;

CL — очистка текстового буфера;

5 — перейти в экранный режим;

<Ctrl+Z> — завершение работы TED;

*Экранный режим*

<Ctrl+K> — вставка пустой строки под отмеченной курсором на экране;

<Ctrl+\> — вставка пробела в месте, отмеченном курсором, и сдвиг полустроки вправо;

<Ctrl+]> — удаление отмеченного символа и сдвиг правой полустроки влево;

<Ctrl+^> — удаление отмеченной строки;

<RETURN>, <BK> — перевод курсора на строку вниз;

<Ctrl+C> — выход из экранного режима в командный.

## 3.3. Операционная система UNIX

Операционная система UNIX — одна из самых популярных в мире операционных систем благодаря тому, что ее сопровождает и распространяет большое число компаний. Была разработана Кеном Томпсоном — сотрудником фирмы Bell Laboratories концерна AT&T в 1969 году как многозадачная система для миникомпьютеров и мэйнфреймов. UNIX вобрала в себя целый ряд новых разработок в области операционных систем. В принципе, она создавалась как операционная система для исследователей. При разработке UNIX была поставлена задача создать систему, которая могла бы удовлетворять непрерывно изменяющимся требованиям сотрудников, занимающихся разнообразными исследованиями [5].

В 1970 году Деннис Ритчи вместе с Кеном Томпсоном переписали код системы с машинно-зависимого языка ассемблера (на котором тогда писались все операционные системы) на язык высокого уровня — Си. Это позволило им написать всего одну версию операционной системы UNIX, которую потом можно было компилировать Си-компиляторами на различных машинах. Операционная система UNIX стала, по сути дела, мобильной, то есть способной работать на различных типах машин практически без перепрограммирования. Кроме того, она позволяет иметь несколько видов Shell, т. е. интерфейсов взаимодействия между ядром и пользователем или интерпретаторов.

В 1974 году UNIX была передана университетам «для образовательных целей», а несколько лет спустя нашла коммерческое применение. С тех пор она выросла в одну из наиболее распространенных операционных систем. Сейчас существуют версии UNIX для многих систем, начиная от PC (персонального компьютера) до суперкомпьютеров, таких, как Cray У-MP.

Для проведения сложных экспериментальных исследований, связанных с большим количеством сложных вычислений над большим объемом данных, требуются значительные системные ресурсы. В этом случае многие UNIX системы позволяют организовать кластер, т. е. многомашинный вычислительный комплекс, где все ресурсы компьютеров (дисковое пространство, память, ресурсы процессора) являются разделяемыми и доступными для любого пользователя в соответствии с его правами. В такой системе существует возможность постоянного наращивания мощности кластера, путем подсоединения дополнительных компьютеров, а работа в ней при

этом остается для пользователя абсолютно «прозрачной», как если бы он работал на одном компьютере с огромными ресурсами.

***Основные компоненты ОС UNIX***

core — ядро системы;

kernel — оболочка ядра системы;

development system — средства разработки программ;

manuals — экранные руководства пользователя и программиста;

VP/ix - эмулятор MS-DOS;

UUCP — средства передачи данных по каналам связи;

STREAMS — механизм сетевых протоколов;

TCP/IP — сетевой протокол;

NFS — сетевая файловая система;

X Window — средства сетевых графических интерфейсов пользователя (GUI);

Looking Cla$8 — командная оболочка на базе GUI.

Ядро управляет основными ресурсами (процессор, оперативная память) и периферийными устройствами обмена и хранения данных (магнитные диски, магнитные ленты, принтеры, терминалы, линии связи и т. д.). Одной из функций ядра ОС UNIX является программная поддержка файловой системы (ФС). Командный интерпретатор Shell обслуживает терминал пользователя и транслирует команды в запрос к ядру ОС.

***Основные понятия, связанные с работой пользователя в ОС UNIX***

*Начало и конец сеанса работ.* Каждый пользователь системы имеет:

- *имя пользователя* (для установления взаимодействия пользователей и начисления расходов);
- *пароль пользователя* (для контроля входа в систему и защиты своих данных).

Пользователи могут быть объединены в группы (например, во время работы над совместными проектами) для разделения общих ресурсов, тогда еще есть имя группы пользователей [2].

Один пользователь, называемый superuser, является администратором системы (его имя root). В частности, он «заводит» (регистрирует) всех прочих пользователей.

Можно сменить свой пароль в любое время:

```
login: peter password: .
> passwd
Changing password for peter
Old password:
New password: ***
```

```
Retype new password: ***
> ^D <Ctrl+D> (выход из сеанса)
login:
```

где > (подсказка, prompt) служит приглашением системы к вводу команды. В конкретной системе форма приглашения может быть изменена, т. е. знак «>» может быть заменен на другой символ или строку символов.

*Командная строка и формат команд.* Командная строка — это последовательность слов, разделенных пробелами. Первое слово командной строки — и есть команда, остальные — параметры.

Типы параметров:

- имя файла = идентификатор (использует символы a — z, A — Z, 0 - 9, _., - );
- опция (ключ) уточняет смысл команды (обычно начинается со знака «минус»).

Например, опция **-al** (может быть со знаком + или без него). Смысл опции зависит от команды. Выражение описывает обычно строку символов или является строкой.

Порядок параметров в команде:

command options expression filfename(s),

*Некоторые простые команды*

Вывод на экран текущей даты:

```
> date Sun Fed 14 11:38 1995
>
```

Получение списка всех (активных) пользователей:

```
> who
дода tty0 Feb 14 08:30
peter tty5 Feb 14 08:32
```

Вариант:

```
> who am 1
дода tty0 РеЬ 14 08:30
```

Другие примеры команд:

```
rm old.news bod.news
rm -fr goodies.c baddies.o
дгер -о "тагу" people
```

Команды разделяются либо концом строки, либо точкой с запятой, например:

```
> who; date
тагу tty0 Feb 14 08:30
sun ГеЬ 14 11:38 1995
>
```

*Исправление ошибок при наборе текста команды.* При наборе текста команды пользователь может совершить ошибку. Для ее исправления предусмотрены следующие возможности.

Исправление последней буквы:

- **backspace**;
- **^H**;
- # (диез).

Исправление последней строки:

**^x**;
**^v**;
e.

*Приостановка/продолжение вывода на экран*

**^s** — приостановка;
**^Q** — продолжение.

Для остановки выполнения команды используются **^C** или клавиша <Break> (не работает по линиям связи).

***Каталоги и файлы.*** При регистрации пользователя администратором системы ему назначается собственный каталог пользователя (Ноте directory).

*Правила (соглашения) по наименованию каталогов и файлов.* В именах каталогов и файлов малые и большие буквы считаются различными. Символы: '.' (точка) и '_' (знак подчеркивания) — не могут использоваться в качестве первой буквы имени. Как правило, имя файла имеет так называемое «расширение», которое характеризует тип файла. Расширение содержит символ, который следует после точки, например:

**.c** — программа на Си (например, **program.c);**

.Ь — текст файла-заголовка (header), включаемый в программу на Си;

**.f** — программа на языке ФОРТРАН;

.р — программа на языке ПАСКАЛЬ;

**.o** — объектный код, полученный транслятором с любого языка;

**.a** — библиотечный (архивный) файл.

*Использование маскирующих метасимволов* *, ? и []. Метасимволы служат для подстановки любых строк и символов в именах файлов и в командах языка заданий Shell:

***** — представляет произвольную строку (возможно, пустую);

? — любой одиночный знак;

[C1 C2] — любая литера из диапазона C1 C2 (в стандарте ASCII).

Примеры:

```
1) > ls c?
c1 c2 c3 c5 c2
2) > 1s c*
cl c12 c2 c23 c3 cs csl cxy cz
3) > 15 ?1*
cl c12
4) > 1s *1*
cl c12 csl
5) > 15 c [12 x y z]
cl c2 cz
6) > 1s c [12 x y 2 *]
cl c2 c12 c25 cz cxy
```

*Неотображаемые («непечатаемые») символы в именах файлов.* Непечатаемыми являются символы со знаком control (^):

**^A (<Ctrl+A>),;**
**^[ <Esc>,**

и т. п., полученные одновременным нажатием клавиши <Ctrl> и указанной после символа ^ клавиши. Они не видны на экране. В некоторых случаях это может привести к недоразумениям, например команда **ls** может показать файл, а rt и другие команды могут не принять имя этого файла (т. к. часть символов не видна).

Выход состоит в использовании метасимволов * в именах (или использование режима **rm -i**):

```
>1s
argon circle square triangle
> rt square
rt: square non-existent (если вместо q в имени ^q)
> Is 5* square
> rt 5*
>.
```

Теперь файл удален.

*Структура корневого каталога.* Как правило, корневой каталог имеет следующую структуру (рис. 3.6), но администратор системы может ее изменить.

*Печать рабочего каталога.* Узнать имя текущего (рабочего) каталога можно следующим образом:

```
> pwd (print working directory)
/users/mary
```

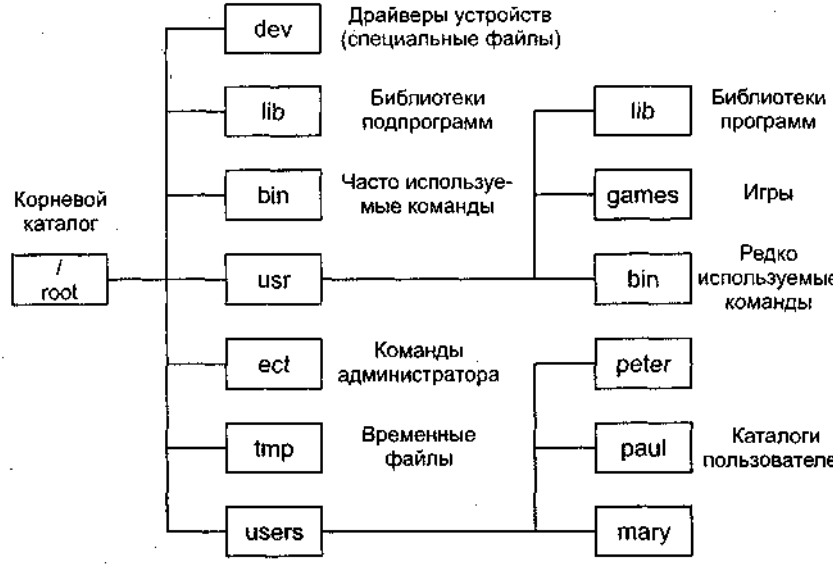



Рис. 3.6. Логическая структура файловой системы ОС Unix

В результате исполнения этой команды на экран выводится полное имя рабочего (текущего) каталога /users/mary, где users — имя охватывающего каталога в иерархии каталогов ФС. Еще один охватывающий каталог (root) — подразумевается по умолчанию (это корневой главный каталог, т. е. root/users/mary эквивалентно /users/mary)/.

Изменить текущий каталог можно с помощью команды **cd** (позднее эта команда будет рассмотрена более подробно).

*Печать содержимого каталога.* Для печати (вывода на экран) каталога используется команда **ls** (list).

```
> ls
> ls -al
> ls /
> ls -l /
> ls -l /bin
> ls -l /bin/bin
> ls /dev
> ls /users
```

Печать текущего каталога:

**-a (all)** — все файлы и подкаталоги;

-1 (1опд) — полную информацию;
/ печать каталога **root**;
печать каталога root полная;
часто используемые команды;
редко используемые команды;
драйверы устройств;
каталоги пользователей.

*Изменение рабочего каталога.* Изменение рабочего каталога производится командой **cd** (change directory).

```
> сё /etc
> ls -1
> cd /usr
> ls -1 bin
> cd
```

— печать команд администратора;
— редко используемые команды;
— без параметров — возврат к собственному (home) каталогу.

*Создание каталога пользователем.* Рабочий каталог пользователя, являющийся корнем вашей ветви ФС, создается администратором системы. Создать нижележащие каталоги можно командой **mkdir** (make directory).

*Уничтожение пустого каталога.* Пустой каталог удаляется посредством команды **rmdir** (remove directory):

```
> rmdir progs
[rmdir: progs not empty] — сообщение, поступающее если каталог не пуст.
```

*Печать содержимого файла.* Команда **cat** (от concatenate) позволяет объединить несколько (один или более) файлов и направить результат на стандартный вывод (Standart Output), обычно — на экран дисплея, например:

```
> cat /etc/motd — просмотреть файл (message of today)
The system will down at 17:00
```

Этот файл, создаваемый администратором для текущих объявлений, обычно печатается автоматически при входе в систему. Если вы забыли его содержание, можно напечатать еще раз, как в приведенном примере.

*Создание файла.* Файл может быть создан командой **cat**.

Если не задано имя входного файла, то будет предполагаться стандартный ввод (клавиатура). Одновременно перенаправим вывод в новый (создаваемый) файл:

```
>cat > newfile
```

Здесь стандартный ввод буферизован, т. е. накапливается несколько (обычно 512) знаков и потом осуществляется запись в файл.

Небуферизованный ввод предпочтительнее (на случай сбоев). Для организации небуферизованного ввода используется ключ -и (unbuffered).

Пример:

```
> cat
Mary101
5a11y113
Jana121
^d [для конца текста (EOP) ]
>
```

Можно добавлять данные в файл:

```
> cat - и >> people
Charlie122
Bill100
^d
>.
```

*Просмотр конца файла.* Команда **tail** печатает конец файла. По умолчанию — 10 последних строк:

```
> tail /usr/pub/greek
```

Явно можно задать количество (со знаком '-') или номер строки, от которой печатать до конца (со знаком '+'):

```
> tail -3 /usr/greek
> tail +6 /usr/greek
```

— три последние строки;

— последние строки, начиная с 6-й.

*Определение типа файла.* Для определения типа файла применяется команда **file**:

```
> Ше /bin/ls /изг/Ып etc/passwd
usr/include/stdio.n
/bin/ls:pure executable — исполняемый;
/usr/bin :directory — каталог;
/etc/passwd:ascii text — текст в коде ASCII;
/usr/include/stdio.h:C program text — текст Си-программы.
```

*Копирование файлов.* Копирование файлов осуществляется командой cp (copy).

Если текущим каталогом является progs/c, то, введя команду:

```
> cp /etc/motd message
```

можно создать в текущем каталоге progs/c файл progs/c/message.

Его можно было бы скопировать в текущий или другой каталог, не меняя имени,

```
> cp /etc/motd progs/c
```

(при этом будет создан файл progs/c/motd).

Если progs/c — текущий каталог, того же эффекта достигает команда:

```
> cp /ets/motd
```

(с точкой в качестве параметра).

Если второй аргумент команды cp — каталог, то в качестве первого можно указать несколько файлов, например:

```
> cp /etc/motd /usr/include/stdio.h progs/c
```

Режим доступа копии файла совпадает с режимом доступа исходного файла.

*Перемещение и переименование файлов.* Перенос и переименование файлов осуществляются командой **mv** (move). Эта команда перемещает файл или каталог из одного места файловой системы в другое. Побочный эффект — переименование файла.

Пример:

```
> mv message mesg
> ls
тезд
```

Отличие от копирования состоит в том, что исходный файл будет уничтожен.

Если файл назначения защищен от записи, то система печатает режим доступа и спрашивает подтверждение переноса, например:

```
> cp теззаде mesg
> chmod 444 message — только чтение
> mv тезд теззаде
message:mode 444? y
```

Второй аргумент команды **mv** может быть каталогом, тогда файл переносится под тем же именем:

```
> mv теззаде progs/c
> cd progs.c
> ls
теззаде
```

Как и в случае команды cp, может быть перенесено несколько Файлов, если второй аргумент — каталог (с точкой в качестве параметра).

*Удаление файлов.* Удаление (уничтожение) файлов производится с помощью команды **rm** (remove).

Пример:

```
> cd progs/c
> rm motd
> ls
message (нет motd)
```

Одной командой rm можно удалить несколько файлов.

Ключ -i позволяет использовать интерактивный вариант исполнения команды, требующий подтверждения на удаление каждого файла:

```
> rm-i file1 file2
file1: n (no - нет)
file2: y (yes - да)
```

Можно форсировать уничтожение ключом -f, даже если файл защищен по записи, например:

```
> rm -f stdio.h
```

ПРИМЕЧАНИЕ 1: ключи **-f** и **-i** взаимно исключают друг друга.

ПРИМЕЧАНИЕ 2: этими средствами нельзя уничтожить каталог, пока не уничтожены все файлы, входящие в его состав.

Для удаления нескольких каталогов используется ключ рекурсивного уничтожения -r (уничтожить все поддиректории):

```
> rm -r progs
```

**Вариант:**

> rm -r * — уничтожить все поддерево.

### *Владелец файла и защита файла*

Каждый файл и каталог имеют владельца - обычно это пользователь, создавший их в первый раз. Владелец может затем назначить тип (вид) защиты файла от трех категорий пользователей:

- владельца (самого себя);
- представителей той же группы пользователей, что и владелец;
- всех остальных пользователей системы.

Каждый файл имеет 3 вида разрешения на доступ:

- чтение r(read) — можно читать (смотреть) содержимое файла или каталога (читать с ключом -l в команде **ls**);
- запись w (write) — можно менять содержимое файла или каталога (создавать или удалять файлы в каталоге);
- выполнять x (execute) — использовать файл как команду UNIX и искать (search) в каталоге.

Все комбинации трех видов разрешения доступа для трех классов пользователей (9 комбинаций) записываются в формате (если все права есть)

*Таблица 3.2.* **Некоторые варианты назначения доступа к файлам / каталогам (в скобках — восьмеричные числа, кратко характеризующие атрибуты)**

| Владелец | Группа | Остальные | Краткая запись |
|---|---|---|---|
| rwx (7) | го/х гок (7) | rwx (7) | 777 |
| rw-(6) | rw-(6) | то-rw-(6) | 666 |
| ... | ... | ... | ... |
| r–(4) | r–(4) | r–(4) | 444 |

Пример:

```
> ls-l /bin
-r~xr-xr-x 1bin 1996 Nov.26 12
```

Эта команда показывает режимы доступа.

*Команда* **chmod** *(установка и изменение режима доступа файлу).* Формат команды **chmod** (change mode) для установки режима:

```
chmod <режим> <файлы>.
```

Пример:

```
> chmod 644 fl 12 f3,
```

где 644 соответствует rw-r--r--.

Формат команды **chmod** для изменения режима:

```
chmod <изменения> <файлы> .
```

В изменениях используются обозначения:

и — user; г — read;
g — group; w — write;
o — other; x — execute;
a — all;
= — назначить;
+ — добавить;
——отнять.

***Работа с текстовыми файлами.*** Она выполняется достаточно простыми средствами печати файлов, поиска строк, замены букв и строк, сравнения файлов и тому подобное. Соответствующие утилиты ОС весьма эффективны для выполнения таких работ по сравнению с обычным текстовым редактором.

*Печать файла.* Простейший способ печати — это перенаправление стандартного вывода команды **cat** на терминал, имеющий устройство печати.

Пример:

```
> cat file > /dev/lp
```

Однако не все терминалы имеют собственное устройство печати.

В случае общего устройства печати система обеспечивает его коллективное использование, при котором заявка на печать, заставшая устройство занятым, ставится в очередь до момента освобождения устройства, после чего ее требование удовлетворяется автоматически.

В этом случае используется команда **lpr** или **1p,** например:

```
> 1pr filea fileb filec.
```

Имеются также команды просмотра очереди заявок на печать **lpstat** и удаления заявки из очереди (**cancel**).

Команды 1pr и 1p не обеспечивают разбиение текста на страницы. Это может быть сделано командой **pr** (prepare) подготовки файла для постраничной печати, которая предшествует печати.

Пример:

```
> pr myfile | 1pr
```

Размеры страницы по умолчанию равны 66 строкам, длина строки — 72 знака.

Ключами **-w** (width — ширина) и -/ (length — длина) можно задать другие размеры.

Примеры:

```
> pr -w 132 -h "Conversion program" conv.c | 1pr
> pr -1 25 addr | 1pr
```

Ключ -А (header) вводит заголовок печати. Двойные кавычки требуются при наличии в заголовке пробелов, иначе они могут быть опущены.

*Разборка и сборка файла.* Многие команды ограничивают размеры файла, который они могут обработать. Если файл слишком велик, он может быть разбит на части командой **split**, а после обработки его можно собрать из этих частей командой **cat**. Каждая часть становится независимым файлом с именами, по умолчанию, **xaa,** хаЬ, хас,.... **xzz.** Размер части (по умолчанию) — 1000 строк.

Пример:

```
> split bigfile
```

Можно изменить размер, устанавливаемый по умолчанию, задавая его явно, например 500 строк:

```
> split -500 bigfile
```

Можно задавать имена частей, например:

```
> split bigfile part
```

В этом случае имена файлов будут: **partaa, partab,... partzz.**

После необходимой обработки всех или некоторых частей исходного файла его сборка из частей выполняется, например, следующим образом:

```
> cat part?? > bigfile.new,
```

где "??" — метасимволы.

Типичным примером применения технологии разборки-сборки является печать отдельных страниц файла.

Пример:

```
>pr bigfile > bigpr
>split -66 bigpr
>lpr xaf xaj
```

Здесь будут напечатаны 6-я и 10-я страницы размером по 66 строк исходного файла.

*Сортировка текстовых файлов.* Утилита **sort** упорядочивает записи файла в алфавитно-цифровом порядке.

Пример:

```
> sort people
Bill Williams 100
Henry Morgan 112
Mary Clark 101
>
```

Записи отсортированы по первой букве имени, однако можно выполнить сортировку по фамилиям:

```
> sort +1 people
Mary Clark 102
Henry Morgan 112
Bill Williams 100
```

Ключ +1 означает, что одно слово (поле) с начала строки текста (записи) было игнорировано при сортировке.

Сортировка по третьему полю с игнорированием лидирующих пробелов выполняется с использованием ключа -b (blank):

```
> sort -b +2 people
Bill Williams100
```

```
Mary Clark101
HenryMorgan112
```

Для сохранения результата сортировки в файле используется ключ -o (output):

```
sort -o sort people +1 people
```

Для слияния уже отсортированных файлов используется ключ -т. Пример:

```
>sort +1 admpeople > sortadm
>sort +1 hardpeople > sorthard
>sort +1 sortpeople > sortsoft
>sort -т +1 sortadm, sorthard, sortsoft > sortall
>.
```

Имеется возможность удаления дублированных записей, используя ключ -и (uniq), а также сортировки по нескольким несмежным ПОЛЯМ.

*Подсчет строк, слов и знаков в файле.* Подсчет числа строк, слов и знаков в заданном файле выполняется командой **wc**.

Пример:

```
> wc people
3951people
>.
```

Ключи **-l** (lines), **-w** (words) и -с (characters) могут указать явно объекты счета, например:

```
> wc -1 people
3 people
> wc -1c people
1051people
>
```

*Поиск строк в файле по образцу (утилита grep).* Утилита **grep** (аббревиатура от global regular expression printer) осуществляет поиск по одному или нескольким файлам и печатает все строки, содержащие предъявленный образец текста, на стандартном выводе. В простейшем случае образец задается постоянной строкой знаков. В общем же случае он задается регулярным выражением.

Пример:

```
> дгер Непгу admpeople hardpeople softpeople
Softpeople: Непгу Могдап 112
>
```

ИЛИ:

```
> дгер Непгу *people
```

```
Softpeople: Henry Morgan 112
> -
```

Ключ -V (invert) предписывает печать всех строк, кроме найденных, например:

```
> дгер -V "Непгу Могдап" Softpeople
Bill Williams 100
Mary Clark 101
>.
```

Двойные кавычки требуются для размещения в образце пробелов.

Регулярные выражения позволяют вести поиск типа: найти все слова из четырех букв, начинающиеся на *A, или* все слова, кончающиеся на **able,** и тому подобное.

Рассмотрим примеры задания образцов посредством регулярных выражений.

Знаки ^ и $ помечают начало и конец строки соответственно:

**"^Genesis"** — найти все строки, начинающиеся словом **Genesis**;

**"eschatus$"** — найти все строки, кончающиеся словом **eschatus**;

**"^Out** т **cold$"** — найти все строки, равные образцу.

Точка помечает любую букву:

**"^d..."** — найти все слова из 4 букв, начинающиеся с буквы d;

**"d...$>"** — то же в конце строки;

**"d\."** — найти все слова из 4 букв, начинающиеся с d и оканчивающиеся точкой. Знак \ (обратная косая черта) отменяет специальное значение следующего символа.

Квадратные скобки задают списки возможных значений знака:

**"^[abcxyz]"** — найти все строки, начинающиеся с букв a, Ь, c, x, y или z;

**"^[Dd][a — z][a — r][a — z]"** — найти все слова из 4 букв, начинающиеся с D или d, в которых последние три буквы малые (от a до z).

Фигурные скобки задают количество повторений (замыкание) предыдущего знака:

**"^[Dd][a — z]{3}"** — то же самое, что и предыдущий пример;

**"^[a — z]{3, 5}"** — найти все слова, содержащие от 3 до 5 малых букв. Частные случаи замыкания обозначаются специальным образом:

* — для 0 и более раз;

\+ — для 1 и более раз;

? — для {0, 1} (ноль или один раз).

Пример:

```
> дгер ".*" people
```

Эта команда просто напечатает все строки файла.

Примеры использования регулярных выражений.

Уничтожение всех пустых строк в файле:

```
> grep-v "^ $" file > newfile
```

Уничтожение всех строк, состоящих только из пробелов:

```
> grep-v "^ *$" file > newfile
```

*Трансляция символов (утилита tr).* Утилита **tr** работает со стандартным вводом и имеет два аргумента, задающих упорядоченные множества знаков, причем каждый знак первого множества заменяется соответствующим знаком второго.

Пример:

```
> tr a-z A-Z <people
MAKY CLARK101
HENRY MORGAN112
BILL WILLIAMS100
```

Ключ *-A* позволяет задать множество символов, которые будут уничтожены, например:

```
> tr-d 0-9 < people
Mary Clark
Непгу Могдап
Bill Williams
>.
```

*Команды сравнения файлов.* В процессе разработки программного обеспечения возникает необходимость сравнения версий файла на разных этапах его разработки. Узнать, чем версии отличаются друг от друга, можно с помощью команды **diff,** которая показывает разницу (difference) двух файлов. Сравнение файлов осуществляется по строкам (записям). В результате выполнения команды печатаются строки измененные (с), уничтоженные (**d**) и добавленные (а) во втором файле-аргументе (по сравнению с первым).

Пример:

```
> cat people
Mary Clark101
5a11y Smith113
Jane Buily121
> cat people.new
Mary Clark101
5a11y White113
James Walker112
> diff people people.new
2 c 2
<5a11y Smith113
```

```
>Sally While113
3 a 2
<Jane Baily121
3 a 3
>James Walker112
```

Знаки < и > соответствуют лишним или отсутствующим строкам Эта команда показывает также номера строк, в которых найдены отличия.

Если строки отличаются только числом разделяющих слова пробелов, такие отличия можно подавить ключом -Ь (blank), например:

```
> diff-b oldfile newfile
>.
```

Другая возможность быстрого сравнения файлов — команда стр (сотраге), реализованная на основе побайтового (побуквенного) сравнения двух файлов.

Пример:

```
> стр people people.new
people, people.new differ: char ,17, line 2
```

В качестве результата печатается число отличающихся байтов (букв) и строк (линий).

Ключ -1 (long) позволяет распечатать разницу файлов в виде байтов (адрес и отличающиеся значения).

Пример:

```
> стр people people.new
26123127
27155150
30150155
1976061
1986061
```

Если файлы сильно отличаются друг от друга, их сравнение может быть произведено с помощью команды **comm** (common), которая показывает, что в двух файлах одинаковое (общее).

Пример:

```
> cat people
Mary Clark  101
Sally Smith113
Jane Baily 121
> cat people.new
Mary Clark 101
Sally White 113
James Walker 112
> comm people people.new
```

```
Mary Clark
5ally  Smith
5a11y White
Jane Baily
James Walker
```

Результат команды **comm** печатается в три колонки: строки первого файла, отсутствующие во втором; строки второго файла, отсутствующие в первом, и строки, общие для двух файлов.

Можно подавить печать одного или двух столбцов, указывая его номер в виде ключа, например (печать только третьего столбца):

```
> comm -1 2 people people.new
Mary Clark
>
```

*Обработка текстовых файлов командой awk.* Awk — это утилита, подобная **grep.** Однако, кроме поиска по образцу, она позволяет проверять отношения между полями строк (записей) и выполнять некоторые действия над строками (генерировать отчеты). Название не является акронимом, оно образовано первыми буквами фамилий авторов (A. V. Aho, P. У. Weinberger и B. W. Kernighan).

Задание поиска-действия следует синтаксису:

```
/<образец>/{<действие>}
```

Как образец, так и действие могут отсутствовать. Найденные по образцу строки при отсутствии заданного действия передаются в канал стандартного вывода (на экран). Образец задается регулярным выражением, как и в утилите **grep.** Если образец отсутствует, то обрабатываются все строки.

Рассмотрим примеры действий, которые можно выполнить командой awk.

Перестановка полей строки выполняется с помощью ссылки на поле $n, где n — номер поля, например:

```
> cat people
Mary Clark 101
Henry Morgan 112
Bill Williams 100
> awk '{print $2 "," $1 "^I" $3}' people
Clark, Mary 1 01
Могдап, Непгу112
Williams, Bill 100
```

Здесь **^I (<Ctrl+I>)** — знак табуляции.

Параметры поиска и обработки с помощью утилиты **awk** могут быть заданы в файле, например:

```
> cat swap
{print $2 "," $1 "^I" $3}
> awk -f swap people
```

Утилита **awk** имеет встроенные образцы и переменные. Образцы **BEGIN** и **END** означают начало и конец файла соответственно. Переменная **NR** (Number of Records) означает число записей (строк) в файле, **NF** — число полей (слов) в записи. Можно использовать переменные, объявленные пользователем.

Пример, подсчитывающий среднее значение третьего поля файла **tennis** (программа действий для **awk** находится в файле **average**):

```
> cat > average
{total = total + $3}
ЕГО {print "Average value is", total/NR}
^D
> awk -f average tennis
Average value is 8.9
```

Образец поиска в **awk** может содержать условные выражения.

Пример выборки из файла **tennis** всех записей, значение третьего поля в которых не меньше 10:

```
> awk '$3 >=10 {print $0}' tennis
Steve Daniel11
Hank Parker18
Jack Austen14
>.
```

Знак $0 (доллар-ноль) — это ссылка на всю запись (строку).

В общем случае выражение для условия подчиняется синтаксису, близкому к синтаксису выражений в языке Си. Кроме того, в команде **awk** допустимо указывать отрезок образцов.

Пример выборки всех записей, сделанных с 1996 до 1998 года:

```
> sort -n -o chard.s chard
> awk '/1996/, /1998/ {if($2 < 8.00 print $0}'
chard.s
1996 7.50 Chateau
1997 7.75 Chateau
1998 5.99 Charles
```

Как видно из примера, в программах действий для **awk** можно использовать управляющие структуры.

Пример цикла для печати полей всех записей файла в обратном порядке:

```
> awk {for (i = NF; i > 0;--i) print $i} fl,
```

где **NF** — число полей в записи.

***Связь пользователь-пользователь.*** Система UNIX предполагает возможность коллективной работы и кооперации пользователей. Это требует развитых средств связи пользователей между собой. В системе имеются следующие возможности коммуникации:

- команда ***write*** (писать) для немедленной посылки сообщения другому пользователю;
- команда ***mail*** (почта), реализующая электронную почту.

*Посылка сообщений командой write.* Командой **write** посылается сообщение указанному пользователю в момент исполнения этой команды. Это означает, что адресат должен быть в этот момент в системе (on line). Немногим нравится, когда сообщение приходит в случайный момент, в особенности — во время набора собственного текста. Поэтому вы должны быть уверены в необходимости такого немедленного взаимодействия.

Типичный пример:

```
> write paul
Срочно пришли отзыв на статью!! !
^D
>
```

Сообщение заканчивается вводом знака конца файла <Ctrl+D>. Адресат увидит у себя на экране:

```
Message from дода tty 00...
Срочно пришли отзыв на статью! ! !
EOF
```

Если вы намерены ждать ответа и вести диалог, можно не вводить знак конца файла до конца диалога.

Пример:

```
> write тагу
Приглашаю сегодня вечером в гости.
Извини, сегодня я иду в театр.
Тогда завтра?
И завтра не могу. Давай на следующей неделе?
Ну, хорошо.
EOP
^D
>
```

На стороне абонента на экране будет:

```
Message from дода tty 00...
Приглашаю сегодня вечером в гости.
Извини, сегодня я иду в театр.
Тогда завтра?
И завтра не могу. Давай на следующей неделе?
```

```
ну/ хорошо.
^D
```

Чтобы узнать, работает ли абонент в системе в данный момент, можно воспользоваться командой ***who.***

Пример:

```
> who
дода tty0008:30
paul tty0308:31
boris tty0704:12
mary tty0809:01
```

Текст сообщения можно взять из файла, перенаправив стандартный ввод для write из этого файла.

Пример (сообщение из файла message):

```
write peter < message
```

Текст сообщения можно формировать, выполняя команды внутри текста. В этом случае команде должен предшествовать восклицательный знак, например:

```
> write peter
Нужные тебе файлы в каталоге:
!pwd
/users/mary/docs/specs
!
/users/mary/docs/specs
^D
>
```

Ответ команды завершается также восклицательным знаком. Ни команда, ни ответ в текст сообщения не попадают.

Пользователь может запретить получение сообщений на терминал командой:

```
> mesg n
>,
```

где n — от no (нет), и, наоборот, разрешить прием командой:

```
> mesg y
>,
```

где y — от yes (да).

Эта же команда без параметров сообщает, в каком состоянии находится терминал пользователя (no или yes), например:

```
> mesg
is y,
```

т.е. прием разрешен.

При входе в систему устанавливается состояние у.

*Электронная почта (mail).* Электронная почта — это средство, позволяющее пользователям посылать друг другу сообщения которые накапливаются в почтовых ящиках, реализованных в системе.

При входе пользователя в систему он извещается о наличии почты, например:

```
login: дода password: . . .
уои nave mail
>
```

В отличие от команды ***write***, в режиме ***mail*** о приходе почты пользователь извещается только после окончания текущей работы. Поступит сообщение:

уои **have mail** (для вас есть почта).

Для получения почты необходимо ввести команду ***mail*** без параметров:

```
> таИ
From peter Wed Jim 9 17:58:23 1999
Завтра в 16:30 можно поиграть в преферанс.
Если согласен, сообщи, где встретимся
?
```

Подсказка (?) означает, что система mail ждет указания о том, что делать с почтой: печатать, сохранить, уничтожить или выйти из команды mail.

Введя знак вопроса, вы получите меню возможных действий (подкоманды):

```
??
q (quit) закончить;
x (exit without changing mail) выйти без изменения почты;
P (print) печатать;
s [file] save — сохранить в файле;
и [Ше] затеадИКои!header — то же без заголовка;
- print previous — печатать предыдущее;
o! (delete) уничтожить;
+ next (по delete) следующее (не уничтожать);
т изег (mail to user) переправить другому пользователю;
! cmd (execute cmd) выполнить команду;
?
```

Кроме этих десяти подкоманд, можно также ввести <Ctrl+D> (то же, что и **q**) *или* нажать клавишу возврата каретки (<Enter>).

При сохранении почты в файле на экране высвечивается следующее письмо, если оно есть:

```
? s from_peter
From peter
.................
?
```

Можно переслать данное письмо другим пользователям, например, использовав подкоманду *m:*

```
? m тагу boris деогде
```

Для посылки почты необходимо ввести команду **mail** с параметром (имя пользователя) и текст письма, заканчивая его знаком конца файла (<Ctrl+D>), например:

```
> mail peter
Извини, я не смогу играть завтра.
^D
>
```

Посылка ответа может быть произведена при просмотре почты. Пример:

```
?! mail peter
Извини, я...
^D
```

Для просмотра почты из файла можно ввести команду **mail** с ключом -F:

```
> mail -f from_peter
```

Для выхода из режима mail необходимо набрать **q**, например:

```
? q
You have mail
>
```

Работа с почтой закончена.

***Стандартные файлы.*** Многие команды работают по умолчанию со стандартными файлами:

Standard Input (S.I.);
Standard Output (8.O.);
Diagnostic Output (D.O.).

Однако есть средства изменения параметров умолчания, т. е. возможность указать другие файлы вместо стандартных. Можно, например, использовать S.O. в качестве D.O. Эти средства называются перенаправление (redirection) ввода и вывода.

*Перенаправление стандартного вывода.* Для перенаправления вывода используется знак >.

Примеры:

> ls-l — вывод на экран = standard output;
> ls-l > **dircnts** — вывод в файл dlrcnts;
> **cat dircnts** — вывод на экран.

Пробелы вокруг символа «>» необязательны. Возможно перенаправление вывода с добавлением (с дописыванием в файл), обозначается >>.

Пример:

> **pwd >> dircnts** — добавить в файл имя текущего каталога.

*Перенаправление стандартного ввода.* Для перенаправления стандартного ввода используется знак <.

Примеры:

> **mail** — ввод сообщения с экрана;
> **mail < message** — ввод сообщения из файла message.

Эта возможность используется реже, чем перенаправление вывода. Тривиальный случай перенаправления ввода:

> **cat < this_file**
> **cat this_file**

(это две эквивалентные команды).

Можно одновременно перенаправить и ввод и вывод, например:

> **cat < left > right**

Нужно, чтобы **left** не равнялось **right**, иначе можно потерять входной файл. Безопаснее использовать знак >>, чем > (т. е. добавление безопаснее, чем запись).

*Организация конвейеров команд.* Конвейером называется группа команд, объединенных программными каналами. Программный канал образуется назначением стандартного вывода одной команды стандартным вводом следующей команды. Для формирования программного канала используется знак | (вертикальная черта).

Пример 1:

> who | wc -1 — создание списка активных пользователей и подсчет их числа (count);
19 — ответ, то есть 19 пользователей.
>

. Пример 2:

| | | | |
|---|---|---|---|
| > **ls -1 /tmp** | \| дгер **maryann** | \| **sort +3nr** | **pr** |
| листинг каталога /tmp | поиск записей, содержащих строку «тагуапп» | сортировка (по 4-му полю) найденныхзаписей | печать упорядоченного списка |

*фильтры.* Так называются команды, которые могут получать данные со стандартного ввода и выводить данные на стандартный вывод. Большинство команд является фильтрами, однако есть исключения, например команда **ls** не может работать со стандартным вводом, а команда lpr не может работать со стандартным выводом.

*Диагностический вывод.* Сообщения об ошибках, возникающих при выполнении команд, используют т. н. диагностический вывод. По умолчанию диагностический (как и стандартный) вывод производится на экран, однако он может быть перенаправлен в любой файл. Для этого используется дескриптор файла (целое число), который для стандартных файлов равен:

0 — Standard input;
1 — Standard output;
2 — Diagnostic output.

Пример:

```
>cat somefile >outfile 2>errfile,
```

где знак > эквивалентен 1>.

Если вы хотите, чтобы сообщения об ошибках нигде не проявлялись, направьте их на **/dev/null.**

Если вы хотите направить сообщения об ошибках туда же, куда производится вывод данных, нужно набрать

```
>cat somefile |& lpr.
```

*Обработка команд в фоновом (background) режиме.* Обычно команды выполняются в интерактивном (foreground) режиме, т. е. «пока вы ждете». Однако, если во время выполнения некоторой команды вы хотите выполнять другие команды, то первую команду можно выполнить в фоновом (background) режиме, например:

```
> nroff doc &
2042
>.
```

Как видно из примера, для организации фонового выполнения команды **nroff** использован завершающий знак &.

Система UNIX создает процесс, который выполняется независимо от командного интерпретатора. Ответ 2042 — это идентификатор созданного процесса (PID).

Стандартный вывод фонового процесса лучше перенаправлять в файл (из-за опасности совмещения на экране стандартного вывода интерактивного и фонового процессов), т. е.:

```
> nroff doc > doc.format &
2042
>
```

или:

```
> nroff doc | lpr &
2042
>
```

Чтобы выяснить состояние фонового процесса, следует использовать команду **ps** (process status):

```
> ps
PID   TTY   TIME   CMD
2036  02    0:05   sh          (login — процесс терминала № 2)
2042  02    0:02   nroff doc   (background)
2043  02    0:01   ps          (cat ps)
2050  08    0:03   sh          (login — процесс терминала № 8)
```

Это показывает, что выполнение команды **nroff** еще не закончилось. Ключи -l и -а в команде **ps** могут дать больше информации об активных процессах:

-l — информация о родителе, адрес, приоритет и т. д.

-а — информация о всех процессах системы.

*Выполнение процессов с низким приоритетом.* Рассмотрим организацию выполнения процесса с низким приоритетом (понижением приоритета). Приоритет процесса определяется значением параметра **nice** (приоритет тем выше, чем меньше **nice).**

```
> nice nroff doc > doc.fmt &
2099
>.
```

Пусть начальное значение **nice** равно 20. Перед исполнением команды этот параметр автоматически инкрементируется (подвергается приращению). По умолчанию величина инкремента равна 10. Таким образом, приоритет созданного процесса будет соответствовать величине **nice** = 30.

Рассмотрим еще один пример:

```
> nice-5 nroff doc > doc.fmt &.
```

Здесь инкремент задан явно и равен -5. В результате параметр **nice** будет равен 25 и соответственно приоритет будет на 5 единиц выше, чем по умолчанию.

*Уничтожение процесса.* Для уничтожения процессов служит команда **kill.**

Пример:

```
> kill 2042  (завершить процесс с PID = 2042)
```

или:

```
> kill 9 2042,
```

число 9 — ключ (сигнал) безусловного останова. Другие сигналы интервала (1, 15), посылаемые системой задаче, можно блокировать написав соответствующую процедуру реакции на сигналы.

***Средства разработки программ.*** Система UNIX обеспечивает богатый набор средств для разработки программ, включающий компиляторы, редактор связей (linker), символьный отладчик, средства ведения программных проектов и разработки языковых процессоров, архивные средства и другие.

*Редакторы ex и VI.* Обозначения ex и **vi** — два различных имени расширенной версии редактора **ей** (который входит в комплект стандартной поставки системы). Эта программа работает как экранно-ориентированный редактор при обращении по имени **vi** и как строчно-ориентированный редактор при обращении по имени **ex.** Редактор **vi** (аббревиатура словосочетания Visual Interpretator — визуальный интерпретатор) может быть включен, а может быть не включен в версию ОС UNIX.

Работа с этими редакторами производится в двух режимах: *командном и ввода текста (1ex( entry).* Переключение в командный режим осуществляется клавишей **<Esc>.**

*Вызов редактора VI*

Пример команды:

```
> vi myfile
```

В качестве параметра может быть указано одно или несколько имен файлов (через пробелы) — для их последовательного вызова на редактирование. Если имя файла не указано, то появится начало пустого файла (курсор в начале первой строки).

Варианты использования команды:

```
> vi + myfile
```

На экране будет находиться конец файла; курсор — в начале последней строки.

```
> vi +10 myfile
```

Файл будет выведен таким образом, что строка 10 окажется в центре экрана, а курсор будет расположен в начале этой строки.

*Выход из редактора VI*

Для выхода из редактора используются два метода:

- если вы хотите запомнить изменения — **Esc:wq!Enter** (нажать на клавишу <Esc>, ввести двоеточие (оно появится в нижней

Пример:

```
> fc -c test.f check prove.f
  ld /lib/frt0.o *.o.-l F77
> ls a.out check.f check.o prove.f prove.o test.f test.o
>
```

Здесь добавлены файл **/lib/frt0.o** стартового модуля для программы на ФОРТРАНе (**/lib/crt0.o** для Си) и библиотека подпрограмм **lF77** для ФОРТРАНа (lc для Си). Могут быть добавлены и другие библиотеки. Обозначение **-lx** является сокращением для **/lib/libx.a** при любом **x**. Следует заметить, что библиотеки указываются последними (не являются ключами команды **ld**).

При автоматическом вызове линкера (редактора связей) стартовый модуль и ряд библиотек вызываются по умолчанию. Чтобы их увидеть, в командах вызова компилятора следует применить ключ — V.

*Библиотеки программ.* Как отмечалось выше, на вход редактора связей могут подаваться не только файлы объектного кода, но и библиотечные файлы, которые оказываются удобным средством хранения объектных модулей, если их становится очень много.

Имя *библиотечного* файла обычно оканчивается на **.a**. Для создания, пополнения и просмотра библиотечных файлов используется команда ar (архив).

Пример создания библиотеки из трех объектных файлов:

```
> ar rcv exam.a test.o check.o prove.o
a - test.o
a - check.o
a - prove.o
>
```

Здесь ключи команды ar означают:

r - заменить (replace) модули в библиотеке;
c - создать (create) библиотеку;
**v** - печатать включаемые модули (verbose).

Теперь мы можем вывести на экран содержимое библиотеки командой ar с ключом **t** (table of content):

```
> ar t exam.a test.o check.o prove.o
```

и ссылаться на библиотеку в командах вызова компиляторов или редактора связей, например:

```
> ar t exar.a
test.o
check.o
prove.o
>
```

части экрана), затем ввести символы **w**, **q**, восклицательный знак и нажать на клавишу <Enter>);

- если нет: **Esc:g!Enter.**

*Вызов компиляторов.* В системе UNIX имеются компиляторы с языков Си, ФОРТРАН-77, ПАСКАЛЬ и другие. Команды вызова компилятора имеют вид **cc, fc**, pc и т. п. Параметрами этих команд являются имена файлов с текстами программ на исходных языках. Имена этих файлов должны иметь расширения **.c, .f, .p** и т. п

Примеры:

```
> cc program.c
> fc test.f
> pc example.p
```

Результатом работы компилятора является файл исполняемого кода, имеющий по умолчанию имя **a.out**. Если вы хотите дать ему другое имя, это имя следует указать явно, с ключом -o, т. е.: -o <имя_файла>.

Пример:

```
>fc -o test test.f
>ls test test.f
>
```

*Редактор связей.* На практике программы создаются из множества раздельно транслируемых модулей, каждый из которых занимает отдельный файл. Результатом компиляции каждого модуля является файл объектного (перемещаемого) кода, имя которого получается заменой исходного расширения **.c** (или **.f,** .p и т. д.) на **.o**. Затем все объектные файлы объединяются с помощью редактора связей в единую программу, помещаемую в файл исполняемого кода.

Редактор связей, (linker) может вызываться как независимой командой **ld,** так и автоматически при выполнении команд вызова компилятора **cc, fc,** pc и т. д. В последнем случае эти команды могут иметь несколько параметров-файлов, имена которых могут оканчиваться не только на **.c, .f, .p,** ..., но и на **.o.**

Файлы исходного текста компилируются, а затем все файлы объектного кода, как полученные в результате компиляции, так и указанные в качестве параметров команды вызова компилятора, передаются редактору связей. Результатом его работы по-прежнему является файл с именем **a.out** (если не указано явно другое имя). При этом, как правило, объектные файлы уничтожаются. Чтобы сохранить их, можно подавить автоматический вызов редактора связей ключом -c (только компиляция) в команде вызова компилятора.

Следует помнить, что порядок размещения модулей в библиотеке существен. Например, если подпрограмма **test** вызывает подпрограмму **check**, то файл **test.o** должен предшествовать файлу **check** в библиотеке.

Для выявления и печати таких зависимостей предназначена команда **lorder.**

***Системное администрирование.*** В функции администратора системы UNIX входит повседневное управление системой во всех аспектах ее существования, таких, как подключение новых пользователей, управление файловой системой, изменение конфигурации и других. Следует заметить, что на персональных ЭВМ эти функции могут исполняться прикладным программистом.

Имеется ряд команд, расположенных обычно в каталоге **/etc,** рассчитанных на управление системой, таких, как **fsck, mount, chown** и т. д. Как правило, каталог **/etc** доступен только суперпользователю, так что системный администратор должен обладать правами суперпользователя.

*Спецпользователи.* Это пользователи, выполняющие действия над системой, недоступные обычным пользователям. Один из них, имеющий неограниченные полномочия, называется суперпользователем и имеет обычно имя **root.** Разные системы могут иметь и других спецпользователей, например пользователя с именем **bin,** обладающих меньшими полномочиями, чем суперпользователь.

Имеются команды, которые может выполнить только суперпользователь, в частности установка даты командой **date,** монтаж файловой системы командой **mount,** создание специальных файлов командой **mknod и др.**

Стать суперпользователем можно несколькими способами. Первый — загрузить систему в режиме единственного пользователя. Другой, применяемый на многопользовательской системе, — выполнить команду **SU** (superuser). Администратору системы рекомендуется входить в систему как обычному пользователю и только в случае необходимости становиться временно суперпользователем по команде **SU.**

*Пользователи и группы.* Имеется два файла с именами **passwd и group,** находящихся в каталоге **/etc**, которые содержат информацию о пользователях и группах пользователей соответственно. Одна запись в файле **passwd** соответствует одному пользователю и имеет следующие текстовые поля, разделенные символом двоеточия.

- имя пользователя;
- пароль пользователя (в закодированном виде);
- целочисленный идентификатор пользователя;

- целочисленный идентификатор группы;
- комментарий, который содержит сведения о месте работы пользователя и может использоваться командой **finger** и учетными программами;
- каталог пользователя;
- интерпретатор команд пользователя.

Пример записи файла **passwd** (с пустым комментарием):

```
mary:KmHulhE:201:10::/users/mary:/bin/csh
```

При наличии комментария его синтаксис определяется учетными программами.

Для некоторых системных программ требуется, чтобы идентификатор суперпользователя был равен нулю, а имя — **root.**

Одна запись в файле **group** соответствует одной группе и представляет собой строку текста со следующими полями, разделенными двоеточиями:

- имя группы;
- пароль группы (в закодированном виде);
- целочисленный идентификатор группы;
- список имен пользователей группы, разделенных запятыми.

Пример записи файла **group** (для группы без пароля):

```
sect2115::10:mary, sas, temp, дез
```

Добавление нового пользователя в системе требует выполнения следующих трех действий, которые обычно реализуются командным файлом с именем **newuser или adduser:**

- добавить запись в файл **passwd** с информацией о пользователе;
- создать каталог пользователя, причем пользователь должен быть владельцем этого каталога;
- добавить или скорректировать запись в файле **group** в соответствии с членством пользователя в некоторой группе.

Добавление и коррекция записей в файлах **passwd и group** могут выполняться текстовым редактором (если нет командного файла **newuser).**

Каталог пользователя создается суперпользователем (возможно, средством скрипта **newuser)** и вначале принадлежит ему. Чтобы изменить пользовательскую и групповую принадлежность каталога (и любого файла), используются команды **chown** (change own) и **chgrp** (change group) соответственно. Их может выполнить только суперпользователь.

Часто новый пользователь забывает свой пароль. Суперпользователь может в этом случае изменить пароль пользователя командой **passwd,** затерев забытый и сообщив пользователю новый:

```
# passwd tary
New passwd: mmm
Retype new passwd: ttt
#
```

• Вводимый пароль (здесь — ttt) не виден.

*Подключение терминалов.* Все терминалы, которые могут быть подключены к системе, должны быть описаны в специальном файле с именем **/etc/ttys** (версия 7) или **/etc/inittab.**

Каждому терминалу в этих файлах соответствует одна строка. Форматы файлов **/etc/ttys** и **/etc/inittab** сходны в том, что первый символ является цифрой, нулевое значение которой соответствует отключенному (логически) терминалу, а единичное — подключенному терминалу. Кроме того, формат обоих файлов предусматривает наличие имени спецфайла, соответствующего терминалу (второе поле в файле **/etc/inittab).**

Типичная строка в файле **/etc/inittab** имеет вид:

```
1:t3:c:/etc/getty tty!3 H O
```

Первое поле имеет подполя, разделенные двоеточием.

При загрузке системы ее последним шагом является запуск начального процесса с номером 1, выполняющего команду **/etc/init.** Команда **init,** отрабатываемая перед выполнением системного стартового командного файла **/etc/re,** просматривает файл **etc/inittab.** Для каждой строки этого файла, начинающейся с ненулевого символа, совпадающего с состоянием начального процесса (при запуске равного единице), команда **init** порождает второй процесс в цепочке (init-getty-login-shell).

Порожденный процесс исполняет команду, указанную в четвертом подполе первого поля файла **/etc/inittab** (в рассмотренном примере — **/etc/getty**).

Команда **getty** выдает на терминал, указанный именем спецфайла во втором поле файла **/etc/inittab,** содержимое файла **/etc/iss**, если он существует, и вслед за этим выводит на терминал текст подсказки из записи файла **/etc/gettydefs,** содержащего характеристики терминала (обычно — **login:).** После этого команда **getty** читает имя пользователя и вызывает команду **login,** передавая ей имя пользователя в качестве параметра.

Команда **login** вводит пароль пользователя и после успешной проверки пароля, выполняет команду из последнего поля записи

данного пользователя в файле **/etc/passwd** (обычно — **/bin/sh** или **/csh**), а также устанавливает в качестве текущего начальный каталог пользователя, указанный в предпоследнем поле записи данного пользователя в файле **/etc/passwd.**

Команда **sh** или **csh** командного интерпретатора выполняет стартовые файлы с предопределенными именами (**.profile** для **sh**; **.cshrc** и **.login** для **csh**), выводит подсказку и ждет ввода очередной команды пользователя.

По окончании сеанса (подачей команды **logout** или ^**D**), а также в случае неверного пароля управление возвращается в команду **getty**, которая перезапускается повторно, если в третьем подполе первого поля записи файла **/etc/inittab** для данного терминала стоит символ C (*continually*); если же в этом поле стоит пробел, то команда **getty** завершается.

*Периодическое выполнение заданий.* Одной из команд, выполняемых в составе системного стартового командного файла **/etc/rc** начальным процессом, может быть команда **/etc/cron,** создающая постоянный процесс, пробуждающийся периодически каждую минуту. Этот процесс просматривает записи файла **/usr/lib/crontab** в поисках заданий, которые должны быть выполнены.

Типичный файл **crontab** может выглядеть следующим образом:

```
cat /usr/lib/crontab
0 0 * * * /etc/backup -fsck
0,15,30,45 2-23 * * * /usr/lib/atrun
```

Первые пять полей записи файла **crontab** означают минуты (0—59), часы (0—23), день месяца (1—31), месяц года (1 — 12) и День недели (0—6, 0 — воскресенье). В каждом из этих полей может быть значение; перечень значений, разделенных запятыми, или границы интервала значений, разделенные минусом. Звездочка означает любое возможное значение.

В приведенном примере первая запись соответствует ежесуточному выполнению (в полночь) сброса и проверки файловой системы в течение всего года, вторая — выполнению программы запуска заданий, запланированных командой at, каждые 15 минут с 2 часов ночи до 11 часов вечера ежедневно в течение всего года.

Команда **at** планирует выполнение командного файла интерпретатора **shell** (shell-скрипта), указанного ее последним аргументом в момент времени, заданный ее первыми аргументами, например:

```
at 2300 jun 16 scriptfile_1
```

Указанный файл будет выполнен в 11 часов вечера в указанный день текущего года. Точность времени запуска зависит от периода

пробуждения постоянного процесса, выполняющего команду cron. Все действия процесса, выполняющего сгоп, фиксируются в учетном файле **/usr/lib/cronlag,** если он существует и открыт на запись в момент выполнения этой команды.

Команду **at** может запустить любой пользователь, чтобы оставить задание на ночное время.

*Управление операционной системой.* Средства управления операционной системой — аппаратно-зависимы, однако приводимое ниже описание этих средств является типичным для большинства версий ОС UNIX.

Операционная система хранится на дисковом томе в некотором заданном формате. Для задания формата используется команда инициализации тома **/lbin/init** с одним обязательным параметром — именем спецфайла для устройства, на котором находится инициализируемый том. Остальные необязательные параметры могут указать размер логического блока, единицы передачи данных между томом и оперативной памятью и размер загрузочной области в байтах (при отсутствии их значения выбираются по умолчанию).

Каждый том имеет одну загрузочную область, содержащую целое, возможно нулевое, число логических блоков. Загрузочная область полностью находится вне какой-либо файловой системы. Изменение ее размера возможно только при переинициализации тома.

Каждая загрузочная область может содержать только одну ОС (или часть одной ОС). ОС состоит из последовательности кодовых сегментов, расположенных в загрузочной области одного или нескольких томов, причем граница между томами может быть внутри некоторого сегмента.

ОС хранится в загрузочном формате. Помимо загрузочной области, она может располагаться также в ряде обычных файлов, каждый из которых содержит целое число кодовых сегментов, заканчивающихся двумя нулевыми байтами. Этот формат не является загрузочным, однако он может быть преобразован в загрузочный командой **/lbin/oscp**.

Содержимое загрузочной области в действительности состоит из одного или нескольких ОС-файлов. Каждый ОС-файл начинается с заголовка, содержащего флаг загружаемое™, номер тома и число томов, занятых операционной системой. Системный загрузчик загружает ОС-файл только в том случае, если флаг установлен в состояние загружаемости. Установку состояния флагов ОС-файлов можно выполнить специальной командой **/lbin/osmark** с параметром, указывающим имя спецфайла для устройства, на которое поставлен том с загрузочной областью. Ключ в этой команде устан-

ливает флаг в состояние загружаемости или незагружаемости. Упомянутая выше команда **/lbin/oscp** позволяет выполнить следующие действия по копированию сегментов ОС:

. копировать ОС из одной (или более) загрузочных областей тома (томов) в загрузочную область другого тома;

. копировать ОС из обычных файлов в ОС-файлы для создания (опция -т, от merge) или модификации (опция -а, от add) ОС в загрузочной области;

« копировать ОС-файлы в обычные файлы для разделения ОС на части (опция — **s**, от split) или в один обычный файл (опция **-f**, от file).

Для проверки целостности ОС в загрузочной области, а также для контроля добавленных сегментов можно выполнить команду **/lbin/osck**, например:

```
osck -V /dev/rhd
```

Опция -V вызывает печать списка имен всех сегментов ОС. Команда проверяет корректность:

- заголовков ОС-файлов;
- списка сегментов;
- контрольной суммы каждого сегмента.

Первоначальная установка ОС или части ОС выполняется командой **optinstall,** а модификация версии ОС — командой **optupdate.** Эти команды следует выполнять в однопользовательском режиме и завершать перезагрузкой системы.

Данные для установки или модификации версии ОС расположены на дистрибутивном томе. Параметром обеих команд является имя (номер) устанавливаемого или модифицируемого программного продукта.

Наконец, имеется возможность загружать несколько разных ОС (или версий одной ОС), используя одну загрузочную область на системном диске **(/dev/rhd).** Этот вариант загрузки выполняет команда **/lbin/chsys** (change system), являющаяся командным файлом. Внутри командного файла **chsys** используются команды **oscp** — для перестройки загрузочной области для новой ОС чтением ее сегментов из обычных файлов и **osck** — для последующей проверки загрузочной области.

Так как команда **chsys** не проверяет, все ли пользователи закончили работу, рекомендуется перед ее выполнением выполнить команду **shutdown.**

*Переконфигурирование операционной системы.* Параметризуемость операционной системы позволяет оптимальным образом настроить ее для работы на заданных аппаратных средствах и с учетом

особенностей использования системы для заданного класса задач. Совокупность значений технических параметров ОС в загрузочной области называют конфигурацией ОС.

Для управления конфигурацией имеется команда **/lbin/uconfig**. Ее необязательный параметр указывает спецфайл устройства, на котором находится загрузочная область (по умолчанию — **/dev/rhd**). Будучи поданной без опций, команда **uconfig** показывает текущее значение параметров. Опция -Г <имя_файла> позволяет установить новые значения нескольких или всех параметров из указанного файла, а опция -d (default) устанавливает всем параметрам значения по умолчанию, используя файл **/etc/uconfigtab.**

Системными параметрами являются (в скобках указаны значения по умолчанию):

- устройство виртуальной памяти (системный диск);
- размер буфера кэш-памяти (1024 байта);
- число буферов кэш-памяти (0, вычисляется динамически);
- длина цепочки буферов чтения (0, вычисляется динамически);
- время активности процесса после интерактивного чтения;
- время резидентности сегмента в памяти перед свопингом на диск (0, вычисляется динамически);
- размер страницы (1024 байта);
- время резидентности страницы в памяти перед свопингом на диск;
- максимальный размер страничного пула виртуальной памяти (О, вычисляется динамически);
- число страниц буфера дисплея, где страница равна 24 строкам дисплея;
- максимальный размер стековой памяти (0, вычисляется динамически);
- минимальная доля страниц в рабочем множестве страниц;
- максимальное число процессов одного пользователя (500).

При выполнении команды **uconfig** для изменения системных параметров нужно быть уверенным, что ОС в загрузочной области совпадает с текущей ОС. В противном случае, результаты выполнения команды будут непредсказуемы.

*Загрузка и выключение системы.* Загрузка осуществляется, когда на ЭВМ только что включили питание. Обычно процесс загрузки в большей или меньшей степени автоматизирован и заключается в последовательном вызове программ, каждая из которых загружает и запускает следующую. Первая программа, самая простая, загружена всегда и запускается при включении питания ЭВМ автоматически или вручную. В процессе загрузки может потребоваться ответить на

вопросы системы, например, касающиеся устройства, на котором находится загружаемая система. В завершение процесса загрузки система выполняет командный файл **/etc/rc,** который, вообще говоря, может содержать любые команды, но обычно содержит команды для выполнения следующих действий:

. демонтаж старых файловых систем;

. монтаж новых файловых систем;

. удаление старых журнальных учетных файлов;

. удаление временных файлов;

• запуск процессов **update и cron.**

Выключение многопользовательской системы производится выполнением командного файла **/etc/shutdown,** который обычно выполняет следующие действия:

• посылает предупреждающие сообщения всем активным пользователям;

• уничтожает все процессы, кроме процесса консоли;

• очищает все буферы обменов с файлами;

• демонтирует файловые системы;

• выполняет, если нужно, процедуры копирования данных;

• выключает питание ЭВМ, если это позволяет аппаратура, в противном случае, питание выключается вручную.

***Файловые системы.*** Файловая система имеет иерархическую руктуру каталогов и файлов, включая корневой каталог. Файловая система располагается на устройстве, которое, как правило, является магнитным диском того или иного типа. Если диск достаточно велик, он может быть разбит на несколько логических дисков, тогда на каждом логическом диске может быть размещена отдельная файловая система [2, 12].

Каждая файловая система, прежде чем стать доступной, должна быть смонтирована. Количество файлов в файловой системе ограничено (65536 для UNIX версии 7).

*Структура файловой системы.* Каждая файловая система имеет четыре основные части:

• загрузочный блок — это самый первый блок диска (блок 0), зарезервированный для системной загрузочной программы;

• суперблок — это первый блок собственно файловой системы (блок 1), он содержит основные данные файловой системы и ее размещении на диске, в том числе о списках свободных *i*-узлов и блоков;

• *i*-узлы — это последовательность блоков вслед за суперблоком. Каждый *i*-узел содержит ссылки на блоки. Имеется ровно один *i*-узел для каждого каталога или файла в файловой системе;

- блоки — оставшееся пространство диска занимают блоки, которые содержат либо действительные данные каталогов и файлов (блоки данных), либо ссылки на блоки (косвенные блоки).

Суперблок содержит следующие данные:

- размер дискового пространства, доступного файловой системе (в блоках);
- число блоков, зарегистрированных для *i*-узлов;
- имя файловой системы;
- имя тома;
- время последнего изменения;
- время последнего копирования (Backup);
- ссылку на список свободных блоков;
- ссылку на список свободных *i*-узлов.

Структура файловой системы представлена на рис. 3.7.

Каждый файл (и каталог) в файловой системе представлен *i*-узлом, содержащим указатели на блоки, составляющие файл.

В *i*-узле содержится также информация о правах доступа к файлу, число ссылок на файл из каталогов и другие данные.

Каждый *i*-узел содержит 13 указателей. Первые 10 указателей непосредственно ссылаются на блоки данных файла. Поскольку

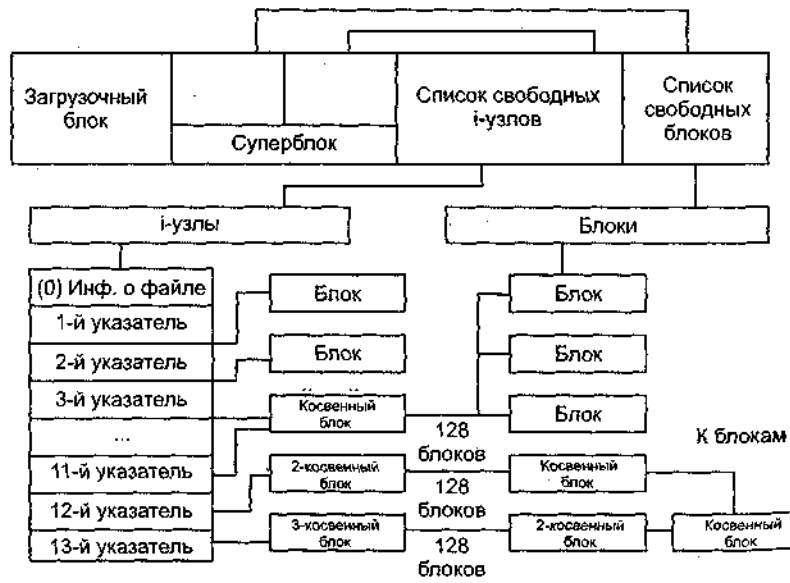



Рис. 3.7. Физическая структура файловой системы на примере ОС Unix

блок содержит 512 байтов, то этого достаточно для обработки файлов до 512 x 10 = 5120 байтов.

Если длина файла больше, чем 5120 байтов, используется 11-й указатель *i*-узла, который ссылается на косвенный блок из 128 ссылок на блоки данных. Использование косвенного блока позволяет увеличить длину файла до величины 512× (10 + 128) = 70656 байтов.

Если и этого недостаточно, то используется 12-й указатель *i*-узла, ссылающийся на дважды косвенный блок, содержащий 128 ссылок на косвенные блоки (рис. 3.7). Тогда максимальный размер файла увеличивается до величины 512 x (10 + 128 + $128^2$) = 8459264 байта.

Наконец, использование последнего, 13-го указателя на трижды косвенный блок из 128 ссылок на дважды косвенные блоки, дает предельную длину в файловой системе:

512 x (10 + 128 + $128^2$ + $128^3$) = 1082201088 байтов.

Другие версии системы могут отличаться количеством ссылок в *i*-узле, косвенных блоках и размером блока данных.

Когда система загружается, имеется только одна из файловых систем, называемая корневой. В ней находятся все важнейшие каталоги (**/dev, /etc, /bin** и пр.). Все остальные файловые системы должны быть созданы и смонтированы.

*Создание и монтаж файловой системы.* Команда **mkfs** создает новую файловую систему. Она расположена в каталоге **/etc** и имеет два параметра:

**/etc/mkfs <имя>** <размер>

Первый параметр является именем специального файла и указывает устройство, на котором создается файловая система. Второй параметр — размер пространства файловой системы в блоках, он используется для определения по некоторым правилам числа блоков после того, как размещены *i*-узлы.

Пример создания файловой системы на флоппи-диске:

```
>/etc/mkfs /dev/flo 2000
isize = 230
```

Ответное сообщение указывает число блоков, выделенное для размещения *i*-узлов. Далее, чтобы сделать файловую систему известной операционной системе, надо ее смонтировать командой **mount.** Эта команда подключает корневой каталог монтируемой файловой системы в один из каталогов корневой файловой системы. Команда Расположена в каталоге **/etc** и имеет два параметра:

**/etc/mount <устройство> <каталог>**

Первый параметр является именем елецфайла для монтируемого логического устройства, содержащего подключаемую файловую систему. Второй — имя уже существующего каталога, под которым монтируется файловая система.

Пример монтажа вновь созданной файловой системы на гибком диске под каталогом, созданным командой **mkdir** в корне корневой файловой системы:

```
>cd/
Mkdir floppy 0
/etc/mount/dev/f10/floppy0
```

Чтобы выяснить, какие файловые системы смонтированы в данный момент, нужно подавать команду **mount** без параметров:

```
>Mount
/ dev/ f10 on / floppy0.
```

Ответом является сообщение об этих системах (в данном случае — одной).

Оно формируется на основе данных о монтаже файловых систем, хранимых в файле

```
/ etc/ mnttab/.
```

Следует заботиться о том, чтобы права доступа корневого каталога монтируемой файловой системы и каталога, под которым производится монтаж, были одинаковыми во избежание ошибок операционной системы.

*Сохранение и восстановление файлов.* Независимо от объема данных в системе важно иметь регулярную процедуру сохранения (копирования) файлов, чтобы обеспечить восстановление в случае их аварийной потери. Возможны различные способы сохранения. Наиболее распространенным является еженедельное полное копирование и ежедневное инкрементное копирование (только изменившихся со времени последнего копирования) файлов. При этом файлы копируются (сбрасываются) на специальное внешнее устройство памяти, обычно магнитную ленту, однако это может быть и «съемный дисковый том, а для малых систем — гибкий диск. На этом устройстве файлы хранятся в специальном архивном формате.

Восстановление утраченных файлов производится путем их поименного копирования из архивной ленты или тома в файловую систему. Обычно таких файлов немного (например, один или два).

В различных реализациях системы могут быть разные команды сброса файлов в архив и восстановления их из архива. Это может быть пара команд **dump и restor,** предназначенных для передачи файлов в архив и обратно. Или это может быть одна команда **cpio**

(или **tcio** для кассетной ленты) с опциями -о или -**i** для сброса в архив и извлечения из архива соответственно.

Наиболее мощным средством сброса в архив в некоторых реализациях служит команда **backup,** являющаяся командным файлом, использующим команды типа **cpio/tcio и fsck.**

Команда **backup** позволяет параметризовать процедуру сброса в архив простым редактированием ее текста, задавая следующие параметры:

- имя каталога сбрасываемой иерархии файлов;
- « имя учетного файла процедуры сброса;
- имя даты последнего сброса;
- имя файла с напоминанием смены архивной ленты (если архив не умещается на одной ленте);
- спецификацию архивного устройства;
- имя учетного файла процедуры проверки файловой системы.

Наконец, для сброса на ленту или гибкий диск и обратного восстановления применяется команда **tar** (tape archive). В отличие от некоторых перечисленных выше команд, она доступна не только администратору системы, но и любому пользователю. Например, для того чтобы сбросить все файлы текущего каталога на гибкий диск, создавая архив впервые (опция с create), нужно выполнить команду:

```
tar cf /dev/f10*
```

Опция **Г (file)** указывает, что следующий параметр является именем спецфайла, соответствующего архивному устройству; * — метасимвол, показывающий, что в архив копируются все файлы.

Для просмотра содержимого архива следует употребить опцию 1 (listing):

```
tar tf /dev/f10
```

Для извлечения из архива указанных файлов нужно выполнить эту же команду с опцией х (extract). Например, для восстановления всех файлов, имена которых оканчиваются на **people,** нужно выполнить команду:

```
tar xf /dev/f10 *people
```

Файлы с этими именами уже должны существовать в текущем каталоге.

В случае, если в текущем каталоге указанных файлов нет, можно восстановить все файлы из архива в указанном каталоге, например:

```
tar xf /dev/f10 тагу,
```

где **mary** – каталог.

*Проверка и восстановление структуры файловой системы*

Структура файловой системы, описанная выше в терминах *i*-узлов, блоков, косвенных блоков и суперблока, может быть нарушена и требовать восстановления. Например, при разрушении информации в трижды косвенном блоке могут появиться следующие проблемы:

- некоторый блок может быть вне системы, т. е. не являться частью файла и не быть в списке свободных блоков;
- могут появиться дубли *i*-узлов, т. е. записи, описывающие один и тот же файл дважды;
- некоторый блок может одновременно быть частью файла и быть в списке свободных блоков;
- некоторый файл может существовать, не будучи включенным ни в один каталог.

Однако структура файловой системы обладает некоторой избыточностью, позволяющей восстанавливать отдельные поломки. Вот некоторые виды избыточности:

- блок данных, являющийся каталогом, содержит имена файлов и номера *i*-узлов, т. е. существует *i*-узел, соответствующий этому каталогу, и этот *i*-узел должен быть каталогом, а не обычным файлом;
- блок, включенный в список свободных блоков, теоретически не может быть частью какого-либо файла. Для проверки этого достаточно сравнить список блоков, занятых файлами, и список свободных блоков;
- блок, принадлежащий файлу, должен принадлежать только одному файлу.

Эти и другие виды избыточности использует программа проверки файловой системы, запускаемая командой **fsck** (File System Check). В различных реализациях существуют разные команды проверки целостности файловой системы: **icheck, dcheck, ncheck** и т. д. Однако их возможности в большей или меньшей степени перекрываются с возможностями команды **fsck.**

Пример выполнения команды **fsck:**

```
/etc/fsck
** Phase 1 - Check Blocks and Sizes
** PHase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** PHase 4 - Check Reference Count
** PHase 5 - Check Free List
```

Из примера видно, что выполнение команды **fsck** производится в несколько этапов.

На этих этапах производится следующая работа:

. проверка целостности *i*-узлов (счетчик связи, тип и формат *i*-узла);

. проверка каталогов, указывающих на *i*-узлы, содержащие ошибки; проверка каталогов, на которые нет ссылок;

• проверка счетчиков связей в каталогах и файлах;

• проверка неверных блоков и дублированных блоков в списках: свободных блоков, неиспользуемых блоков, которые должны быть, но не включены в список свободных блоков и, наконец, счетчика общего числа свободных блоков.

Команда **fsck** по умолчанию всегда проверяет корневую файловую систему. Все другие файловые системы проверяются, если их имена занесены в файл **/etc/checklist.**

Следующий пример показывает действия команды **fsck** и администратора в случае обнаружения дубля *i*-узла для файла **/user/src/sys/ux.** Администратор принимает решение удалить этот плохой файл, отвечая в диалоге согласием y (yes) на вопросы команды.

```
/etc/fsck
** Phase 1 - Check Blocks and Sizes
528627 BAD I=66
** Phase 2 - Check Pathnames
DUP/BAD I=66 OWNER=root MODE=100755
SIZE = 78409 MTIME = jul 16 18:45 1997
FILE=/usr/src/sys/ux
REMOVE ? y
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Count
DUP/BAD I=66 OWNER=root MODE=100755
SIZE = 78409 MTIME = jul 16 18:45 1997
FILE=/usr/src/sys/ux
CLEAR ? y
UNREF FILE I = 36 OWNER = root MODE = 100600
SIZE = О МИМЕ = jul 17 09:40 1997
RECONNECT ? y
** Phase 5 - Check Free List
157 BLK(S) MISSING
BAD FREE LIST
SALVAGE ? y
** Phase 6 - Salvage Free List
302 files 5833 blocks 371 free
```

Обратите внимание на появление шестой фазы, которой не было при нормальном выполнении команды. На этой фазе уточня-

ется и восстанавливается список свободных блоков. После выполнения команды **fsck,** связанного с «починкой» файловой системы может появиться сообщение:

```
***** BOOT UNIX (N0 SYNC!) *****,
```

требующее перезагрузки системы без выполнения команды 8упс.

Если этого не сделать, работа по восстановлению списка свободных блоков будет утрачена, так как копии управляющих таблиц и буфера в оперативной памяти остались старыми. Для их обновления требуется перезагрузка без выгрузки буферов на диск командой **sync.**

Необходимым условием правильной работы команды **fsck** является наличие пустого каталога **/lost+found,** расположенного в корневом каталоге. Если при выполнении команды **fsck** будут найдены каталоги, на. которые никто не ссылается в проверяемой файловой системе, они будут подключены в каталог **/lost+found** для дальнейшего изучения их принадлежности.

*Контроль использования дисковой памяти.* Регулярное выполнение команды **du** (disk usage) позволяет выявить пользователей, захвативших слишком много дисковой памяти. Команда печатает число блоков, занятых каждым файлом и каталогом в дереве, указанном параметром команды (именем каталога).

Пример:

```
du /
```

В результате исполнения этой команды будут выведены объемы всех файлов и каталогов.

Использование команды **find** помогает выявить долго не используемые файлы большого объема. Например, для того чтобы выявить все файлы, к которым не обращались ни по записи, ни по чтению последние 90 дней, следует ввести команду:

```
find / -т time+90 -a tirae+90 -print
```

С помощью команды **find** администратор может также найти файлы, представляющие опасность для операционной системы или бесполезно занимающие пространство на диске, даже если место расположения этих файлов в иерархии каталогов неизвестно, например:

```
find / -пате дапдег -print
```

При этом поиск производится, начиная с корневого каталога /. Ключ **-пате** указывает последующее имя файла **danger,** а ключ -P**rint** предписывает вывод полного имени файла **danger** на экран.

Команда **df** (disk free) показывает число всех свободных блоков или, если она явно указана, — определенной файловой системы.

Следует помнить, что учетные файлы (типа **/usr/lib/cronlog**) могут расти неограниченно и требуют периодической чистки или сброса в архив.

***Работа с руководствами для пользователя.*** Тексты руководств для пользователей находятся в различных подкаталогах **man?** каталога **/usr/man,** где вопросительный знак — метасимвол, принимающий значения от 1 до 8, в соответствии с нумерацией руководств по системе. Файлы этих подкаталогов содержат исходные тексты отдельных руководств. Например, подкаталог man1, содержащий файлы текстов команд, может иметь следующий вид (фрагмент):

```
> ls /usr/man/man1
...
cpio.1 grep.1 mknod.1 ren.1 test.1
...
```

Исходные тексты руководств хранятся в этих файлах в виде, подготовленном для команды форматирования **nroff** (newrunoff). На выходе команды форматирования появится текст в формате, пригодном для выдачи на печать или терминал. Во многих системах текст руководства в выходном формате создается и запоминается в файлах каталога /usr/man/cat?/*, где ? и * — метасимволы в обычном смысле, выполнением команды **catman**.

Выдача руководства на терминал или печать выполняется командой man, которая в соответствии с указанным аргументом ищет в начале текст руководства в выходном формате и выдает его на стандартный вывод. В противном случае, она ищет исходный текст и вызывает команду nroff, результат работы которой поступает на стандартный вывод. Добавление новых руководств требует знания форматов для команды nroff, работающей с пакетом man-макроопределений.

*Структура информации о функциях Unix.* Руководство UNIX Reference Manual содержит 8 разделов:

1. Commands (команды).
2. System calls (системные вызовы).
3. Subroutines (подпрограммы).
4. Special files (спецфайлы).
5. File format and conversion (формат файлов и соглашения).
6. Games (игры).
7. Macro packages and language Conventions (макропакеты и языковые соглашения).

8. Administrator commands and procedures (команды и процедуры администратора).

Описание команды состоит из следующих разделов:

NAME (имя и функция);
SYNOPSIS(синтаксис);
DESCRIPTION (описание функции);
FILE (используемые файлы);
8EE ALSO (смежные команды);
DIAGNOSTIC (реакция на ошибки);
BUGS (замеченные некорректности).

*Ядро ОС UNIX.* Как и в любой другой многопользовательской операционной системе, обеспечивающей защиту пользователей друг от друга и защиту системных данных от любого непривилегированного пользователя, в ОС UNIX имеется защищенное ядро, которое управляет ресурсами компьютера и предоставляет пользователям базовый набор услуг.

Следует заметить, что удобство и эффективность современных вариантов ОС UNIX не означает, что вся система, включая ядро, спроектирована и структуризована наилучшим образом. Как мы показали в первой части курса, ОС UNIX развивалась на протяжении многих лет (это первая в истории операционная система, которая продолжает завоевывать популярность в таком зрелом возрасте — уже больше 25 лет). Естественно, наращивались возможности системы, и, как это часто бывает в больших системах, качественные улучшения структуры ОС UNIX не поспевали за ростом ее возможностей.

В результате ядро большинства современных коммерческих вариантов ОС UNIX (как мы отмечали ранее, почти все они основаны на UNIX System V) представляет собой не очень четко структурированный монолит большого размера. По этой причине программирование на уровне ядра ОС UNIX продолжает оставаться искусством (если не считать отработанной и понятной технологии разработки драйверов внешних устройств). Эта недостаточная технологичность организации ядра ОС UNIX многих не удовлетворяет. Отсюда стремление к полному воспроизведению среды ОС UNIX при полностью иной организации системы (в частности, с применением микроядерного подхода, который мы кратко рассмотрим в конце курса).

*Общая организация традиционного ядра ОС UNIX.* Одно из основных достижений ОС UNIX состоит в том, что система обладает свойством высокой мобильности. Смысл этого качества состоит в том, что вся операционная система, включая ее ядро, сравнительно просто переносится на различные аппаратные платформы. Все час-

ти системы, не считая ядра, являются полностью машинно-независимыми. Эти компоненты аккуратно написаны на языке Си, и для их переноса на новую платформу (по крайней мере, в классе 32-разрядных компьютеров) требуется только перекомпиляция исходных текстов в коды целевого компьютера.

Конечно, наибольшие проблемы связаны с ядром системы, которое полностью скрывает специфику используемого компьютера, но само зависит от этой специфики. В результате продуманного разделения машинно-зависимых и машинно-независимых компонентов ядра (видимо, с точки зрения разработчиков операционных систем, в этом состоит наивысшее достижение разработчиков традиционного ядра ОС UNIX) удалось добиться того, что основная часть ядра не зависит от архитектурных особенностей целевой платформы, написана полностью на языке Си и для переноса на новую платформу нуждается только в перекомпиляции.

Однако сравнительно небольшая часть ядра является машинно-зависимой и написана на смеси языка Си и языка ассемблера целевого процессора. При переносе системы на новую платформу требуется переписывание этой части ядра с использованием языка ассемблера и учетом специфических черт целевой аппаратуры.

Машинно-зависимые части ядра хорошо изолированы от основной машинно-независимой части, и при хорошем понимании назначения каждого машинно-зависимого компонента переписывание машинно-зависимой части является в основном технической задачей (хотя и требует высокой программистской квалификации).

Машинно-зависимая часть традиционного ядра ОС UNIX включает следующие компоненты:

- раскрутки и инициализации системы на низком уровне (пока это зависит от особенностей аппаратуры);
- первичной обработки внутренних и внешних прерываний;
- управления памятью (в той части, которая относится к особенностям аппаратной поддержки виртуальной памяти);
- переключения контекста процессов между режимами пользователя и ядра;
- связанные с особенностями целевой платформы части драйверов устройств.

*Основные функции.* К основным функциям ядра ОС UNIX принято относить следующие:

- инициализации системы — функция запуска и раскрутки. Ядро системы обеспечивает средство раскрутки (bootstrap), которое обеспечивает загрузку полного ядра в память компьютера и запускает ядро;

- управления процессами и нитями — функция создания, завершения и отслеживания существующих процессов и нитей («процессов», выполняемых на обшей виртуальной памяти) Поскольку ОС UNIX является мультипроцессной операционной системой, ядро обеспечивает разделение между запущенными процессами времени процессора (или процессоров в мультипроцессорных системах) и другими ресурсами компьютера для создания внешнего ощущения того, что процессы реально выполняются в параллель;
- управления памятью — функция отображения практически неограниченной виртуальной памяти процессов в физическую оперативную память компьютера, которая имеет ограниченные размеры. Соответствующий компонент ядра обеспечивает разделяемое использование одних и тех же областей оперативной памяти несколькими процессами с использованием внешней памяти;
- управления файлами — функция, реализующая абстракцию файловой системы, — иерархии каталогов и файлов. Файловые системы ОС UNIX поддерживают несколько типов файлов. Некоторые файлы могут содержать данные в формате ASCII, другие будут соответствовать внешним устройствам. В файловой системе хранятся объектные файлы, выполняемые файлы и т. д. Файлы обычно хранятся на устройствах внешней памяти; доступ к ним обеспечивается средствами ядра. Б мире UNIX существует несколько типов организации файловых систем. Современные варианты ОС UNIX одновременно поддерживают большинство типов файловых систем;
- коммуникации и обеспечения возможности обмена данными между процессами, выполняющимися внутри одного компьютера (IPC — Inter-Process Communications), между процессами, выполняющимися в разных узлах локальной или глобальной сети передачи данных, а также между процессами и драйверами внешних устройств;
- программного интерфейса — функция, обеспечивающая доступ к возможностям ядра со стороны пользовательских процессов на основе механизма системных вызовов, оформленных в виде библиотеки функций.

*Принципы взаимодействия с ядром.* В любой операционной системе поддерживается некоторый механизм, который позволяет пользовательским программам обращаться за услугами ядра *ОС.* В операционных системах наиболее известной советской вычислительной машины БЭСМ-6 соответствующие средства общения с яд-

ром назывались экстракодами, в операционных системах IBM они назывались системными макрокомандами и т. д. В ОС UNIX такие средства называются системными вызовами.

Название не изменяет смысл, который состоит в том, что для обращения к функциям ядра ОС используются «специальные команды» процессора, при выполнении которых возникает особого рода внутреннее прерывание процессора, переводящее его в режим ядра (в большинстве современных ОС этот вид прерываний называется trap — ловушка). При обработке таких прерываний (дешифрации) ядро ОС распознает, что на самом деле прерывание является запросом к ядру со стороны пользовательской программы на выполнение определенных действий, выбирает параметры обращения и обрабатывает его, после чего выполняет «возврат из прерывания», возобновляя нормальное выполнение пользовательской программы.

Понятно, что конкретные механизмы возбуждения внутренних прерываний по инициативе пользовательской программы различаются в разных аппаратных архитектурах. Поскольку ОС UNIX стремится обеспечить среду, в которой пользовательские программы могли бы быть полностью мобильны, потребовался дополнительный уровень, скрывающий особенности конкретного механизма возбуждения внутренних прерываний. Этот механизм обеспечивается так называемой библиотекой системных вызовов.

Для пользователя библиотека системных вызовов представляет собой обычную библиотеку заранее реализованных функций системы программирования языка Си. При программировании на языке Си использование любой функции из библиотеки системных вызовов ничем не отличается от использования любой собственной или библиотечной Си-функции. Однако внутри любой функции конкретной библиотеки системных вызовов содержится код, являющийся, вообще говоря, специфичным для данной аппаратной платформы.

*Принципы обработки прерываний.* Конечно, применяемый в операционных системах механизм обработки внутренних и внешних прерываний в основном зависит от того, какая аппаратная поддержка обработки прерываний обеспечивается конкретной аппаратной платформой. К счастью, к настоящему моменту (и уже довольно давно) основные производители компьютеров де-факто пришли к соглашению о базовых механизмах прерываний.

Если говорить не очень точно и конкретно, суть принятого на сегодня механизма состоит в том, что каждому возможному прерыванию процессора (будь то внутреннее или внешнее прерывание) соответствует некоторый фиксированный адрес физической оперативной памяти. В тот момент, когда процессору разрешается пре-

рваться по причине наличия внутренней или внешней заявки на прерывание, происходит аппаратная передача управления на ячейку физической оперативной памяти с соответствующим адресом — обычно адрес этой *ячейки* называется «вектором прерывания» (как правило, заявки на внутреннее прерывание, т. е. заявки, поступающие непосредственно от процессора, удовлетворяются немедленно).

Дело операционной системы — разместить в соответствующих ячейках оперативной памяти программный код, обеспечивающий начальную обработку прерывания и инициирующий полную обработку.

В основном ОС UNIX придерживается общего подхода. В векторе прерывания, соответствующем внешнему прерыванию, т. е. прерыванию от некоторого внешнего устройства, содержатся команды, устанавливающие уровень выполнения процессора (уровень выполнения определяет, на какие внешние прерывания процессор должен реагировать незамедлительно) и осуществляющие переход на программу полной обработки прерывания в соответствующем драйвере устройства. Для внутреннего прерывания (например, прерывания по инициативе программы пользователя при отсутствии в основной памяти нужной страницы виртуальной памяти, при возникновении исключительной ситуации в программе пользователя и т. д.) или прерывания от таймера в векторе прерывания содержится переход на соответствующую программу ядра ОС UNIX.

***Управление устройствами.*** Управление внешними устройствами — это одна из важнейших функций любой операционной системы. Система должна обеспечивать эффективный и удобный доступ к периферийным устройствам, а также обеспечивать возможность унифицированной разработки программного обеспечения для вновь подключаемых внешних устройств. Рассмотрим, как эта проблема решается в ОС UNIX.

*Устройство как специальный файл.* Для доступа к внешним устройствам в ОС UNIX используется универсальная абстракция файла. Помимо настоящих файлов (обычных файлов или каталогов), которые реально занимают память на магнитных дисках, файловая система содержит так называемые специальные файлы, для которых, как и для настоящих файлов, отводятся отдельные *i*-узлы, но которым на самом деле соответствуют внешние устройства. Это решение позволяет естественным образом работать в одном и том же интерфейсе с любым файлом или внешним устройством.

*Драйверы устройств.* Очевидно, что простое объявление внешнего устройства специальным файлом не даст возможности работать с этим устройством, если не создан и соответствующим образом не

подключен к системе специальный программный код, соответствующий специфике данного устройства. Как и в большинстве современных операционных систем, такого рода программный код в ОС UNIX называется драйвером устройства (в этом контексте слово драйвер лучше всего понимать в значении «управляющий»).

Для профессионалов в области операционных систем драйверы ОС UNIX, в сущности, не представляют ничего нового. По-простому говоря, в любой системе драйвер устройства — это многовходовой программный модуль со своими статическими данными, который умеет инициировать работу с устройством, выполнять заказываемые пользователем обмены (на ввод или вывод данных), завершать работу с устройством и обрабатывать прерывания от устройства. Однако в любой операционной системе имеется своя технология разработки драйверов. В частности, в ОС UNIXразличаются символьные, блочные и потоковые драйверы. Символьные драйверы являются простейшими в ОС UNIX и предназначаются для обслуживания устройств, которые реально ориентированы на прием или выдачу произвольных последовательностей байтов (например, простой принтер или устройство ввода с перфоленты). Такие драйверы используют минимальный набор стандартных функций ядра UNIX, которые главным образом заключаются в возможности взять данные из виртуального пространства пользовательского процесса и/или поместить данные в такое виртуальное пространство.

Блочные драйверы более сложны. Они работают с использованием возможностей системной буферизации блочных обменов ядра ОС UNIX. В число функций такого драйвера входит включение соответствующего блока данных в систему буферов ядра ОС UNIX и/или взятие содержимого буферной области в случае необходимости.

Наконец, наиболее сложной организацией отличаются потоковые драйверы. Фактически, такой драйвер представляет собой конвейер модулей, обеспечивающий многоступенчатую обработку запросов пользователя. Потоковые драйверы в среде ОС UNIX в основном предназначены для реализации доступа к сетевым устройствам, которые должны работать в соответствии с многоуровневыми сетевыми протоколами.

Как и в большинстве развитых операционных систем, в ОС UNIX возможны два способа включения драйвера в состав ядра ОС. Первый способ состоит в полном включении драйвера в состав ядра на стадии генерации системы (т. е. драйвер статически объявляется частью ядра системы).

Второй способ позволяет обойтись минимальным количеством статических объявлений на стадии генерации ядра (фактически обеспечиваются лишь необходимые элементы статических таблиц). В любой момент работы системы такой драйвер может быть динамически загружен в ядро системы. После появления (статического или динамического) в ядре ОС UNIX драйверы всех разновидностей функционируют единообразно.

***Управление процессами и нитями.*** В операционной системе UNIX традиционно поддерживается классическая схема мультипрограммирования. Система поддерживает возможность параллельного (или квазипараллельного в случае наличия только одного аппаратного процессора) выполнения нескольких пользовательских программ. Каждому такому выполнению соответствует процесс операционной системы. Каждый процесс выполняется в собственной виртуальной памяти, и, тем самым, процессы защищены один от другого, т. е. один процесс не в состоянии неконтролируемым образом прочитать что-либо из памяти другого процесса или записать в нее. Однако контролируемые взаимодействия процессов допускаются системой, в том числе за счет возможности разделения одного сегмента *памяти* между виртуальной памятью нескольких процессов.

Конечно, не менее важно (а на самом деле, существенно более важно) защищать саму операционную систему от возможности ее повреждения каким бы то ни было пользовательским процессом. В ОС UNIX это достигается за счет того, что ядро системы работает в собственном «ядерном» виртуальном пространстве, к которому не может иметь доступа ни один пользовательский процесс.

Ядро системы предоставляет возможности (набор системных вызовов) для порождения новых процессов, отслеживания окончания порожденных процессов и т. д. С другой стороны, в ОС UNIX ядро системы — это полностью пассивный набор программ и данных. Любая программа ядра может начать работать только по инициативе некоторого пользовательского процесса (при выполнении системного вызова), либо по причине внутреннего или внешнего прерывания (примером внутреннего прерывания может быть прерывание из-за отсутствия в основной памяти требуемой страницы виртуальной памяти пользовательского процесса; примером внешнего прерывания является любое прерывание процессора по инициативе внешнего устройства). В любом случае считается, что выполняется ядерная часть обратившегося или прерванного процесса, т. е. ядро всегда работает в контексте некоторого процесса.

В последние годы в связи с широким распространением так называемых симметричных мультипроцессорных архитектур компьютеров (Symmetric Multiprocessor Architectures — SMP) в ОС UNIX был внедрен механизм легковесных процессов (light-weight processes), или нитей, или потоков управления (threads). Можно сказать, что нить — это процесс, выполняющийся в виртуальной памяти, используемой совместно с другими нитями одного и того же «тяжеловесного» (т. е. обладающего отдельной виртуальной памятью) процесса. В принципе легковесные процессы использовались в операционных системах много лет назад. Уже тогда стало ясно, что программирование с неконтролируемым использованием общей памяти приносит больше хлопот и неприятностей, чем пользы, по причине необходимости использования явных примитивов синхронизации.

Однако до настоящего времени в практику программистов так и не были внедрены более безопасные методы параллельного программирования, а реальные возможности мультипроцессорных архитектур для обеспечения распараллеливания нужно было как-то использовать. Поэтому опять в обиход вошли легковесные процессы, которые теперь получили название threads (нити). Наиболее важно (с нашей точки зрения) то, что для внедрения механизма нитей потребовалась существенная переделка ядра. Разные производители аппаратуры и программного обеспечения стремились как можно быстрее выставить на рынок продукт, пригодный для эффективного использования на SMP-платформах. Поэтому версии ОС UNIX опять несколько разошлись.

*Пользовательская и ядерная составляющие процессов.* Каждому процессу соответствует контекст, в котором он выполняется. Этот контекст включает содержимое пользовательского адресного пространства — пользовательский контекст (т. е. содержимое сегментов программного кода, данных, стека, разделяемых сегментов и сегментов файлов, отображаемых в виртуальную память), содержимое аппаратных регистров — регистровый контекст (регистр счетчика Команд, регистр состояния процессора, регистр указателя стека и Регистры общего назначения), а также структуры данных ядра (контекст системного уровня), связанные с этим процессом. Контекст процесса системного уровня в ОС UNIX состоит из «статической» и «динамических» частей. Для каждого процесса имеется одна статическая часть контекста системного уровня и переменное число динамических частей.

*Статическая часть контекста процесса* системного уровня включает следующее.

Описатель процесса, т. е. элемент таблицы описателей существующих в системе процессов. Описатель процесса включает, в частности, следующую информацию:

- состояние процесса;
- физический адрес в основной или внешней памяти *и-области процесса;*
- идентификаторы пользователя, от имени которого запущен процесс;
- идентификатор процесса;
- прочую информацию, связанную с управлением процессом.

*U-область (u-area)* — индивидуальная для каждого процесса область пространства ядра, обладающая тем свойством, что хотя и-область каждого процесса располагается в отдельном месте физической памяти, и-области всех процессов имеют один и тот же виртуальный адрес в адресном пространстве ядра. Именно это означает, что какая бы программа ядра не выполнялась, она всегда выполняется как ядерная часть некоторого пользовательского процесса и именно того процесса, u-область которого является «видимой» для ядра в данный момент времени. U-область процесса содержит:

- указатель на описатель процесса;
- идентификаторы пользователя;
- счетчик времени, в течение которого процесс реально выполнялся (т. е. занимал процессор) в режиме пользователя и режиме ядра;
- параметры системного вызова;
- результаты системного вызова;
- таблицу дескрипторов открытых файлов;
- предельные размеры адресного пространства процесса;
- предельные размеры файла, в который процесс может писать;

и т. д.

*Динамическая часть контекста процесса* — это один или несколько стеков, которые используются процессом при его выполнении в режиме ядра. Число ядерных стеков процесса соответствует числу уровней прерывания, поддерживаемых конкретной аппаратурой.

***Принципы организации многопользовательского режима.*** Основной проблемой организации многопользовательского (правильнее сказать, мультипрограммного) режима в любой операционной системе является организация планирования «параллельного» выполнения нескольких процессов. Операционная система должна обладать четкими критериями для определения того, какому готовому к выполнению процессу и когда предоставить ресурс процессора. Исторически ОС UNIX является системой разделения вре-

мени, т. е. система должна прежде всего «справедливо» разделять ресурсы процессора(ов) между процессами, относящимися к разным пользователям, причем таким образом, чтобы время реакции каждого действия интерактивного пользователя находилось в допустимых пределах. Однако в последнее время возрастает тенденция к использованию ОС UNIX в приложениях реального времени, что повлияло и на алгоритмы планирования.

Ниже мы опишем общую (без технических деталей) схему планирования разделения ресурсов процессора(ов) между процессами в UNIX System V Release 4.

Наиболее распространенным алгоритмом планирования в системах разделения времени является кольцевой режим (round robin). Основной смысл алгоритма состоит в том, что время процессора делится на кванты фиксированного размера, а процессы, готовые к выполнению, выстраиваются в кольцевую очередь. У этой очереди имеются два указателя — начала и конца. Когда процесс, выполняющийся на процессоре, исчерпывает свой квант процессорного времени, он снимается с процессора, ставится в конец очереди, а ресурсы процессора отдаются процессу, находящемуся в начале очереди. Если выполняющийся на процессоре процесс откладывается (например, по причине обмена с некоторым внешним устройством) до того как он исчерпает свой квант, то после повторной активизации он становится в конец очереди (не смог доработать — не вина системы). Это прекрасная схема разделения времени в случае, когда все процессы одновременно помещаются в оперативной памяти.

Однако ОС UNIX всегда была рассчитана на то, чтобы обслуживать больше процессов, чем можно одновременно разместить в основной памяти. Другими словами, часть процессов, потенциально готовых выполняться, размещалась во внешней памяти (куда образ памяти процесса попадал в результате свопинга). Поэтому требовалась несколько более гибкая схема планирования разделения ресурсов процессора(ов). В результате было введено понятие приоритета. В ОС UNIX значение приоритета определяет, во-первых, возможность процесса пребывать в основной памяти и на равных конкурировать за процессор. Во-вторых, от значения приоритета процесса, вообще говоря, зависит размер временного кванта, который предоставляется процессу для работы на процессоре при достижении своей очереди. В-третьих, значение приоритета влияет на место процесса в общей очереди процессов к ресурсу процессора(ов).

Традиционное решение ОС UNIX состоит в использовании динамически изменяющихся приоритетов. Каждый процесс при своем образовании получает некоторый устанавливаемый системой стати-

ческий приоритет, который в дальнейшем может быть изменен с помощью системного вызова **nice**. Этот статический приоритет является основой начального значения динамического приоритета процесса, являющегося реальным критерием планирования. Все процессы с динамическим приоритетом не ниже порогового участвуют в конкуренции за процессор (по схеме, описанной выше). Однако каждый раз, когда процесс успешно отрабатывает свой квант на процессоре, его динамический приоритет уменьшается (величина уменьшения зависит от статического приоритета процесса). Если значение динамического приоритета процесса достигает некоторого нижнего предела, он становится кандидатом на откачку (свопинг) и больше не конкурирует за процессор.

Процесс, образ памяти которого перемещен во внешнюю память, также обладает динамическим приоритетом. Этот приоритет не дает процессу право конкурировать за процессор (да это и невозможно, поскольку образ памяти процесса не находится в основной памяти), но он изменяется, давая в конце концов процессу возможность вновь вернуться в основную память и принять участие в конкуренции за процессор.

Правила изменения динамического приоритета для процесса, перемещенного во внешнюю память, в принципе, очень просты. Чем дольше образ процесса находится во внешней памяти, тем более высок его динамический приоритет (конкретное значение динамического приоритета, конечно, зависит от его статического приоритета).

Конечно, раньше или позже значение динамического приоритета такого процесса перешагнет через некоторый порог, и тогда система принимает решение о необходимости возврата образа процесса в основную память. После того как в результате свопинга будет освобождена достаточная по размерам область основной памяти, процесс с приоритетом, достигшим критического значения, будет перемещен в основную память и будет в соответствии со своим приоритетом конкурировать за процессор.

*Что понимается под концепцией «реального времени» в ОС UNIX.* Известно, что существуют по крайней мере два понимания термина «мягкое реальное время (soft realtime)» и « жесткое реальное время (hard realtime)».

Жесткое реальное время означает, что каждое событие (внутреннее или внешнее), происходящее в системе (обращение к ядру системы за некоторой услугой, прерывание от внешнего устройства и т. д.), должно обрабатываться системой за время, не превосходящее верхнего предела времени, отведенного для таких действий. Режим жесткого реального времени требует задания четких временны

характеристик процессов, и эти временные характеристики должны определять поведение планировщика распределения ресурсов процессора(ов) и основной памяти.

Режим мягкого реального времени, в отличие от этого, предполагает, что некоторые процессы (процессы реального времени) получают права на получение ресурсов основной памяти и процессора(ов), существенно превосходящие права процессов, не относящихся к категории процессов реального времени. Основная идея состоит в том, чтобы дать возможность процессу реального времени опередить в конкуренции за вычислительные ресурсы любой другой процесс, не относящийся к категории процессов реального времени. Отслеживание проблем конкуренции между различными процессами реального времени относится к функциям администратора системы.

В своих самых последних вариантах ОС UNIX поддерживает концепцию мягкого реального времени. Это делается способом, не выходящим за пределы основополагающего принципа разделения времени. Как мы отмечали выше, имеется некоторый диапазон значений статических приоритетов процессов. Некоторый поддиапазон этого диапазона включает значения статических приоритетов процессов реального времени. Процессы, обладающие динамическими приоритетами, основанными на статических приоритетах процессов реального времени, обладают следующими особенностями:

- каждому из таких процессов предоставляется неограниченный сверху квант времени на процессоре. Другими словами, занявший процессор процесс реального времени не будет с него снят до тех пор, пока сам не заявит о невозможности продолжения выполнения (например, задав обмен с внешним устройством);
- процесс реального времени не может быть перемещен из основной памяти во внешнюю, если он готов к выполнению и в оперативной памяти присутствует хотя бы один процесс, не относящийся к категории процессов реального времени (т. е. процессы реального времени перемещаются во внешнюю память последними, причем в порядке убывания своих динамических приоритетов);
- любой процесс реального времени, перемещенный во внешнюю память, но готовый к выполнению, переносится обратно в основную память, как только в ней образуется свободная область соответствующего размера (выбор процесса реального времени для возвращения в основную память производится на основании значений динамических приоритетов).

## 3.4. Операционная система LINUX, графическая оболочка X Window

Linux — свободно распространяемая версия UNIX, первоначально была разработана Линусом Торвальдсом (Linus Torvalds). Linux был создан с помощью многих UNIX-программистов и энтузиастов из Internet, тех, кто имеет достаточно навыков и способностей развивать систему. Ядро Linux не использует коды AT&T или какого-либо другого частного источника, и большинство программ Linux разработаны в рамках проекта GNU из Free Software Foundation в Cambridge, Massachusetts. Но в него внесли лепту также программисты всего мира [7].

5 октября 1991 года Линус объявил первую «официальную» версию Linux, версия 0.02. Основное внимание уделялось созданию ядра. Никакие вопросы поддержки работы с пользователем, документирования, тиражирования и т. п. даже не обсуждались. Да и сегодня сообщество Linux-истов считает эти вопросы вторичными по сравнению с «настоящим программированием» — развитием ядра.

После версии 0.03 Линус скачком перешел в нумерации к версии 0.10, так как над проектом стало работать много специалистов и любителей. После нескольких последовавших пересмотров версий Линус присвоил очередной версии номер 0.95, чтобы тем самым отразить свое впечатление о том, что скоро возможна уже «официальная» версия. Это было в марте 1992 года. Примерно через полтора года — в декабре 1993 года версия ядра все еще была Linux 0.99.pl14 — асимптотически приближаясь к 1.0. А на данный момент версия ядра — 1.2..

Сегодня Ыпих — это полноценная ОС семейства UNIX, способная работать с X Windows, TCP/IP, Emacs, UUCP, mail и USENET. Практически все важнейшие программные пакеты были поставлены и на Linux, т. е. для Linux теперь доступны и коммерческие пакеты. Все большее разнообразие оборудования поддерживается по сравнению с первоначальным ядром.

Ыпих — многозадачная и многопользовательская операционная система для бизнеса, образования и индивидуального программирования. Ыпих принадлежит семейству UNIX-подобных операционных систем, которая может работать на компьютерах Intel 803δό, 80486 и Pentium. Рекомендуемые конфигурации компьютеров:

минимум — Intel 80386 DX 40MHz/4Mb(RAM)/80Mb(HDD);
рекомендуемое — Pentium 100MHz/16Mb/540Mb;
оптимальное — Pentium 133MHz/32Mb/1Gb.

Linux поддерживает широкий спектр программных пакетов от т.У до X-Windows, компиляторов GNU *C/C++*, протоколов TCP/IP. Это гибкая реализация ОС UNIX, свободно распространяемая под генеральной лицензией GNU.

Linux может любой вышеназванный персональный компьютер превратить в рабочую станцию. Бизнесмены инсталлируют Ыпих в сетях машин, используют операционную систему для обработки данных в сфере финансов, медицины, распределенной обработки, в телекоммуникациях и т. д.

***Системные характеристики.*** Linux — это полная многозадачная многопользовательская операционная система (точно также, как и другие версии UNIX). Ыпих достаточно хорошо совместим с рядом стандартов на уровне исходных текстов, включая ШЕЕ POSIX.1, System V и BSD. Он создавался имея в виду такую совместимость.

Другие специфические внутренние черты Ыпих включают контроль работ по стандарту POSIX (используемый оболочками, такими, как csh и bash), псевдотерминалы (pty), поддержку национальных и стандартных клавиатур динамически загружаемыми драйверами клавиатур.

Ядро может само эмулировать команды 387-FPU, так что системы без сопроцессора могут выполнять программы, ориентированные на него (т. е. с плавающей точкой).

Linux поддерживает различные типы файловых систем для хранения данных. Некоторые файловые системы, такие, как файловая система ext2fs, были созданы специально для Linux. Поддерживаются также другие типы файловых систем, такие, как Minix-1 и Xenix. Реализована также файловая система MS-DOS, позволяющая прямо обращаться к файлам MS-DOS на жестком диске. Поддерживается также файловая система ISO 9660 CD-ROM для работы с дисками CD-ROM.

Linux обеспечивает полный набор протоколов TCP/IP для сетевой работы. Поддерживается весь спектр клиентов и услуг TCP/IP, таких, как PTP, Telnet, NNTP и SMTP.

Ядро Linux сразу создано с учетом специального защищенного режима для процессоров Intel 80386 и 80486. В частности, Linux использует парадигму описания памяти в защищенном режиме и другие новые свойства процессоров.

... увеличения объема доступной памяти Linux осуществляет так е разбиение диска на страницы: то есть на диске может быть выделено До 256 Мбайт «пространства для свопинга» (swap space). (Swap space не совсем подходящее имя, в Ыпих в область свопинга

выгружается не весь процесс, а только отдельные его части, в которых нет необходимости.) Когда системе нужно больше физической памяти, то она с помощью свопинга выводит неактивные страницы на диск. Это позволяет выполнять более объемные программы и обслуживать одновременно больше пользователей. Однако свопинг не исключает наращивания физической памяти, поскольку он снижает быстродействие, увеличивает время доступа.

Ядро также поддерживает универсальный пул памяти для пользовательских программ и дискового кэша. При этом для кэша может использоваться вся память, и, наоборот, кэш уменьшается при работе больших программ.

Выполняемые программы используют динамически связываемые библиотеки, т. е. выполняемые программы могут совместно использовать библиотечную программу, представленную одним физическим файлом на диске (иначе, чем это реализовано в механизме разделяемых библиотек SunOS). Это позволяет выполняемым файлам занимать меньше места на диске, особенно тем, которые многократно используют библиотечные функции. Есть также статические связываемые библиотеки для тех, кто желает пользоваться отладкой на уровне объектных кодов или иметь «полные» выполняемые программы, которые не нуждаются в разделяемых библиотеках. В Linux разделяемые библиотеки динамически связываются во время выполнения, позволяя программисту заменять библиотечные модули своими собственными.

*Совместимость.* Linux представляет собой комбинацию BSD UNIX и System V Release 4 UNIX. Это полная многозадачная многопользовательская операционная система (точно также, как и другие версии UNIX). Linux достаточно хорошо совместим с рядом стандартов на уровне исходных текстов, включая IEEE POSIX.1, ...m V и BSD.

...дня Linux — это полноценная ОС семейства UNIX, способ... тать с X Windows, TCP/IP, Emacs, UUCP, mail и USENET. ...ски все важнейшие программные пакеты были поставлены ... теперь для него доступны и коммерческие пакеты. Сей... льшее разнообразие оборудования поддерживается по ... первоначальным ядром.

***...окон...кий интерфейс.*** В Linux применяется графический ...ий интерфейс (GUI) X Window. Для этого интерфей... много программ управления окнами — менеджеров ... AfterStep, Wfwm, KDE, GNOME. Два последних ...ляют, при желании, сделать Desktop («рабочий ...ким на Desktop Windows 95 (рис. 3.8).

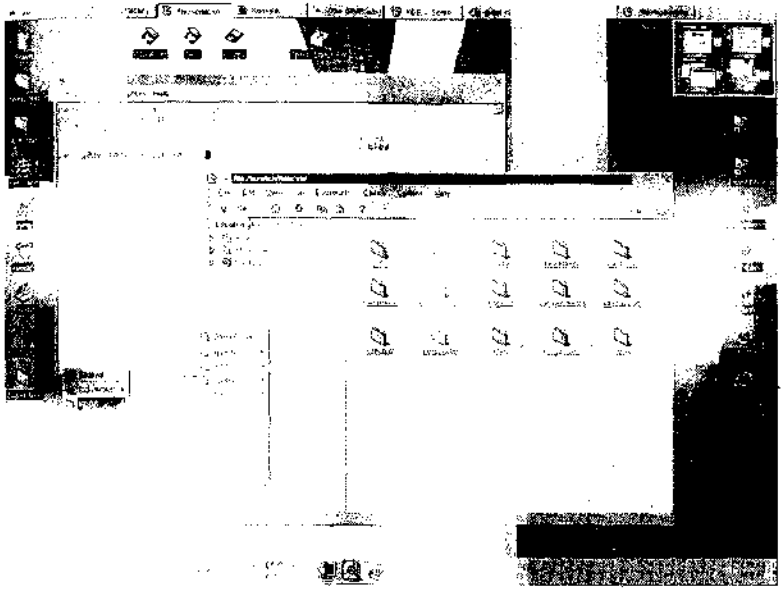


*Рис.* 3.8. Операционная система Linux с KDE

*Поддерживаемые файловые системы.* Linux поддерживает различные типы файловых систем для хранения данных. Некоторые файловые системы, такие, как файловая система ext2fs, были созданы специально для Linux. Поддерживаются и другие типы файловых систем, такие, как Minix-1, Xenix и файловая система ISO 9660 CD-ROM для работы с дисками CD-ROM. Реализована также файловая система M8-BO5, позволяющая прямо обращаться к файлам M5-DOS на жестком диске. Кроме того, имеется возможность выполнять приложения как DOS, так и Windows.

*Сетевые возможности.* Linux обеспечивает полный набор протоколов TCP/IP для работы в сети Internet. Поддерживается весь спектр клиентов и услуг TCP/IP, таких, как PTP, TELNET, NNTP и SMTP. Имеется возможность получить доступ к сети Internet без установки сетевого адаптера, посредством установки модема и протокола PPP; а также при наличии многопортовых модемов или последовательных плат — Linux обеспечивает эффективный и надежный шлюз PPP для удаленных пользователей, звонящих по коммутируемой линии.

*Небольшие требования к вычислительным ресурсам.* В Linux объединены мощь и гибкость рабочей UNIX-станции, возможность использования полного набора приложений Internet и полнофункцио-

нальный графический интерфейс, при незначительных требован... к вычислительным ресурсам. Все это свободно инсталлируется * любом PC, оснащенном процессором *386,* 486 или Pentium. Компьютер с процессором 386, памятью 8 Мбайт и винчестером емкость ю 100 Мбайт вполне может работать в качестве не только клиента, но и сервера сети. Это минимальные требования, но как сервер Web компьютер с Pentium MMX 166МГц с 64 Мбайт памяти и ОС Linux может поспорить с компьютером Pentium II 300МГц с 128Мбайт памяти на базе Windows NT.

Аппаратные требования для Linux: минимальные (рекомендуемые):

*Память:* 4Мб (32 Мб);
*Процессор:* 80386 (IP-166 МГц) или совместимый;
*Винчестер:* 100 Мб (600 Мб).

***Таблица 3.3.*** **Сравнительные характеристики в потребностях различных ОС**

| Потребность в оперативной памяти Требования к системе | Linux | 05/2 | Wiondows NT |
|---|---|---|---|
| Только командная строка, никаких графических сред (05/2 и NT в этом режиме не работают) | 2 МБ | нет | нет |
| Только загрузка системы | 6 МБ | 4МБ | 2 МБ |
| Типовой состав операционной системы | 8 МБ | 8 МБ | 6 МБ |
| Квалифицированный пользователь, работающий с большим количеством приложений одновременно | 12 МБ | 16МБ | 24 МБ |
| Потребность в дисковой памяти | 15МБ | 20 МБ | 50 МБ |

*Дистрибутив Red Hat.* Сейчас наиболее популярным вариантом Linux является дистрибутив Red Hat Ыпих. Самыми яркими особенностями Red Hat, резко выделяющими ее на фоне других бесплатных систем, являются наличие средств управления пакетами (Red Hat Package Manager, KPM) и графической панели управления (Control panel). KPM служит для установки программ, проверки их целостности, обновления и удаления ПО. С помощью KPM также можно найти немало полезной информации по установленному программному обеспечению, в частности описание назначения программы или перечень файлов. Панель управления приближает Red Hat к лучшим коммерческим системам UNIX. С помощью панели управления в среде X Window System можно осуществлять контроль практически за всеми ресурсами компьютера, а именно: за пользователями и группами, локальными и сетевыми файловыми системами, принтерами, сетевыми настройками и другими параметрами.

*Оконная система X как базовое средство графических интерфейсов в среде ОС LINUX/UNIX.* Для нормальной организации работы пользовательских программ с графическими терминалами (если учитывать отмеченные выше стандартные требования к графическому интерфейсу) требуется наличие некоторого базового слоя программного обеспечения, скрывающего аппаратные особенности терминала; обеспечивающего создание окон на экране терминала, управление этими окнами и работу с ними со стороны пользовательской программы; дающего возможность пользовательской программе реагировать на события, происходящие в соответствующем окне (ввод с клавиатуры, движение курсора, нажатие клавиш мыши и т. д.). Такой базовый слой графического программного обеспечения принято называть оконной системой.

Среди пользователей UNIX предпринималось несколько попыток создания оконных систем, и большинство из них успешно использовалось практически (например, оконная система NeWS компании 5ип Microsystems, интерфейс которой основывался на использовании языка Postscript). Однако ни одна из этих систем не выходила за пределы ведомственного использования, что, естественно, резко ограничивало мобильность программ, обладающих графическим интерфейсом. Успеха удалось добиться группе молодых исследователей и программистов из Массачусетского технологического института, которые создали оконную систему под кратким и предельно скромным названием X (кстати, именно так правильно называть систему X Window System, т. е. «оконная система X»). В настоящее время оконная система X является фактическим стандартом опорных средств графического интерфейса. Система X, Дополнительные библиотеки, а также ряд готовых интерфейсных средств распространяются бесплатно. В то же время сегодня именно оконная система X является базовым механизмом организации графических интерфейсов пользователя в большинстве UNIX-систем.

*Общая организация X Window.* По-видимому, оконная система X победила потому, что организация системы очень точно соответствует общей идеологии ОС UNIX. UNIX — это традиционно сетевая операционная система. Девиз Билла Джоя и всей компании Sun Microsystems «The Network is the Computer» — «Сеть — это компьютер» в полной мере относится к направлению ОС UNIX в целом. Популярная ныне архитектура организации программно-аппаратных средств «клиент-сервер» всегда была совершенно естественной в мире UNIX. Специализация и разделение функций в сети — это и значит, что для пользователя компьютер и сеть неразличимы.

На этих идеях построена и оконная система X. Поскольку Ос UNIXявляется интерактивной операционной системой, то каждый работающий в системе пользователь взаимодействует с системой через предоставленный ему терминал (рабочую станцию) и, вообще говоря, вызывает программы, которые будут выполняться на других компьютерах локальной или территориально распределенной сети. Именно эти программы обеспечивают интерфейс с данным пользователем, т. е. они должны иметь возможность работать с терминалом пользователя вне зависимости от того, где они выполняются. Более того, чтобы возможности графического интерфейса этих программ удовлетворяли общим требованиям, нужно, чтобы программы могли не заботиться о таких деталях, как многооконная организация экрана, текущее расположение окон, управление мышью и т. д.

Оконная система X предоставляет в точности требуемые возможности. На стороне пользовательского терминала находится сервер системы X, обеспечивающий единообразное управление графическим терминалом вне зависимости от его специфических аппаратных характеристик. В других компьютерах сети (которые, фактически, являются серверами с точки зрения организации вычислительного процесса) установлены клиентские части системы X, создающие впечатление у выполняемой программы, что она взаимодействует с локальным терминалом, а на самом деле поддерживающие точно специфицированный протокол взаимодействий с сервером системы X.

*Клиентская и серверная части оконной системы X,* хотя в целом соответствуют идеологии архитектуры «клиент-сервер», обладают тем своеобразием, что серверная часть системы находится вблизи пользователя (т. е. основного клиента вычислительной сети), а клиентская часть системы базируется на мощных серверах сети. Конечно, система X обладает достаточной гибкостью, чтобы допустить расположение серверной и клиентской частей системы в одном компьютере, в разных компьютерах одной локальной сети и удаленных компьютерах, входящих в состав территориально распределенной сети. В зависимости от конфигурации системы X-сервер может обслуживать один или несколько графических экранов, клавиатуру и мышь, реально представляя собой процесс, группу процессов или выделенное компьютерное устройство (X-терминал).

Для обеспечения требуемой гибкости, взаимодействия клиентской и пользовательской частей системы X по мере возможностей не должны были зависеть от используемых сетевой среды передачи данных и сетевых протоколов. Конечно, полная независимость — это теоретически недостижимая цель (всегда и везде), но что касает-

ся системы X, то она действительно умеет работать в большинстве распространенных сетевых сред (в том числе, естественно, в стандартных для ОС UNIX сетях, основанных на семействе протоколов TCP/IP).

Основой взаимодействия между клиентом и сервером оконной системы X является так называемый X-протокол, представляющий собой точную спецификацию допустимых запросов от клиента к серверу и допустимых ответов сервера к клиенту. Как указывается в документации оконной системы X, X-протокол обладает следующими особо привлекательными качествами:

- при использовании этого протокола обработка взаимодействий клиента и сервера ведется единообразно, независимо от того, основана она на внутренних механизмах IPC или на реальных сетевых обменах; это позволяет добиться прозрачности сетевой среды как с точки зрения конечного пользователя, так и с точки зрения разработчика прикладных программ;
- за счет наличия строгой и не зависящей от окружения спецификации X-протокол может быть реализован на различных языках в различных операционных средах;
- X-протокол может быть реализован на основе любого надежно поддерживаемого потока байтов (обеспечиваемого внутренними механизмами IPC или внешними сетевыми механизмами); многие из пригодных механизмов являются стандартными и реализованы в большинстве архитектур.

Для большинства (хотя и не для всех) приложений X-протокол не порождает существенных задержек при работе с графическими терминалами. Обычно задержки вызываются скорее временными потребностями самих терминалов, а не расходами на протокольные взаимодействия клиента и сервера.

Одним из клиентов оконной системы обычно является так называемый «оконный менеджер» (window manager). Это специально выделенный клиент оконной системы, обладающий полномочиями на управление расположением окон на экране терминала. Некоторые из возможностей X-протокола (связанные, например, с перемещением окон) доступны только клиентам с полномочиями оконного менеджера. Во всем остальном оконный менеджер является обычным клиентом.

*Базовые библиотеки.* Понятно, что теоретически любая прикладная программа, которой требуется взаимодействовать с X-сервером, могла бы работать с ним, обмениваясь сообщениями в соответствии с X-протоколом. Однако, конечно же, это неудобно. Для выполнения любого, самого простого действия с терминалом клиенту требу-

ется обменяться несколькими сообщениями с X-сервером, причем для *наиболее* распространенных действий последовательность таких сообщений предопределена. Библиотека Си-функций, которая поставляется вместе с оконной системой X и облегчает взаимодействие Си-программы с X-сервером в соответствии с X-протоколом, называется XLib. Сам X-протокол достаточно компактен, поскольку в нем специфицированы мелкие сообщения, которые, как правило, можно реально использовать только в некоторых комбинациях. XLib — это уже довольно большая библиотека. Это потому, что каждая функция библиотеки XLib основана на использовании нескольких протокольных сообщений (а после этого общее количество функций XLib определяется законами комбинаторики). Вместе с тем XLib — это всего-навсего интерфейсная библиотека над X-протоколом. Если не подниматься над уровнем XLib, то для создания любого графического объекта или сценария графического протокола в каждой прикладной программе придется повторять примерно одни и те же последовательности вызовов функций XLib.

Уровнем, который позволяет использовать ранее созданные графические образы и/или заготовки интерфейсов, является библиотека XI (X Toolkit) Intrinsics. Эта библиотека служит для создания и использования уже существующих элементов пользовательского интерфейса, называемых виджетами (widgets). Библиотека XI Intrinsics выполнена в объектно-ориентированном стиле, так что каждый виджет представляет собой класс, который может использоваться для порождения новых классов, представляющих собой комбинированные виджеты, и т. д.

По своим идеям XI Intrinsics *является* мощным средством разработки пользовательских графических интерфейсов. Однако эта библиотека разрабатывалась в МТИ и в основном являлась исследовательским прототипом. Хотя несколько компаний представили в свободное использование собственные наборы виджетов, их общее количество оказывается недостаточным для быстрого и качественного производства графических пользовательских интерфейсов. Это позволило занять лидирующее положение на коммерческом рынке инструментальному пакету консорциума Open Software Foundation (OSP) Motif.

***Ключи при запуске программ и их интерпретация.*** *В* отличие от программ командной строки, у которых бывают однобуквенные ключи, которые можно группировать вместе (ls -laR), и длинные ключи, начинающиеся с двойного минуса (ls -help). У программ под X бывают *только* длинные ключи, причем *начинаются* они всегда с одиночного символа «-».

Поскольку разбор командной строки выполняется не самой программой, а специальной библиотечной функцией, то есть несколько ключей, поддерживаемых всеми программами.

Стандартные ключи X-программ:

| Ключ | Назначение |
|---|---|
| -help | Выдать краткую справку по ключам программы |
| -display дисплей | Запустить программу на указанный дисплей |
| -fg цвет<br>-foreground цвет | Цвет букв («цвет переднего плана») |
| -bg цвет<br>-background цвет | Цвет фона |
| -fn шрифт<br>-font шрифт | Основной шрифт |
| -geometry геометрия | Сделать окно указанного размера и/или поместить его в указанном месте на экране |
| -iconic | Запустить программу сразу «свернутой» в пиктограмму |

*Управление цветами.* Поскольку X-Window работает с дисплеями, у которых цвет кодируется триплетом RGB (Red, Green, Blue), то использует именно такое представление. Цвета можно указывать одним из двух способов: либо триплетом RGB в шестнадцатеричном виде, либо по имени.

При первом способе цвет указывается как «#RRGGBB» — символ «решетка» и затем шесть шестнадцатеричных цифр — по две цифры на красный, зеленый и синий.

Например, «#FFFFFF» — белый, «#000000» — черный, «#CC00CC» — темно-фиолетовый, а «#92FF41» — желто-зеленый.

Во втором случае указывается английское название — «white», «black», «red», «darkblue», «тагоон» И Т. Д.

Те, кто знаком с HTML, сразу заметят, что в X цвета указываются точно так же. Дело в том, что первоначально World Wide Web создавалась именно под платформу X-Window, и способ указания цветов перешел из X в HTML практически без изменений.

X «знает» по именам несколько сотен цветов. В их число входят «стандартные» цвета типа white, blue, green, их подвиды типа lightblue, darkblue и даже градации яркости — SteelBlue1... SteelBlue4, а также «экзотика» типа moccasin и PowderBlue. Многие из этих цветов на глаз практически не различимы (а некоторые и просто фигурируют под разными названиями — например, cyan и cyan1). Конкретный набор цветов определяется настройкой конкретного X-сер-

вера, но основные цвета (такие, как red, green, black) есть всегда. Заглавные и маленькие буквы в именах цветов не различаются.

Посмотреть список цветов (и их определений) можно командой `showrgb`. Она печатает R-, O- и B-компоненты цвета (в десятичной системе) и его имя. Поскольку `showrgb` выводит на экран огромное количество информации, лучше всего перенаправить вывод команде `less`.

Ключи «`-fg`» и «`-bg`» позволяют указывать программам, какие цвета использовать для букв и для фона.

Единственная проблема — в каждой программе используется много разных элементов, у которых по умолчанию могут быть разные цвета. А эти два ключа могут указывать только *один* цвет сразу для *всех* элементов. Но для несложных программ (особенно на основе Athena Widgets, например xterm) это вполне приемлемо.

> ***Замечание:*** *аналогичная проблема существует и со шрифтами. В X-Window есть средство для более тонкой настройки (причем не обязательно из командной строки), именуемое ресурсами.*

Кроме того, многие программы позволяют менять цвета прямо изнутри — в разнообразных меню настройки.

*Управление шрифтами в X-Window.* Для сравнения рассмотрим, как именуются шрифты в Windows. Там для однозначного указания шрифта служат три параметра: гарнитура, начертание и размер. Например, Arial Bold 12pt (Гарнитура Arial, жирный, размер 12 пунктов).

В X-Window имя шрифта состоит из 14 компонентов, разделенных дефисами, например:

```
-b&h-lucida-bold-r-normal-sans-12-120-75-75-p-79-iso8859-1
```

Общий формат такой:

```
-fndry-fmly-wght-slant-sWdth-adstyl-pxlsz-ptSz-resx
-resx-resy-spc-avgWdth-rgstry-encdng
```

Поля fndry и fmly — это фирма-создатель и гарнитура шрифта — аналог *гарнитуры* в Windows (но в X может одновременно использоваться, например, два разных шрифта Times от разных фирм).

Поля wght, slant и sWdth — это соответственно «жирность» (вес) шрифта, наклонность и ширина — аналог «начертания» в Windows.

Поле pxlsz указывает размер шрифта в пунктах.

Поле spc определяет вид шрифта — постоянной ширины (m monospace) или пропорциональный (p — proportional).

Пара полей rgstry и encdng задает кодировку шрифта — например «iso8859-1» или «koi8-r».

Обычно не требуется указывать шрифт с полной конкретностью — 7. е. перечислять все его 14 параметров, а нужен, к примеру, «постоянной ширины, прямой, 12 пунктов». В этом случае достаточно воспользоваться шаблоном — указать лишь требуемые параметры, а в остальных позициях — «*»:

```
-*-*-*-r-*-*-12-*-*-*-m-*-*-*
```

X-Window автоматически подберет первый подходящий шрифт. Это дает определенную гибкость. Во-первых, программа «просит» не слишком конкретно, и ей подберут какой-нибудь из имеющихся (зачастую это совершенно разные шрифты, но с одинаковыми параметрами, а потому равно подходящие). Во-вторых, это позволяет, например, заставить программу использовать вместо европейских шрифтов русские — достаточно сделать русские шрифты более приоритетными.

Хотя такой «детальный» способ очень удобен для программ, человеку запоминать и набирать такие длинные имена, наоборот, чрезвычайно неудобно. Поэтому для многих шрифтов есть «псевдонимы» (aliases) — короткие имена, аналогичные используемым в Windows. К примеру, за названием «lucidasans-italic-12» скрывается шрифт «-b&h-lucida-medium-i-normal-sans-12-120-75-75-p-71-iso8859-1».

*Программы для выбора и просмотра шрифтов: xfontsel и xfd.* Для подбора шрифта служит программа xfontsel (X Font 5elec1or) — рис. 3.9.

Она отображает меню в виде формата шрифта, в котором можно выбирать нужные параметры, текущий выбранный шаблон, и как выглядит первый подходящий к нему шрифт.

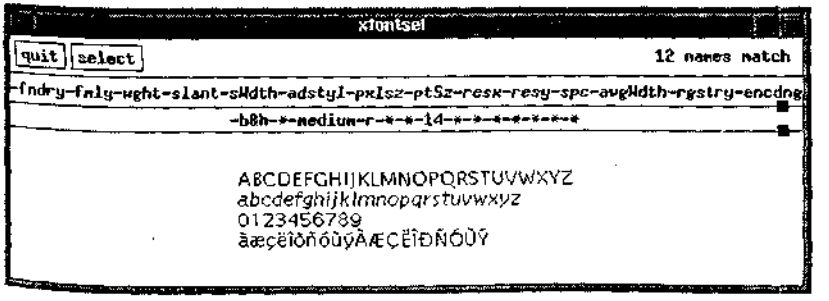



Рис. 3.9. Окно программы xfontsel

При стандартной инсталляции Linux xfontsel сразу при запуске (шаблон из одних «*») покажет корейский шрифт — просто он оказался самым первым в списке. Вообще, в дистрибутиве X-Window сразу имеются японские и корейские шрифты, что позволяет, к примеру, свободно просматривать в Netscape сайты *.jp* и *.kr* (естественно, при условии владения соответствующим языком).

Поскольку шрифты есть не для всех комбинаций параметров, то по мере выбора параметров из меню некоторые пункты в других меню становятся «недоступными» — например, шрифты гарнитуры Times есть только в кодировках iso8859 и koi8, а в jis — нет.

Подобрав шрифт, можно нажать кнопку [Select] — при этом название будет скопировано в «карман» (буфер обмена, clipboard), и его затем можно вставить, например, в xterm средней кнопкой мыши.

Для просмотра всех символов шрифта, а не только тех, что отображает в своем примере xfontsel, служит программа xfd (X Font Displayer).

Ей надо указывать требуемый шрифт (или шаблон) ключом «`-fn`». При этом имя шрифта следует заключать в одинарные апострофы, поскольку оно обычно содержит символы, имеющие специальное значение для интерпретатора командной строки (shell), такие, как «*» и «&».

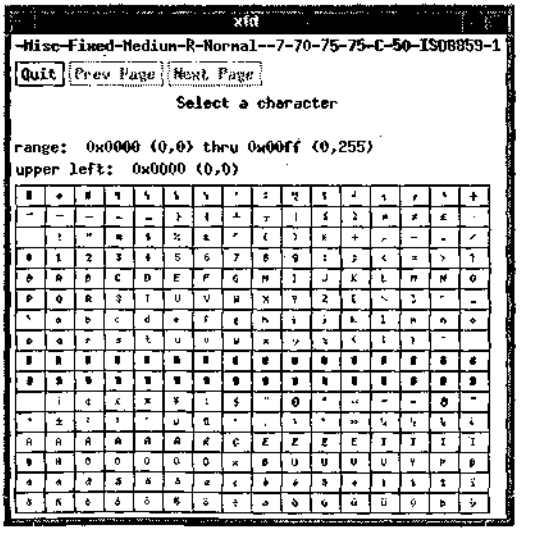



Рис. 3.10. Окно программы xfd, запущенной командой «`xfd -fn 5x7`»

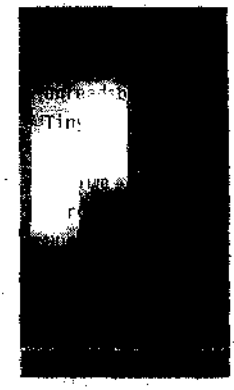


Рис. 3.11. Меню выбора шрифта в Xterm

*Как указывать шрифты программам.* Указать программе, какой шрифт использовать, можно при помоши ключа `-fn`. Ситуация здесь аналогична таковой с цветами — указывается *один* шрифт для *всех* элементов. Но практически любой хороший текстовый редактор позволяет выбрать используемый шрифт где-нибудь в меню.

Довольно интересен способ выбора шрифта в программах Xterm и NXterm. При нажатии <Ctrl+ПраваяКнопкаМыши> появляется меню шрифтов, в котором самый нижний пункт называется Selection. Если выбрать в xfontsel шрифт и «отметить» его, нажав Select (или просто набрать где-нибудь имя шрифта и выделить его мышью), а потом выбрать в меню Xterm пункт Selection, то Xterm сменит шрифт на указанный.

*Геометрия окон при запуске программ.* Координаты и размеры окон от запуска к запуску программы не запоминаются, но их можно указать при запуске. Для этого служит ключ `-geometry`». Под геометрией окна понимается его размер и позиция на экране.

У ключа `-geometry` есть три варианта:

`geometry` *Размер* •

`geometry` *Позиция*

`geometry` *РазмерПозиция*

Размер указывается как «ШИРИНАxВЫСОТА» (между размерами — латинская буква «*x*»). В зависимости от программы ширина и высота указываются или в пикселах (в большинстве программ) или в символах (например, в Xterm). Причем указывается размер внутренней части окна, без учета обрамления (которое зависит от менеджера окон и может быть каким угодно).

Позиция указывается в пикселах от края экрана. Обычно она выглядит как «+X+Y». В этом случае левый край окна будет в X пикселах от левого края экрана, а верхний край — в V пикселах от верхнего края экрана.

Но иногда нужно поместить окно, например, вплотную к правому краю. В этом случае надо в горизонтальной координате вместо «+» указать «-». При этом окно будет расположено так, что его правый край окажется в X пикселах от правого края экрана.

Аналогично вертикальную позицию можно так же указывать от нижнего края экрана. Несколько примеров приведены на иллюстрации ниже.

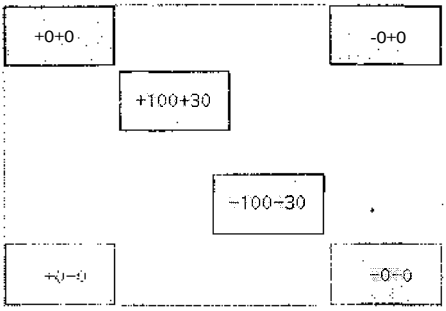



Рис. 3.12. Координаты окон на экране

Несколько примеров указания геометрии:

```
xterm -fn 6x10 -geometry 80x24+30+200 &
xclock -geometry 48x48-0+0 &
xload -geometry 48x48-96+0 &
xbiff -geometry 48x48-48+0 &
```

Здесь Xterm размещается приблизительно в левой средней части экрана, а часы, индикаторы загрузки и писем — в правом верхнем углу.

Обычно ключ `-geometry` используется не при запуске программ «вручную», а для указания позиции программам, запускаемым автоматически при старте X Window (из инициализационных файлов X и Window Manager'a).

### *Текстовый редактор NEdit*

*Запуск и основные возможности* NEdit

NEdit был создан в Национальной Ускорительной Лаборатории Ферми (Фермилаб). Название расшифровывается как «Nirvana Editor» (вероятно, это обещание достижения указанного состояния при использовании NEdit:).

NEdit — редактор с графическим интерфейсом для обычных текстовых файлов. Он очень похож в обращении на редакторы под Windows и Macintosh.

Основные достоинства NEdit:

- простота освоения и использования;
- практически все действия можно выполнять как при помоши мыши, так и с клавиатуры;
- команда отмены действия (Undo) не имеет лимита на число операций;
- можно отмечать как потоковые, так и прямоугольные блоки текста; копирование и перемещение блоков можно делать простым перетаскиванием мышью;

- многооконность — каждый файл открывается в своем окне, а каждое окно можно разбить на несколько частей для одновременного просмотра и редактирования разных частей файла;
- практически все настройки можно устанавливать непосредственно в меню программы — никакого специального знания X-Window и редактирования файлов настроек не требуется;
- программируемость — можно как записывать последовательности нажатий клавиш, так и создавать макросы на встроенном C-подобном языке.

NEdit позволяет выделять цветом конструкции языков программирования (причем эта возможность настраиваемая) и содержит еще много функций, полезных при написании программ — автоматический «умный» отступ, подсветку парных скобок, возможность компиляции прямо из редактора и т. д.

Довольно подробная документация содержится в меню [Help], в частности очень полезен для начинающих пункт «Getting Started».

Для запуска достаточно набрать «`nedit`». Можно сразу указать в командной строке файлы, которые следует загрузить.

*Некоторые «нестандартности» и возможности* NEdit

«Перетаскивание» отмеченных блоков текста делается не левой кнопкой мыши, а средней.

Самый верхний пункт в каждом меню (выглядящий как «——— - -») позволяет «приклеить» меню к экрану, так что оно будет все время доступно.

Переключение между режимами «вставка» и «замена» производится не кнопкой <Ins>, а <Ctrl+B> (в некоторых версиях (напри-

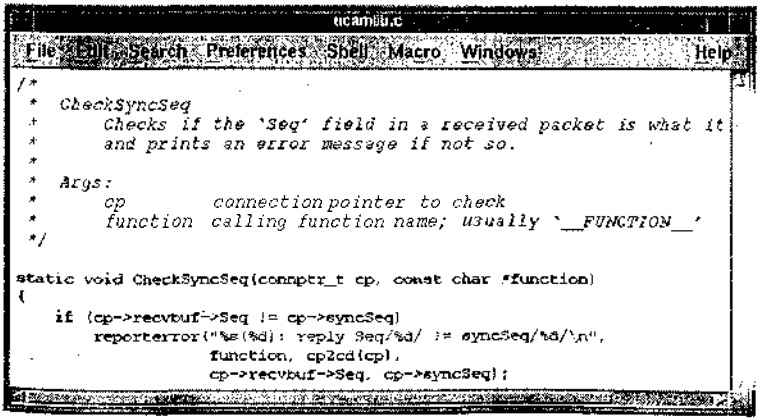



Рис. 3.13. Окно NEdit

мер, в установленной на Sky) надо использовать пункт Overtype в меню Preferences).

Можно включить отображение «строки состояния», в которой показывается номер строки и позиция, — для этого используется пункт Statistics Line в меню Preferences или комбинация <Alt+A>.

*Настройка.* Меню Preferences позволяет менять «поведение» редактора с текущим файлом — автоматический перенос слов, отступ, шрифт И т. д.

В подменю Default Settings можно менять настройки по умолчанию — они будут использоваться для всех следующих файлов. Там же настраивается поддержка языков программирования (пункт Landиаде Modes и меню Зуп1ах Highlihting). Для сохранения настроек служит пункт Save Defaults.

***Манипуляции с изображениями XV.*** Графический редактор XV разработан в Университете Пенсильвании. Эта программа предназначена для просмотра изображений в различных форматах и простейших манипуляций с ними. XV отличается тем, что использует свою собственную библиотеку интерфейсных элементов. И хотя существуют и более мощные программы (например, GIMP), понимающие большее число форматов файлов, вследствие удобства и простоты обращения XV чрезвычайно популярен.

При запуске появляется окно изображения, в котором, если не указан никакой файл, первоначально отображается заставка программы. По нажатию <Правой> кнопки на изображении появляется окно управления.

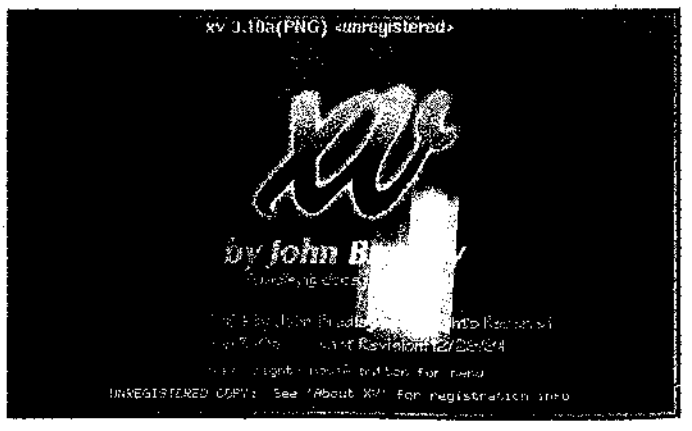



Рис. 3.14. Окно заставки xv

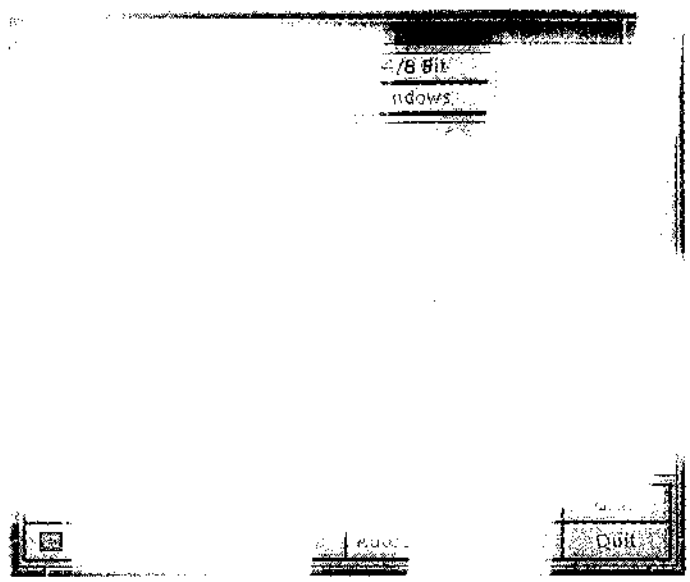



Рис. 3.15. Окно управления XV

В окне управления справа расположены кнопки для работы с файлами, снизу — для простых манипуляций с изображениями (копирование, поворот и т. д.). Центральная часть окна отведена под список файлов, который заполняется по мере просмотра и позволяет быстро вернуться к предыдущим файлам.

Сверху же расположены шесть меню:

Display — для управления отображением цветов;

24/8 Bit — для смены режима работы (например, редактировать цвета можно только в режиме 8 бит, а наилучшее качество достигается при 24);

Algorithms — позволяет применить к изображению больше десятка разных алгоритмов, например для повышения четкости;

Root — позволяет отобразить картинку не в окне, а разнообразными способами на рабочей поверхности экрана в качестве «обоев». Интерактивно практически не используется;

Windows — дает доступ к дополнительным окнам — редактора Цветов, информации об изображении, справки по использованию клавиш и мыши;

ImageSize — манипуляция с размерами изображения.

Хотя XV имеет функции для простейшего рисования, они практически не используются.

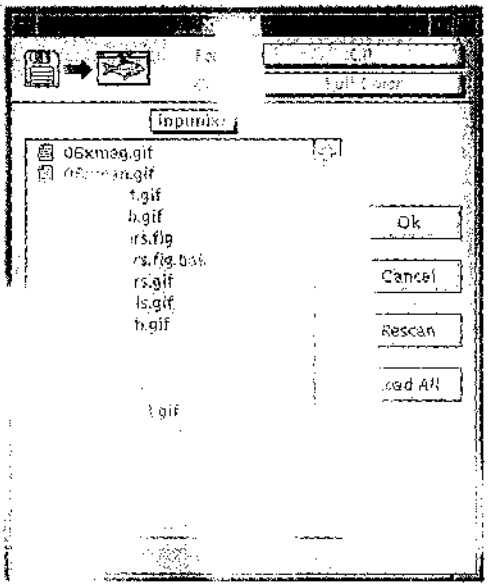



Рис. 3.16. Окно загрузки файла

*Просмотр файлов и простейшие манипуляции с изображениями.* Кнопка Load вызывает окно загрузки файла. Если включить переключатель Browse, *то* при загрузке файлов окно Load будет оставаться на экране, что позволяет быстро просматривать много файлов.

Изображения, превышающие по размерам экран, XV показывает в сжатом виде, чтобы они целиком помещались на экране. Для обхода этого ограничения можно воспользоваться ключом «-nolimits».

Загруженное изображение можно поворачивать на 90°, зеркально отображать и применять к нему преобразования из меню Algorithms.

Левой кнопкой мыши можно отметить прямоугольный фрагмент изображения. Тогда все операции (кроме поворота и изменения карты цветов) будут применяться к отмеченному фрагменту. Для отмены достаточно щелкнуть мышью вне выделенной области. Кнопка Crop (в окне управления) урезает изображение до выделенной области.

Для сохранения изображения служит кнопка Save. Файл можно сохранить в другом формате, выбрав соответствующий вариант из списка вверху. Если включить опцию Selected Area, то будет сохранена не вся картинка, а только выделенная часть.

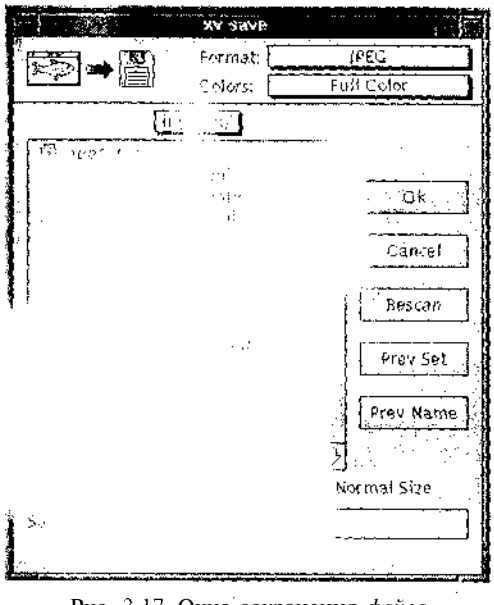



Рис. 3.17. Окно сохранения *файла*

*Копирование окон с экрана.* При помощи XV можно копировать изображения окон с экрана и даже весь экран (именно так были сделаны почти все иллюстрации в данном курсе).

Для этого служит кнопка Grab (дословно «grab» означает «хватать»). Она вызывает окно, в котором можно установить параметры захвата: убирать ли на это время окна самого XV; как копировать: по нажатию кнопки мыши над окном (Grab) или автоматически (AutoGrab) после заданной паузы скопировать окно под мышью; интервал ожидания перед копированием.

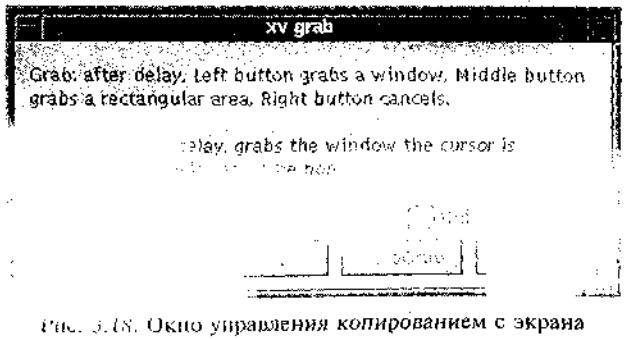



Рис. 3.18. Окно управления *копированием* с экрана

Чтобы скопировать весь экран, надо указать на место на экране, не занятое никакими окнами, — это будет «самое нижнее окно» (root window), которое содержит в себе все остальные окна.

***Графический редактор «GIMP».*** Расшифровывается как Gnu Image Manipulation Program. Как следует из названия, это программа для манипуляций с изображениями. По возможностям GIMP похож на редакторы PaintShop Pro и Adobe PhotoShop.

При первом запуске производится начальная настройка программы. Далее сразу же появляется основная панель инструментов, в верхней части которой расположено меню.

Выбрав в меню File команды New или Open, можно создать новое изображение или загрузить его с диска.

В окне редактирования можно вызвать меню при помощи правой кнопки мыши.

GIMP «знает» про такое свойство изображений, как прозрачность (т. н. «альфа-канал»), и отображает прозрачные области шахматной текстурой.

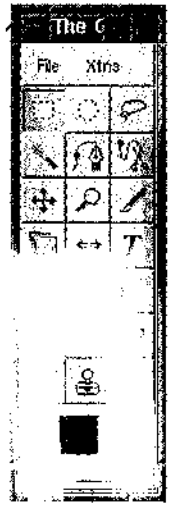



Рис. 3.19. Панель инструментов GIMP

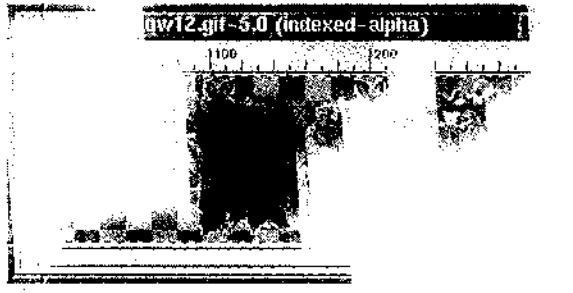



Рис. 3.20. Экран редактора GIMP

***Просмотр файлов в форматах PostScript и PDF: ghostview, gv, AcroRead.*** Формат PostScript был разработан в середине 80-х годов фирмой Adobe как язык для принтеров. Поскольку в то время Unix использовался исключительно на мощных и дорогих компьютерах, то к ним обычно были подключены дорогие PostScript-принтеры. Поэтому термин «печать» в графических программах под Unix обычно означает выдачу данных в формате PostScript, которые потом либо отправляются на принтер (если он не

PostScript, то фильтр принтера автоматически преобразует их в нужный формат), либо сохраняются в файл с расширением .ps.

Файлы PostScript бывают двух видов: собственно одно- или многостраничные файлы, которые можно сразу печатать на PostScript-принтере (файлы .ps), и Encapsulated PostScript — формат для рисунков, которые затем вставляются в другие документы (например, в TeX), такие файлы обычно имеют расширение .eps.

Достоинством PostScript является его универсальность — файл, созданный в одной системе, можно использовать и в другой. К недостаткам относится, во-первых, большой объем (особенно если в файлах есть растровые изображения — они записываются неком-

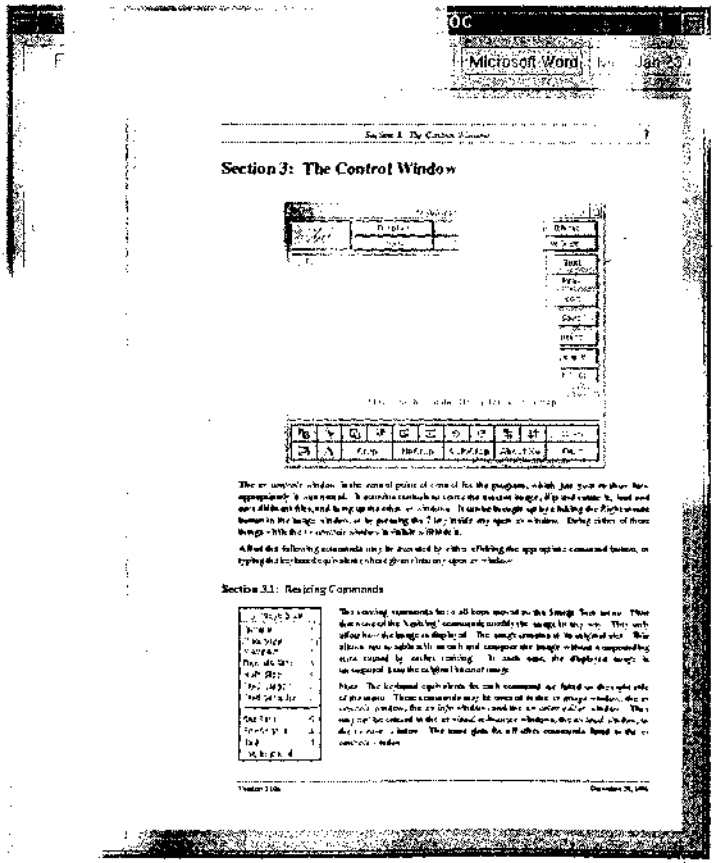



Рис. 3.21. Окно gv с документом

пактно), а, во-вторых, то, что программы просмотра показывают файл «как картинку» — смотреть можно, а, например, найти определенное слово в тексте — нельзя.

Формат PDF (Portable Document Format) был разработан в начале 90-х годов той же фирмой Adobe, и, в частности, в нем исправлены вышеозначенные недостатки PostScript.

И PostScript и PDF — форматы для оконечных устройств вывода (принтеры и дисплеи), файлы в них никогда не редактируются, а только создаются из файлов других форматов (`.dvi`, `.html`, `.doc` и т. д.).

*Стандартный просмотрщик PostScript: ghostview.* GhostView — самая старая в Unix программа для просмотра PostScript.

Клавишами она практически не управляется (из-за плохих настроек по умолчанию) — попытки нажимать стрелки или <PgUp/PgDn> приводят к любопытным эффектам (например, поворот изображения).

Щелчок левой кнопкой мыши на списке страниц не показывает указанную страницу, а отмечает ее. Для перехода же к другой странице используется средняя кнопка мыши.

Меню Magstep позволяет менять масштаб изображения.

Более новый просмотрщик PostScript: gv. Gv — это тот же GhostView, переработанный в сторону улучшения пользовательского интерфейса. Файл для просмотра можно или указать в командной строке, или открыть командой Open из меню File.

В левой части окна расположен список страниц. Щелчок левой кнопкой показывает нужную страницу, правой или средней кнопкой можно отметить несколько страниц, чтобы потом распечатать их или сохранить в другой файл. Кнопки над списком позволяют быстро отмечать сразу все нечетные/четные страницы.

Вместо полосок прокрутки изображения используется «карта страницы», по которой можно передвигать «изображение окна просмотра». Передвигать страницу можно также клавишами со стрелками или просто перемещая изображение левой кнопкой мыши.

В верхней части окна расположены меню. File — для работы с файлами. 3la(e — для настроек (в нем очень полезно включать опцию Antialias для лучшего качества изображения). Раде — для перемещения по страницам и отметки. Далее расположены меню ориентации изображения (Portrait, Landscape и т. д.), масштаба изображения и формата бумаги — *в* заголовках этих меню отображаются текущие параметры.

Кроме PostScript-файлов, gv «по совместительству» умеет показывать и формат PDF.

## Задания к главе 3

1. ***Использование команд операционной системы UNIX.***

   Перейдите в каталог /etc.

   Укажите шаблоны, которые подходят для следующих имен файлов:

   — всех имен;

   — всех имен, которые начинаются с «.»;

   — всех имен, которые начинаются с «.c»;

   — всех имен, которые начинаются с «a» и оканчиваются на «.f»;

   — всех имен, которые оканчиваются не на «V», «w», «x», «y» или «r»;

   — всех имен, которые состоят из трех строчных букв;

   — всех имен, которые содержат, по крайней мере, одну гласную букву;

   — все элементы, которые начинаются с буквы и имеют длину в два символа.

   Что происходит, если shell не находит подходящего элемента?

2. ***Процессы. Порождение процессов. Процессы-родители и процессы-потомки. Взаимодействие процессов.***

   Определите номер процесса вашей стартовой программы shell.

   Завершите свой стартовый процесс shell командой kill.

   Какие процессы выполняются под управлением пользователя (любого из зарегистрировавшихся в системе)?

   Выполните следующую команду:

   **sleep 3600 &**

   Запомните выданный номер процесса PID.

   Завершите фоновый процесс (sleep) командой **kill.** Объясните результат (номер процесса (PIO) берется из задания 4).

   Создайте конвейер из команд **cat /etc/passwd** и nl. Объясните принцип взаимодействия процессов и причину вывода результата выполнения команды nl на терминал пользователя.

   Создайте конвейер команд, в результате выполнения которых можно получить число пользователей, работающих в текущий момент.

3. ***Текстовый редактор VI. Создание и сохранение файлов. Основные режимы работы. Назначение и функции командного режима. Назначение и функции режима редактирования.***

   Скопируйте файл /etc/passwd в ваш входной каталог.

   Вставьте перед первой строкой следующую строку:

   # file /etc/passwd

   Вставьте после строки с вашим регистрационным именем следующую строку:

   user00:x:0:1:superuser:/home/user00:/sbin/ksh

   Измените СЮ в вашей строке и в следующих 3 строках на 100.

# Глава 4. Среды и оболочки операционных систем

Понятие оболочки операционной системы возникло в связи с расширением функций первой из массовых ОС — OS/360 (1964 год). Эта классическая операционная система первоначально была рассчитана на пакетную обработку заданий, поскольку технологии тех времен не могли предложить в качестве терминала ничего лучше электрической пишущей машинки. Появление первых алфавитно-цифровых терминалов (1971 год) вызвало к жизни широкий перечень диалоговых систем обрамления OS/360/370/375 — CRJE (Control Remote Job Entry — Диалоговый удаленный ввод заданий), CICS (Customer Information Control System — система управления использованием информации) и другие средства. Кроме того, наиболее популярные средства наблюдения и контроля за вычислительным процессом включали в свой состав текстовые редакторы и частично перекрывали функции утилит операционной системы.

## 4.1. Диалоговые мониторы ЕС ЭВМ

В свое время было известно большое число разновидностей диалоговых мониторов (JEC, RIM, PRIMUS, FOCUS и др.) отечественной разработки, поскольку существовали различные подходы к компенсации недостатков средств ОС ЕС. Наиболее существенным из этих недостатков является отчуждение пользователя от вычислительного процесса, заложенное в разделении труда оператора ЭВМ и программиста, являющееся основном принципом ОС ЕС ЭВМ.

Рассмотрим основные принципы организации и функционирования диалоговых мониторов на примере системы PRIMUS.

Система PRIMUS выделяет каждому пользователю терминал и *закрепляет* за ним на диске рабочий набор данных (РНД), *выполняющий* функции буфера для формирования и редактирования текстов (исходных данных, программ и *заданий).*

Пользователь общается с системой на 2 уровнях: командном и подкомандном. Команды (или функции) вызываются вводом идентификатора команды (обычно 4-символьного) в ответ на подсказку ПОСЫЛАЙТЕ КОМАНДУ. Идентификатор команды обычно сопровождается аргументами, наиболее употребительными из которых являются имя набора данных и название тома. Большинство команд предполагает использование подкоманд, как правило, односимвольных; в этом случае экран терминала обычно может быть разбит на *3* зоны: подкомандная строка, информационная зона, строка меню. На рис 4.7 приведены основные функции PRIMUS, соответствующие процессу подготовки и отладки программ.

***Основные группы функций***

*Работа с наборами данных на МД*

DSET — создание НД, с указанием по запросу системы параметров DSN, VOL,DCB, SPACE.

INPT — заполнение буфера (РНД) вводимыми текстовыми данными (основной режим формирования файлов программ, заданий, данных).

SAVE [имя_файла[,V=имя_тома]] — запись содержимого РНД в личный файл пользователя на МД.

COPY [имя_файла[,V=имя_тома]] имя_файла — копирование личного НД в РНД; если опущен параметр V, подразумевается имя тома, заданное командой DVOL(см. ниже).

CORR — переход в режим редактирования содержимого РНД.

Основные подкоманды COKK:

— D — удаление строки или группы строк;

— I — вставка строки или группы строк;

— K — замена символов.

В данном режиме используется функциональная часть клавиатуры для вставки/удаления символов и других операций: <P12> — вставка символа; <P10> — удаление символа и т. д.

*P* 'текст' — контекстный поиск, позволяющий локализовать строку НД, содержащую «текст»;

C 'текст1', 'текст2' — контекстная замена, локализующая «текст1» и заменяющая его в РНД на «текст2»;

O X — переход к строке РНД номер X;

H — переход в режим просмотра и редактирования в шестнадцатеричном формате представления данных;

E — выход из редактора;

FROM — команда копирования в РНД фрагментов из личных НД;

PRINT [имя_файла[,V=имя_тома]] — распечатка РНД или личного НД;

LOOK [файл[,V=том]] — просмотр набора данных «на месте», без копирования в РНД. Допускается редактирование в режиме подкоманды K (вставки и удаления строк не разрешены).

*Работа с томами НД и библиотеками*

CONT [файл[,V=том]] — просмотр содержимого библиотечного НД, состоящего из именованных разделов. Разделы библиотеки, имена которых выводятся на экран, могут быть подвергнуты обработке в соответствии со следующими подкомандами:

«*» — просмотр раздела (см. LOOK);

«#» — печать раздела (см. PRNT);

« — » — удаление раздела;

«+» — переименование раздела;

«<» — копирование раздела в РНД (см. COPY);

«E» — окончание работы.

VTOC [метка_тома] — просмотр оглавления тома МД.

Подкоманды:

«*» — просмотр НД (или оглавления библиотечного НД);

«-» — удаление НД;

«+» — переименование НД;

«E» — окончание работы с командой.

Экран, выводимый командой VTOC, представляет собой таблицу, содержащую следующие сведения о НД: DSNAME, DCB, SPACE, фактически занятое пространство.

*Работа с заданиями*

SUBMIT, EXEC — запуск задания, JCL — операторы которого находятся в РНД или личном НД на МД;

D A — просмотр списка активных заданий;

D N — просмотр входных/выходных очередей;

CONS — просмотр копии экрана главной дисплей-консоли оператора ЭВМ;

OPER — переход в режим консоли оператора ЭВМ, позволяющий пользователю не только наблюдать за вычислительным процессом, но и активно вмешиваться в его ход;

SOUT *имя_задания* — просмотр результатов работы заданий.

Подкоманды:

— L X — выбор для просмотра одного из нескольких выходных файловзадания;

— H [X] — вывод файла (или всех результатов) на печать;

— C — уничтожение результатов в выходной очереди.

***Некоторые специальные команды***

DASD — просмотр имен и адресов накопителей на МД;

DVOL — установка по умолчанию имени текущего НМД;

ACRS — работа с указателем пользователей системы (регистрация), установка/изменение статуса, исключение из пользователей;

CANC — завершение работы системы PRIMUS.

Кроме вышеперечисленных, диалоговая система PRIMUS обеспечивает также функции регистрации пользователей, копирование НД и некоторые другие, частично перекрывая и дополняя возможности утилит ОС ЕС.

## 4.2. Монитор PCTools для ПЭВМ

PCTools являлся первым из известных средств расширения командного языка MS-DOS, предусматривавшим достаточно типовые средства работы с устройствами, файлами, текстами. В дальнейшем был вытеснен программным средством Norton Commander.

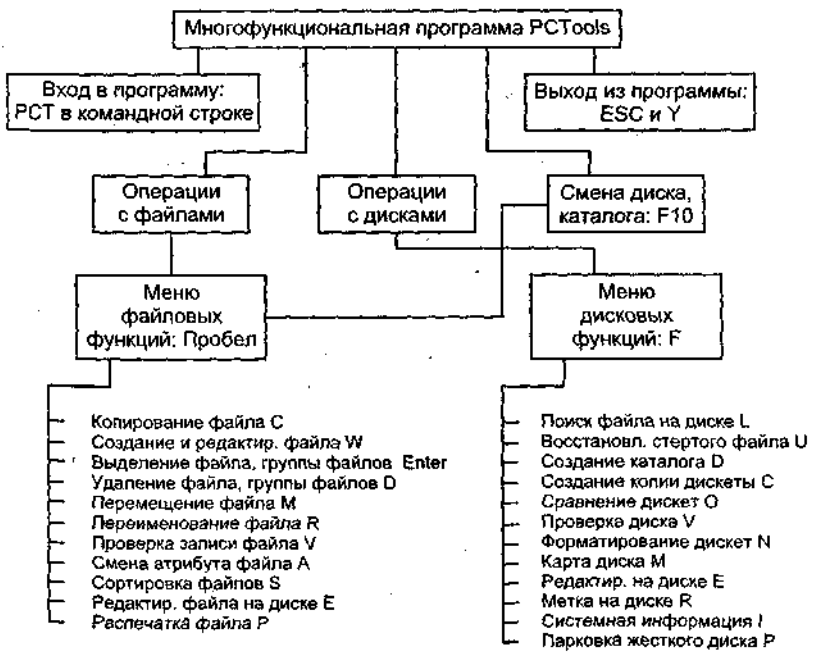



Рис. 4.1. Основные функции PCTools

*Запуск PCTools.* Для запуска pctools надо ввести команду pct с клавиатуры. Если программу запустить с ключом pct /Rnnnk, программа становится резидентной в основной оперативной памяти и может быть вызвана в любой момент нажатием комбинации клавиш <Ctrl+Esc>. Вместо *nnn* следует указать объем памяти, которую будет занимать программа (в килобайтах), который не должен быть меньше 64K. При этом на диске в корневом каталоге будет создан файл pctools.ovl.

Для выхода из PCTools надо нажать <Esc> и подтвердить выход У. После запуска программа предлагает нажать <P3> для переходах меню дисковых функций или любую другую клавишу для перехода к меню файловых функций. Для смены диска или каталога надо нажать <F10>. Соответствующие функции выбираются нажатием выделенной клавиши. После выбора функции обычно появляется уточняющее меню, в котором с помощью стрелок следует выбрать режимы работы. Нажатие клавиши <Esc> в уточняющем меню вызывает аварийное прерывание функции и выход в меню предыдущего уровня. Используя клавиши <Enter> и <P1>, можно выбрать несколько файлов из представленных на экране, и тогда функции будут выполняться со всеми выделенными файлами. Для получения помощи в меню файловых или дисковых функций следует нажать клавишу H. Программа PCTools при относительно небольшой собственной длине выполняет такое же множество функций, как Norton Соттапдег плюс многие функции из Norton Utilities. Эту программу полезно иметь на системных и ремонтных дискетах.

Средство PCTools представляет пользователю 2 основных режима работы.

- файловые функции (работа на логическом уровне) — рис. 4.2;
- специальные функции (работа на физическом уровне) — рис. 4.3.

*Файловые функции.* Программа PCTools выполняет следующие операции с файлами в меню файловых функций (рис. 4.2): создание и редактирование файла W, удаление файла D, перемещение файла M, переименование файла K, копирование файла на диск или на дискету C, проверка записи файла на диске V, смена атрибута файла A, сортировка по имени, расширению, размеру, времени создания файла 5, редактирование файла прямо на диске E, распечатка файла на принтере P и др.

Для выполнения этих функций следует нажать на клавишу с указанной буквой в данном меню.

*Дисковые функции.* Программа PCTools производит следующие операции с файлами, каталогами и дисками в меню дисковых

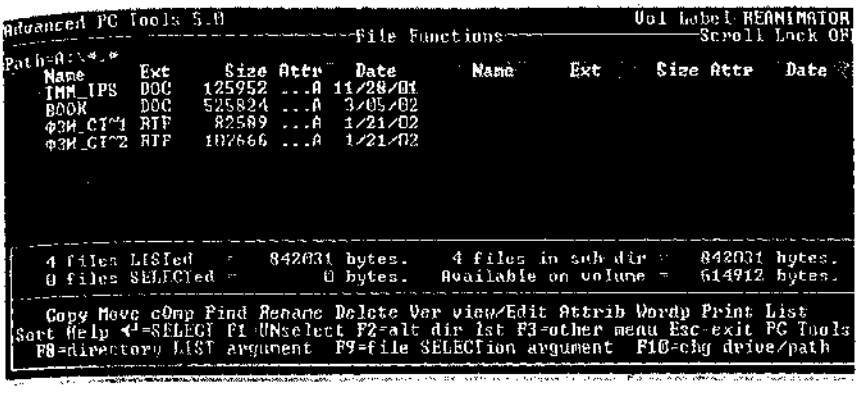



Рис. 4.2. Режим работы с файлами PCTools

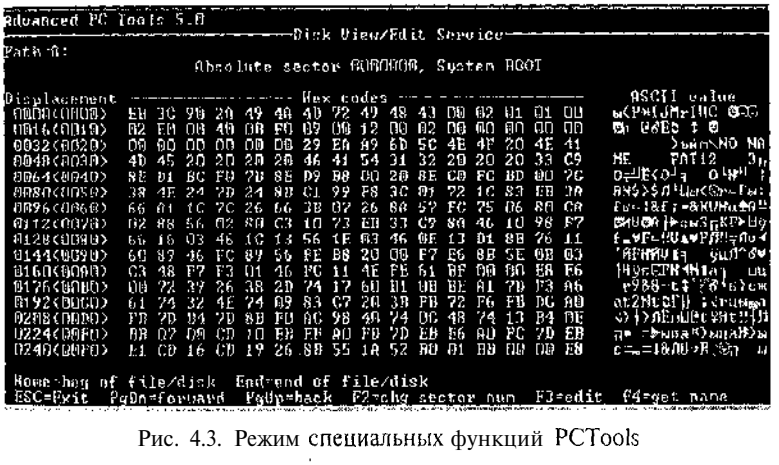



Рис. 4.3. Режим специальных функций PCTools

Функций (рис. 4.3, F3): поиск файлов на диске по имени или расширению L, восстановление удаленных файлов U, создание, переименование, перемещение и смена каталогов D, создание точных копий дискет C, сравнение дискет O, проверка диска на наличие Физических дефектов и отметка поврежденных кластеров V, форматирование дискет N, вывод на экран карты диска или дискеты M, просмотр и редактирование информации прямо на диске E, создание и смена метки диска или дискеты K, выдача системной информации о компьютере I, парковка жесткого диска P и др. Для выполнения этих функций надо нажать на клавишу с указанной буквой в данном меню.

## 4.3. Оболочка NORTON COMMANDER (DOS) и ее графические аналоги для Windows

Оболочка Norton Commander (NC) является самой распространенной из используемых в настоящее время надстроек над DOS, преобразующих ее командный пользовательский интерфейс в интерфейс типа «меню». Она настолько привычна для пользователей ПК, что с ней не хотят расставаться даже те из них, кто уже давно работают в среде Windows, применяя оболочку Norton Commander в качестве «файлового манипулятора». Причины привязанности многих к «классической» оболочке Norton Commander в ее исключительной простоте, привычности работы с ней, в экономном использовании ею ресурсов ПК. Оболочка Norton Commander стала настолько неотъемлемым, естественным атрибутом IBM — совместимого ПК, что всякий, кто по необходимости или любознательности знакомится с новой оболочкой, невольно проецирует ее возможности на возможности оболочки Norton Commander. Продолжением оболочки для сред Windows являются Windows Commander, Norton for Windows, PAK Manager.

Оболочка Norton Commander разработана американской фирмой Peter Norton Computing, которая с 1990 года входит в состав корпорацииSymantec.

Мы не предполагаем здесь подробно останавливаться на возможностях и приемах работы с NC, поскольку ниже будет подробно описан и разобран «нортонообразный» продукт для ОС Windows 95/98/NT/2000 — Par Manager.

***Основные возможности оболочки.*** Оболочка Norton Commander обеспечивает (см. также Приложение 4):

- отображение деревьев каталогов и содержимого каталогов (характеристик входящих в них файлов) в форме, наиболее удобной для восприятия человеком;
- выполнение всевозможных действий с каталогами, файлами и целыми поддеревьями файловых структур, включая их создание, копирование, пересылку, переименование, удаление и поиск, а также смену атрибутов файлов;
- в максимальной степени естественную работу с архивами, включая отображение их содержимого, а также создание, обновление и распаковку архивов (архив представляет собой файл, в котором находится группа сжатых по специальному алгоритму файлов);

- визуализацию файлов, подготовленных популярными текстовыми и графическими редакторами, системами управления базамиданных, электроннымитаблицамиидругимиприкладными программами;
- подготовку текстовых файлов;
- выполнение из ее среды практически всех командDOS;
- запуск программ, для чего используются различные, наиболее удобные для пользователя способы;
- выдачу информации о компьютере в целом, о дисках и об оперативной памяти;
- поддержку межкомпьютерной связи через последовательный или параллельный порт (с использованием модуля Commander Link);
- поддержку электронной почты через модем по телефонным линиям связи (при помощи модуля Terr90, разработанного для корпорации Symantec фирмой BAUSCH datacom GmbH).

Оболочка Norton Commander, как и любая другая оболочка, упрощая взаимодействие пользователя с ПК, полностью все же не освобождает его от необходимости знать пользовательский интерфейс DOS, так как многие функции доступны только на уровне системы или реализуются на этом уровне гораздо эффективнее.

Оболочка Norton Commander столь привлекательна не в последнюю очередь благодаря великолепным высокоскоростным средствам визуализации данных и развитым средствам электронной почты.

*Визуализация файла* состоит в форматировании его содержимого (в подготовке для вывода на экран в форме, пригодной для восприятия) с последующим отображением результата на экране монитора. Формат файла распознается оболочкой автоматически, исходя из расширения его имени и, при необходимости, внутренней структуры.

К достоинствам рассматриваемой оболочки относятся:

- высокая степень интеграции функций;
- удобство выдачи команд DOS — выдавать их из среды оболочки даже удобнее, нежели взаимодействуя с DOS непосредственно;
- поддержка иерархической системы меню (вложенных меню) для запуска программ;
- простота освоения и удобство использования;
- высокая устойчивость в работе и приемлемая защищенность от ошибок пользователя;

- наличие удобного и понятного контекстно-чувствительного интерактивного справочника;
- поддержка манипулятора типа мышь.

Наряду с неоспоримыми достоинствами имеются и некоторые недостатки. Среди них:

- отсутствие средств сортировки каталогов в дереве файловой структуры;
- невозможность выполнения групповых операций над файлами в различных каталогах, не говоря уже о файлах на различных дисках;
- невозможность выполнения каких-либо действий с группами файлов, найденными на диске средствами оболочки (начиная с версии 4.0, выполнение действий с отдельными файлами возможно).

Оболочка Norton Commander не предъявляет к оборудованию ПК никаких особых требований и может работать на всех используемых в настоящее время компьютерах, оборудованных винчестерским накопителем (правда, и без него ядро оболочки использовать можно).

Для размещения всех файлов, образующих оболочку, требуется около 1,8 Мбайт дискового пространства. Основные составляющие файлы пакета: nc.exe — загрузчик, nc.cfg — описание конфигурации оболочки, nc.hlp — справка, ncedit.exe — редактор текстов, pstamn.exe — файловый менеджер, nczip.exe — упаковщик, packer.exe — программа-интерпретатор упаковщика, ncff.exe — поиск файлов, ncclean.exe — чистка дисков, ncdd.exe — копирование дискет, nclabel.exe — изменение метки диска, ncnet.exe — работа в локальной сети, ncsf.exe — форматирование дискет, ncsi.exe — просмотр системной информации, *.msg — сообщения соответствующей утилиты, спец. программы просмотра, гашения экрана, прослушивания wav файлов.

Norton Commander запускается файлом nc.exe (либо ncsmall.exe, ncmain.exe). После запуска программы высвечивает на экране в так называемом «окне» содержимое активного каталога.

Типовая конфигурация экрана NC была приведена выше (рис. 1.28), и здесь мы отметим лишь, что она включает:

- панели (левую и правую) для отображения информации о файлах, дисках и пр., а также для манипулирования файлами и программами;
- ниспадающее (свешивающееся, pull-down) меню, предназначенное для управления видом панелей, конфигурирования системы и др. функций;

. зону командной строки и протокола работы;

. меню использования функциональной клавиатуры.

Для передвижения по «окну» можно использовать стандартные клавиши управления положением курсора (стрелки<←→↑↓>, <Del>, <Ins>, <Home>, <End>, <PgUp>, <PgDn>).

Стрелки служат для перемещения курсора в активном «окне». Клавиши <PgUp> и <PgDn> служат для постраничного перемещения по тексту соответственно вверх и вниз (перелистывание экранных страниц). Клавиши <Home> и <End> предназначаются для перемещения курсора в начало и конец строки соответственно. Клавиша <Ins> выполняет подсветку (выбор) программ для последующего копирования удаления и т. д. Для отмены подсветки достаточно повторно нажать на <Ins>. Работает только с именами файлов и не работает с именами подкаталогов.

Клавиша <Del> служит для удаления символа, на котором находится курсор. Последний при этом остается на прежнем месте, а символы справа от курсора сдвигаются на одну позицию влево. Для перехода в другой подкаталог достаточно подвести «подсветку» (highlight) *к* имени подкаталога (подкаталоги изображаются в окне заглавными буквами) и нажать клавишу <Enter>. После этого в окне будет отображено содержимое данного подкаталога. Содержимое второго окна останется без изменения — это окно *не активно.* Для смены активного окна (т. е. окна, в котором производится работа) достаточно нажать клавишу <TaЬ> или <Ctrl+I>.Эта операция меняет активное окно. Для возврата в подкаталог предыдущего уровня надо подвести подсветку к имени подкаталога предыдущего уровня и нажать <Enter> или <Ctrl+PgUp> (первый способ не срабатывает, если командная строка не пуста, второй же способ игнорирует заполненность командной строки).

Norton Commander позволяет также запустить любые заранее выбранные программы, используя специальное пользовательское меню. Для этого достаточно нажать функциональную клавишу <P2> и выбрать из предлагаемого списка программу.

### *Операции над файлами*

#### *Выделение файлов и каталогов*

Выделение файлов и каталогов используется, как правило, непосредственно перед их копированием, перемещением или удалением.

1. Для выделения (отмены выделения) малых групп или одиночных файлов и каталогов нужно использовать клавишу <Ins> или правую кнопку мыши.

2. Для выделения (отмены выделения) всех файлов в каталоге удобно использовать клавишу <*>, расположенную справа. Для выделения всех файлов также можно использовать комбинацию клавиш <++Enter>, для отмены выделения <-+Enter>. Клавиши <+> и <-> расположены справа. Клавишу <*> можно использовать для инверсии выделения.

3. Для выделения группы файлов с одинаковым расширением нужно нажать клавишу <+>, расположенную справа, затем ввести в окно Выбор файлов вместо правой звездочки расширение файла, например *.txt, затем нажать <Enter>. Для отмены такого выделения использовать <-+Enter>. Аналогично выделяют группу файлов с одинаковым именем, например nc.* в каталоге NC.

*Копирование файлов и каталогов*

\. Открыть каталог с файлами и (или) подкаталогами.

2. Осуществить выделение файлов и каталогов, предназначенных для копирования. Если копируется один файл или один каталог, то на него достаточно поместить курсор.

3. На противоположной панели открыть каталог для приема файлов и подкаталогов или вставить дискету и открыть ее (<Alt+F1> или <Alt+F2>), если копирование производится на дискету.

4. При копировании курсор должен находиться на стороне копируемых файлов или каталогов.

При копировании файлов с дискеты на диск курсор должен находиться со стороны выделенных файлов на дискете, а на диске следует открыть каталог для приема файлов.

5. Нажать клавишу <P5> и <Enter>, если копируются только файлы. При копировании нескольких подкаталогов после нажатия <P5> надо указать мышью Включая подкаталоги и затем <Enter>. Прерывание или отмена копирования — <Esc>.

*Перенос (перемещение) файлов и каталогов*

Отличается от копирования тем, что вместо <P5> надо нажать P6.

*Переименование файлов и каталогов*

1. Установить курсор на файл или каталог.

2. Нажать клавишу <P6> и очистить клавишей <Del> или <BS> окно Переименование файлов.

3. Ввести в окно Переименование файлов новое имя (расширение) файла или каталога и нажать <Enter>.

*Удаление файлов и каталогов*

\. Выделить подлежащие удалению файлы и каталоги и установить курсор со стороны выделенных файлов (каталогов).

3. Нажать клавишу P8, появится окно Удаление файлов и затем нажать <Enter>, если удаляются только файлы. В появившемся втором окне подтверждения удаления указать Все и <Enter>. При удалении одного файла с атрибутом «Архивный» окно подтверждения удаления не появляется.

При удалении нескольких подкаталогов после нажатия <P8> надо указать мышью Включая подкаталоги и затем <Enter>. В появившимся втором окне подтверждения удаления указать Все и <Enter>. Отмена удаления — <Esc>.

*Поиск файлов на диске*

1. Нажать <Alt+F7>. Появляется трафарет «Поиск файла». Отмечаем, если не отмечено, Место поиска — Весь диск С: или Каталоги (вводим имя каталога для поиска). Диск при необходимости можно сменить кнопкой Диск, если поиск проводится не на диске С:.

2. Ввести в окно Найти файлы имя файла или его часть и (или) расширение файла или его часть. Недостающие части имени или расширения заменить знаками <*>, когда число недостающих букв неизвестно, или знаками <?> вместо недостающих букв, когда число недостающих букв известно.

3. Указать мышью кнопку Старт. Повторный поиск — кнопка Новый.

4. Выход из режима поиска файлов — <Esc>.

*Поиск каталогов на диске*

1. После нажатии <Alt+F10> появляется трафарет Дерево каталогов. При первом поиске создается файл treeinfo.ncd со списком всех каталогов на диске С:. Этот файл надо обновить, если он устарел, предварительно удалив его с диска.

2. В окно Поиск ввести первую букву или цифру имени каталога. Если найденный каталог не тот, что нужен, то вводить вторую букву имени и т. д., пока не будет найден искомый каталог. Если буква или цифра не вводится, значит такой каталог отсутствует.

3. Чтобы проверить, нет ли других каталогов с тем же именем, следует нажать <Ctrl+Enter> несколько раз, пока не будут найдены все каталоги.

4. Для перехода в искомый каталог необходимо нажать <Enter>, когда курсор установлен на нем в результате поиска.

5. Выход из режима поиска каталога — <Esc>.

*Быстрый поиск файла в известном каталоге*

1. Поиск файлов будет успешней и быстрее, если предварительно Установить сортировку файлов по имени (<Ctrl+F3>). Открыть каталог с файлами. Если в каталоге есть подкаталоги, то установить курсор на первый файл.

2. Нажать клавишу <Alt> и ввести первую букву (цифру) в имени файла. Курсор переместится на файл, начинающийся с введенной буквы. Если файла на эту букву нет, то буква не будет вводиться. Продолжать ввод при необходимости следующей буквы имени файла, пока не будет найден искомый файл.

*Смена атрибута файла*

Файл может иметь один из следующих четырех атрибутов:

1. Архивный (Archive) — это основной атрибут файлов. При создании файл, как правило, по умолчанию получает этот атрибут.

2. Только для чтения (Read Only) — применяется для защиты файла от случайного удаления. При удалении такого файла будет выдан запрос на удаление. Архивный файл удаляется без запроса.

3. Скрытый (Hidden) — применяется для защиты от удаления особо важных файлов.

4. Системный (System) — аналогичен скрытому файлу.

Для смены атрибута файла используют Меню, Файл, Атрибут, выбирают мышью или пробелом один из указанных атрибутов и устанавливают его (Set). Для смены атрибута одновременно у целой группы файлов ее выделяют, затем Меню, Файл, Атрибут (F9-F-A), устанавливают новый атрибут (Set) и удаляют старый атрибут (Clear).

Для того чтобы сделать видимыми (или невидимыми) скрытые и системные файлы, необходимо вызвать Меню <F9>, пункт Option, Configuration, Show Hidden Files, указать OK.

*Действие Norton Commander при нажатии клавиши <Enter>*

1. Если выделено имя каталога, то при нажатии <Enter> будет выведено оглавление каталога.

2. Если курсор установлен на файл с расширением exe, com, bat, то при нажатии <Enter> начнется исполнение этого файла (запуск программы).

3. Если выделено имя архива (файлы с расширением arj, zip, arc, pak, ice, zoo), то при нажатии <Enter> Norton Commander войдет в архив и выведет оглавление архива. Это свойство NC используется для архивации.

4. Если выделенный файл имеет расширение, указанное в файле nc.ext, то действие Norton Commander при нажатии <Enter> будет определяться содержанием файла nc.ext.

***Структура файла nc.ext и его редактирование.*** В данном файле содержится связь файлов, содержащих данные, с приложениями, обрабатывающими соответствующие файлы, например:

- текстовый файл*.txt — с текстовым редактором;

. файл базы данных *.dbf — с системой управления базами данных;

• программы*.bas, *.pas — со своей средой программирования;

. архивный файл — со своим архиватором.

Тогда при нажатии <Enter> Norton Commander будет автоматически загружать соответствующий выделенный файл в соответствующую ему программу. Для обеспечения такого режима работы служит файл nc.ext. Он находится в каталоге NC со всеми остальными файлами Norton Commander и как любой другой текстовый файл может быть отредактирован в любом текстовом редакторе для DOS, в том числе и в редакторе, встроенном в Norton Commander. Для редактирования файла nc.ext проще всего найти его курсором и нажать <P4>.

В файле nc.ext указывают расширение файла, например txt, а затем командный файл той программы, с которой он связывается, например lexicon. В конце каждой команды ставится !.!

Файл nc.ext может иметь, например, следующую структуру:

txt: lexicon !.!
doc: windows winword !.!
rtt: windows winword!.!
pas: turbo !.!
bas: qbasic !.!
rar: rar en !.!

***Меню пользователя и редактирование файла nc.mpu.*** Меню пользователя Norton Commander значительно облегчает и ускоряет запуск прикладных программ и поэтому широко применяется на практике. Меню пользователя содержится в файле nc.mpu. Если файл nc.mpu находится в каталоге Norton Commander (NC), то такое меню пользователя является Главным (Main).

Главное меню вызывается при нажатии <P2> из любого каталога, если в нем нет своего файла nc.mpu. В последнем случае будет вызываться Локальное меню (когда файл nc.mpu находится в любом каталоге, кроме каталога NC). Файл nc.mpu является текстовым и Редактируется в любом текстовом редакторе, но чаще всего для этого применяется встроенный в Norton Commander редактор.

Сначала в файле nc.mpu указывают ту клавишу (латинскую букву или цифру), которую надо будет нажимать для запуска программы. Затем на этой же строчке после двоеточия дается комментарий к запускаемой программе. На второй строке после двух пробелов записывается команда DOS, в общем случае с указанием полного пути. Иногда продолжение этой команды записывается на третьей, четвертой и т. д. строке, если используется команда DOS cd. Например, при нажатии клавиши W будет запущена Windows.

2. Нажать клавишу <Alt> и ввести первую букву (цифру) в име ни файла. Курсор переместится на файл, начинающийся с введен ной буквы. Если файла на эту букву нет, то буква не будет вводить ся. Продолжать ввод при необходимости следующей буквы имен файла, пока не будет найден искомый файл.

*Смена атрибута файла*

Файл может иметь один из следующих четырех атрибутов:

1. Архивный (Archive) — это основной атрибут файлов. При со здании файл, как правило, по умолчанию получает этот атрибут.

2. Только для чтения (Read Only) — применяется для защиты файла от случайного удаления. При удалении такого файла буде выдан запрос на удаление. Архивный файл удаляется без запроса.

3. Скрытый (Hidden) — применяется для защиты от удаления осо бо важных файлов.

4. Системный (System) — аналогичен скрытому файлу.

Для смены атрибута файла используют Меню, Файл, Атрибут выбирают мышью или пробелом один из указанных атрибутов устанавливают его (Set). Для смены атрибута одновременно у це лой группы файлов ее выделяют, затем Меню, Файл, Атрибут (P9 F-A), устанавливают новый атрибут (Set) и удаляют старый атри бут (Clear).

Для того чтобы сделать видимыми (или невидимыми) скрыты и системные файлы, необходимо вызвать Меню <P9>, пункт Option Configuration, Show Hidden Files, указать OK.

*Действие Norton Commander при нажатии клавиши <Enter>*

1. Если выделено имя каталога, то при нажатии <Enter> буде выведено оглавление каталога.

2. Если курсор установлен на файл с расширением exe, com, bal то при нажатии <Enter> начнется исполнение этого файла (запус программы).

3. Если выделено имя архива (файлы с расширением arj, zip, lzh arc, pak, ice, 200), то при нажатии <Enter> Norton Commander вой дет в архив и выведет оглавление архива. Это свойство NC исполь зуется для архивации.

4. Если выделенный файл имеет расширение, указанное в файл nc.ext, то действие Norton Commander при нажатии <Enter> буде определяться содержанием файла nc.ext.

***Структура файла nc.ext и его редактирование.*** В дан ном файле содержится связь файлов, содержащих данные, с прило жениями, обрабатывающими соответствующие файлы, например:

- текстовый файл*.txt — с текстовым редактором;

, файл базы данных *.dbf — с системой управления базами данных;

• программы*.bas, *.pas — со своей средой программирования;

. архивный файл — со своим архиватором.

Тогда при нажатии <Enter> Norton Commander будет автоматически загружать соответствующий выделенный файл в соответствующую ему программу. Для обеспечения такого режима работы служит файл nc.ext. Он находится в каталоге NC со всеми остальными файлами Norton Commander и как любой другой текстовый файл может быть отредактирован в любом текстовом редакторе для DOS, в том числе и в редакторе, встроенном в Norton Commander. Для редактирования файла nc.ext проще всего найти его курсором и нажать <P4>.

В файле nc.ext указывают расширение файла, например txt, а затем командный файл той программы, с которой он связывается, например lexicon. В конце каждой команды ставится !.!

Файл nc.ext может иметь, например, следующую структуру:

txt: lexicon !.!
doc: windows winword !.!
ПС: windowswinword!.!
pas: turbo !.!
bas: qbasic !.!
rar: rar en !.!

***Меню пользователя и редактирование файла nc.mpu.*** Меню пользователя Norton Commander значительно облегчает и ускоряет запуск прикладных программ и поэтому широко применяется на практике. Меню пользователя содержится в файле nc.mpu. Если файл nc.mpu находится в каталоге Norton Commander (NC), то такое меню пользователя является Главным (Mat).

Главное меню вызывается при нажатии <P2> из любого каталога, если в нем нет своего файла nc.mpu. В последнем случае будет вызываться Локальное меню (когда файл nc.mpu находится в любом каталоге, кроме каталога NC). Файл nc.mpu является текстовым и редактируется в любом текстовом редакторе, но чаще всего для этого применяется встроенный в Norton Commander редактор.

Сначала в файле nc.mpu указывают ту клавишу (латинскую букву или цифру), которую надо будет нажимать для запуска программы. Затем на этой же строчке после двоеточия дается комментарий к запускаемой программе. На второй строке после двух пробелов записывается команда DOS, в общем случае с указанием полного пути. Иногда продолжение этой команды записывается на третьей, четвертой и т. д. строке, если используется команда DOS cd. Например, при нажатии клавиши W будет запущена Windows.

2. Нажать клавишу <Alt> и ввести первую букву (цифру) в имени файла. Курсор переместится на файл, начинающийся с введенной буквы. Если файла на эту букву нет, то буква не будет вводиться. Продолжать ввод при необходимости следующей буквы имени файла, пока не будет найден искомый файл.

*Смена атрибута файла*

Файл может иметь один из следующих четырех атрибутов:

1. Архивный (Archive) — это основной атрибут файлов. При создании файл, как правило, по умолчанию получает этот атрибут.

2. Только для чтения (Read Only) — применяется для защиты файла от случайного удаления. При удалении такого файла будет выдан запрос на удаление. Архивный файл удаляется без запроса.

3. Скрытый (Hidden) — применяется для защиты от удаления особо важных файлов.

4. Системный (System) — аналогичен скрытому файлу.

Для смены атрибута файла используют Меню, Файл, Атрибут, выбирают мышью или пробелом один из указанных атрибутов и устанавливают его (Set). Для смены атрибута одновременно у целой группы файлов ее выделяют, затем Меню, Файл, Атрибут (P9-F-A), устанавливают новый атрибут (Set) и удаляют старый атрибут (Clear).

Для того чтобы сделать видимыми (или невидимыми) скрытые и системные файлы, необходимо вызвать Меню <P9>, пункт Option, Configuration, Show Hidden Files, указать OK.

*Действие Norton Commander при нажатии клавиши <Enter>*

1. Если выделено имя каталога, то при нажатии <Enter> будет выведено оглавление каталога.

2. Если курсор установлен на файл с расширением exe, com, bat, то при нажатии <Enter> начнется исполнение этого файла (запуск программы).

3. Если выделено имя архива (файлы с расширением ad, zip, lzh, arc, pak, ice, zoo), то при нажатии <Enter> Norton Commander войдет в архив и выведет оглавление архива. Это свойство NC используется для архивации.

4. Если выделенный файл имеет расширение, указанное в файле nc.ext, то действие Norton Commander при нажатии <Enter> будет определяться содержанием файла nc.ext.

***Структура файла nc.ext и его редактирование.*** В данном файле содержится связь файлов, содержащих данные, с приложениями, обрабатывающими соответствующие файлы, например:

- текстовый файл *.txt — с текстовым редактором;

. файл базы данных *.dbf — с системой управления базами данных;

• программы *.ba$, *.pa8 — со своей средой программирования;

. архивный файл — со своим архиватором.

Тогда при нажатии <Enter> Norton Commander будет автоматически загружать соответствующий выделенный файл в соответствующую ему программу. Для обеспечения такого режима работы служит файл nc.ext. Он находится в каталоге NC со всеми остальными файлами Norton Commander и как любой другой текстовый файл может быть отредактирован в любом текстовом редакторе для DOS, в том числе и в редакторе, встроенном в Norton Commander. Для редактирования файла nc.ext проще всего найти его курсором и нажать <P4>.

В файле nc.ext указывают расширение файла, например txt, а затем командный файл той программы, с которой он связывается, например lexicon. В конце каждой команды ставится !.!

Файл nc.ex! может иметь, например, следующую структуру:

txt: lexicon !.!

doc: windows winword !.!

ПС:windowswinword!.!

pas: turbo !.!

bas: qbasic !.!

rar: rar en !.!

***Меню пользователя и редактирование файла nc.mnu.*** Меню пользователя Norton Commander значительно облегчает и ускоряет запуск прикладных программ и поэтому широко применяется на практике. Меню пользователя содержится в файле nc.mnu. Если файл nc.mnu находится в каталоге Norton Commander (NC), то такое меню пользователя является Главным (Main).

Главное меню вызывается при нажатии <P2> из любого каталога, если в нем нет своего файла nc.mnu. В последнем случае будет вызываться Локальное меню (когда файл nc.mnu находится в любом каталоге, кроме каталога NC). Файл nc.mnu является текстовым и редактируется в любом текстовом редакторе, но чаще всего для этого применяется встроенный в Norton Commander редактор.

Сначала в файле nc.mnu указывают ту клавишу (латинскую букву или цифру), которую надо будет нажимать для запуска программы. Затем на этой же строчке после двоеточия дается комментарий к запускаемой программе. На второй строке после двух пробелов записывается команда DOS, в общем случае с указанием полного пути. Иногда продолжение этой команды записывается на третьей, четвертой и т. д. строке, если используется команда DOS cd. Например, при нажатии клавиши W будет запущена Windows.

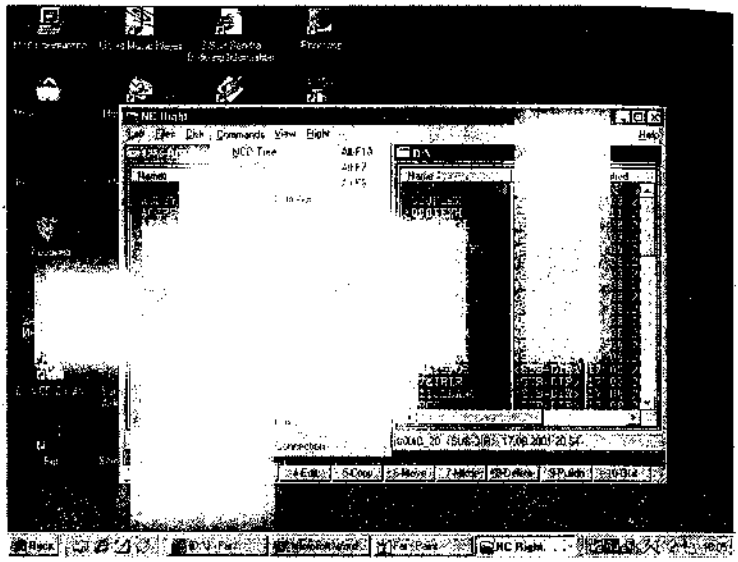


Рис. 4.4. Типичный экран NC for Windows в среде Windows 98. Видны панели файлов и меню Commands

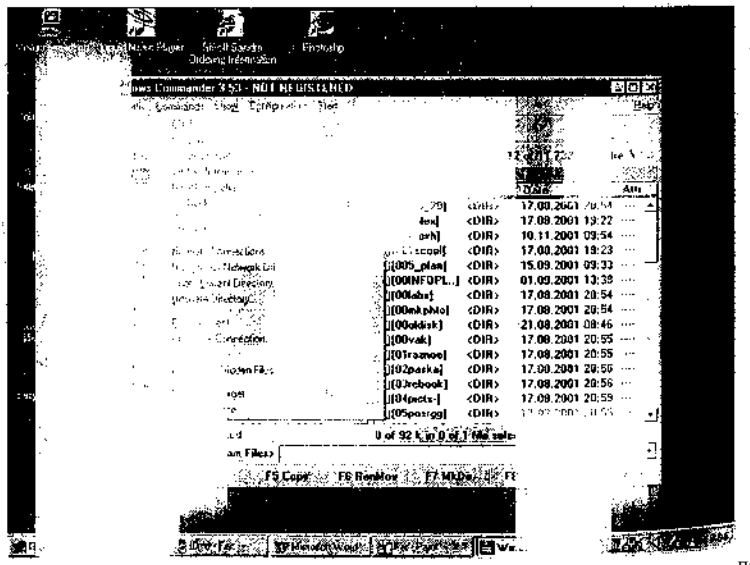


Рис. 4.5. Типичный экран Windows Commander в среде Windows 98. Видны панели файлов и меню Commands .

Структура файла пс.тпи может иметь следующий вид:
W: Windows 3.1.1.
дат
N: Norton Utilities
norton

***Дисковые функции Norton Commander***

1. Копирование дискет (Copy Diskette). Позволяет создавать точные копии дискеты.

2. Форматирование дискет (Format Diskette). Лучше использовать режим безопасного форматирования (Safe Format). При форматировании возможно создание Системной дискеты.

*3.* Создание метки на диске, дискете (Label Disk). Метка на диске (Volume Label) содержит не более 11 символов. Это могут быть латинские буквы, цифры и некоторые спецсимволы.

4. Очистка диска (Disk Cleanup). Позволяет очистить диск от ненужных и временных файлов.

Наиболее распространенными оболочками, унаследовавшими основные черты Norton Commander, являются, по-видимому, Windows Commander, Norton Гог Windows, FAR Manager. Ниже кратко перечислены основные особенности первых двух оболочек, после чего мы подробно остановимся на Far Manager, который является отечественной разработкой.

***Norton for Windows*** (см. рис. 4.4), как это очевидно, использует элементы интерфейса Windows — кнопки, линии прокрутки и пр. Основные особенности:

- поддержка 32-разрядной архитектуры;
- поддерживаются все программы просмотра файлов, характерные для Windows;
- реализована улучшенная функция перемещения файлов и каталогов (буксировка) из разных окон, в том числе и с рабочего стола Windows;
- поддержка длинных имен файлов (LFN);
- улучшенная система меню;
- реализована контекстная подсказка по правой кнопке мыши;
- полная совместимость cWindows 98;
- включена программа-клиент PTP для легкой передачи файлов по данному протоколу.

***Windows Commander*** (см. рис. 4.4—4.7) представляет собой средство манипулирования файлами наподобие Winfile.exe (манипулятор Файлами Windows). Однако он унаследовал все основные черты NC — две панели, меню, команды. Основные возможности:

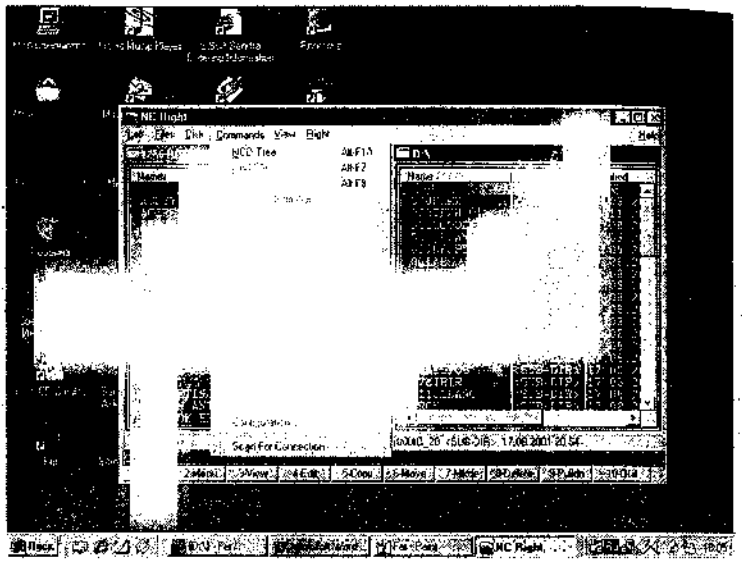


Рис. 4.4. Типичный экран NC for Windows в среде Windows 98. Видны панели файлов и меню Commands

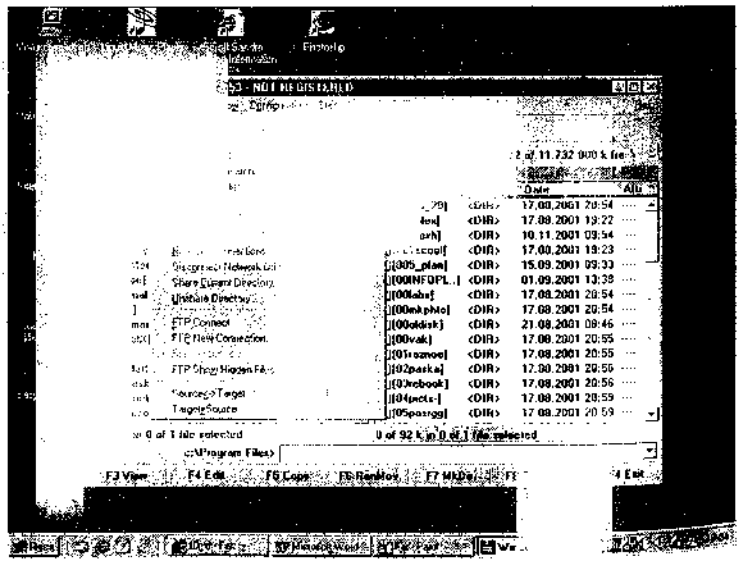


Рис. 4.5. Типичный экран Windows Commander в среде Windows 98. Видны панели файлов и меню Commands .

Структура файла пс.тпи может иметь следующий вид:
W: Windows 3.1.1.
win
N: Norton Utilities
norton

***Дисковые функции Norton Commander***

1. Копирование дискет (Copy Diskette). Позволяет создавать точные копии дискеты.

2. Форматирование дискет (Format Diskette). Лучше использовать режим безопасного форматирования (Safe Format). При форматировании возможно создание Системной дискеты.

3. Создание метки на диске, дискете (Label Disk). Метка на диске (Volume Label) содержит не более 11 символов. Это могут быть латинские буквы, цифры и некоторые спецсимволы.

4. Очистка диска (Disk Cleanup). Позволяет очистить диск от ненужных и временных файлов.

Наиболее распространенными оболочками, унаследовавшими основные черты Norton Commander, являются, по-видимому, Windows Commander, Norton Гог Windows, FAR Manager. Ниже кратко перечислены основные особенности первых двух оболочек, после чего мы подробно остановимся на Раг Manager, который является отечественной разработкой.

***Norton for Windows*** (см. рис. 4.4), как это очевидно, использует элементы интерфейса Windows — кнопки, линии прокрутки и пр. Основные особенности:

- поддержка 32-разрядной архитектуры;
- поддерживаются все программы просмотра файлов, характерные для Windows;
- реализована улучшенная функция перемещения файлов и каталогов (буксировка) из разных окон, в том числе и с рабочего стола Windows;
- поддержка длинных имен файлов (LFN);
- улучшенная система меню;
- реализована контекстная подсказка по правой кнопке мыши;
- полная совместимость cWindows 98;
- включена программа-клиент PTP для легкой передачи файлов по данному протоколу.

***Windows Commander*** (см. рис. 4.4—4.7) представляет собой средство манипулирования файлами наподобие Winfile.exe (манипулятор Файлами Windows). Однако он унаследовал все основные черты NC — две панели, меню, команды. Основные возможности:

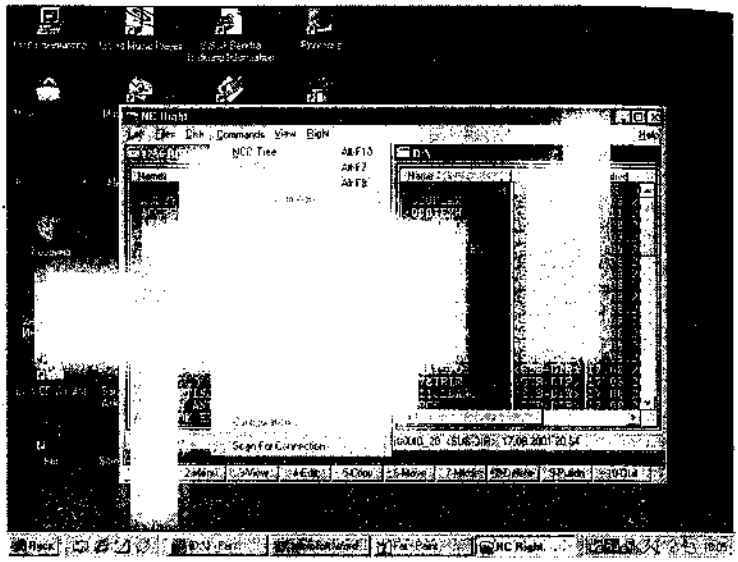


Рис. 4.4. Типичный экран NC for Windows в среде Windows 98. Видны панели файлов и меню Commands

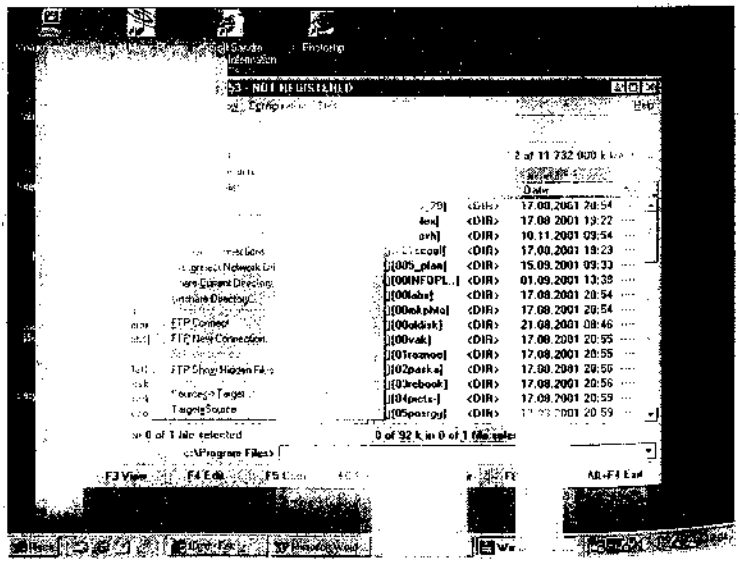


Рис. 4.5. Типичный экран Windows Commander в среде Windows 98. Видны панели файлов и меню Commands

Структура файла пс.тпи может иметь следующий вид:
W: Windows 3.1.1.
win
N: Norton Utilities
norton

***Дисковые функции Norton Commander***

1. Копирование дискет (Copy Diskette). Позволяет создавать точные копии дискеты.

2. Форматирование дискет (Format Diskette). Лучше использовать режим безопасного форматирования (Safe Format). При форматировании возможно создание Системной дискеты.

*3.* Создание метки на диске, дискете (Label Disk). Метка на диске (Volume Label) содержит не более 11 символов. Это могут быть латинские буквы, цифры и некоторые спецсимволы.

4. Очистка диска (Disk Cleanup). Позволяет очистить диск от ненужных и временных файлов.

Наиболее распространенными оболочками, унаследовавшими основные черты Norton Commander, являются, по-видимому, Windows Commander, Norton Tor Windows, FAR Manager. Ниже кратко перечислены основные особенности первых двух оболочек, после чего мы подробно остановимся на Far Мапа§ег, который является отечественной разработкой.

***Norton for Windows*** (см. рис. 4.4), как это очевидно, использует элементы интерфейса Windows — кнопки, линии прокрутки и пр. Основные особенности:

- поддержка 32-разрядной архитектуры;
- поддерживаются все программы просмотра файлов, характерные для Windows;
- реализована улучшенная функция перемещения файлов и каталогов (буксировка) из разных окон, в том числе и с рабочего стола Windows;
- поддержка длинных имен файлов (LFN);
- улучшенная система меню;
- реализована контекстная подсказка по правой кнопке мыши;
- полная совместимость с Windows *98;*
- включена программа-клиент PTP для легкой передачи файлов по данному протоколу.

***Windows Commander*** (см. рис. 4.4—4.7) представляет собой средство манипулирования файлами наподобие Winfile.exe (манипулятор файлами Windows). Однако он унаследовал все основные черты NC — две панели, меню, команды. Основные возможности:

- поддержка функций перемещения файлов с помощью мыши (буксировка), включая и вывод на печать;
- копирование, перемещение, переименование, удаление «деревьев каталогов», включая удаление непустых каталогов;
- обработка архивов как субдиректорий — автоматически подключаются архиваторы pkzip, arj, lha, гаг, ис2 и асе. Встроенная распаковка архивов форматов ARJ, LZH, G2 и TAK. Встроенная упаковка архивов ZIP на основе метода ZLib (Jean-loup Gailly);
- встроенный просмотр (<F3>) файлов любой длины в шестнадцатеричном, двоичном коде, а также ASCII (DOS) или ANSI (Windows);
- конфигурирование запуска внешних программ или внутренних команд;
- поддержка FTP-клиента и т. д.

На рис. 4.6 как раз отображен экран настройки на связь с FTP-сервером. Конфигурации каждой настройки запоминаются в меню *Connection* и включают в себя:

- адрес FTP-сервера (здесь — **ftp.inion.ru**);
- имя пользователя и пароль (здесь — **anonymous**);

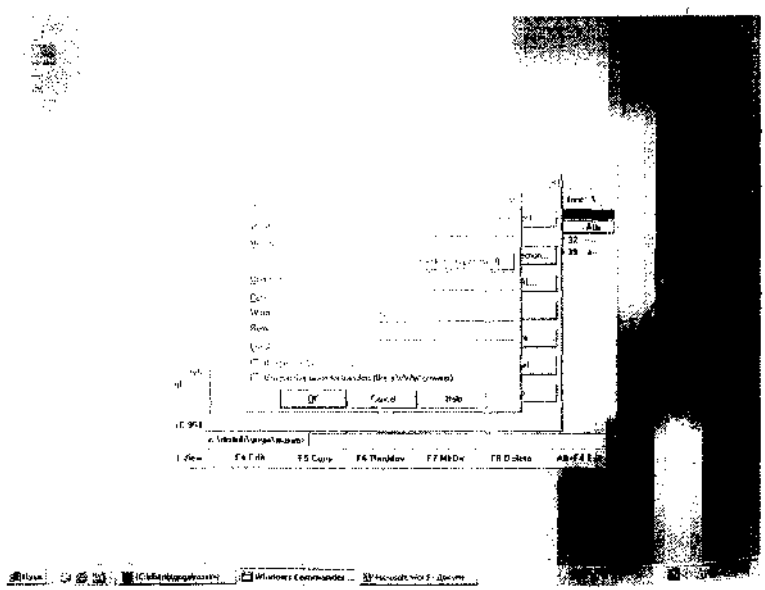
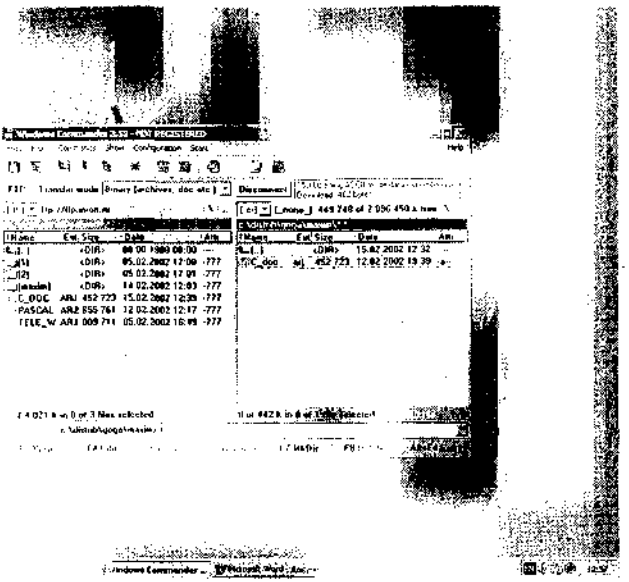


Рис. 4.6. Настройка FTP-клиента Windows Commander на связь с сервером



7. Режим связи с FT

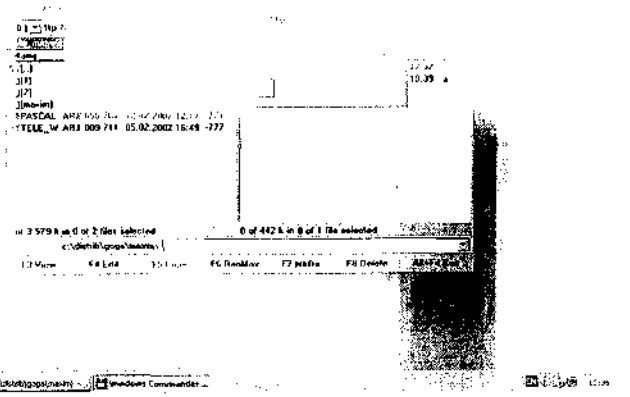


Рис. 4.8. Windows Commander — режим передачи файла (Upload) из локального каталога в удаленный

- поддержка функций перемещения файлов с помощью мыши (буксировка), включая и вывод на печать;
- копирование, перемещение, переименование, удаление «деревьев каталогов», включая удаление непустых каталогов;
- обработка архивов как субдиректорий — автоматически подключаются архиваторы pkzip, arj, lha, гаг, ис2 и асе. Встроенная распаковка архивов форматов ARJ, LZH, G2 и TAK. Встроенная упаковка архивов ZIP на основе метода ZLib (Jean-Loup Gailly);
- встроенный просмотр (<F3>) файлов любой длины в шестнадцатеричном, двоичном коде, а также ASCII (DOS) или ANSI (Windows);
- конфигурирование запуска внешних программ или внутренних команд;
- поддержка FTP-клиента и т. д.

На рис. 4.6 как раз отображен экран настройки на связь с FTP-сервером. Конфигурации каждой настройки запоминаются в меню *Connection* и включают в себя:

- адрес FTP-сервера (здесь — **ftp.inion.ru**);
- имя пользователя и пароль (здесь — **anonymous**);

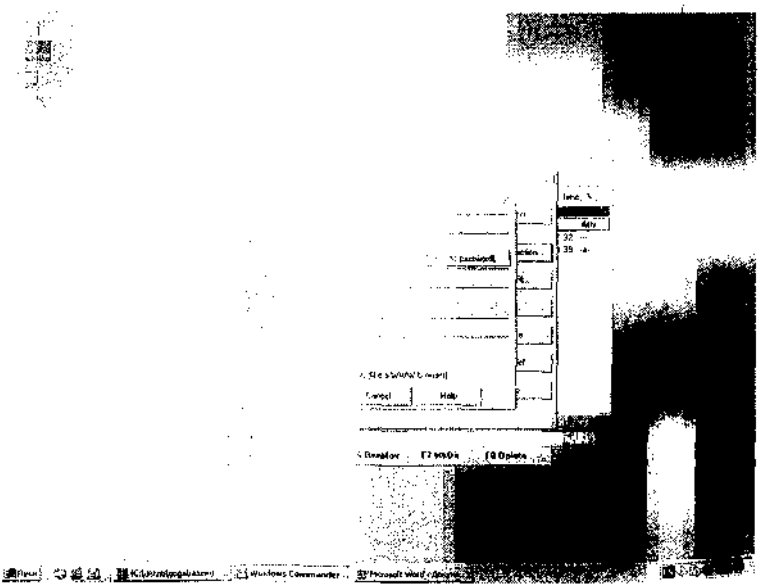


Рис. 4.6. Настройка FTP-клиента Windows Commander на связь с сервером

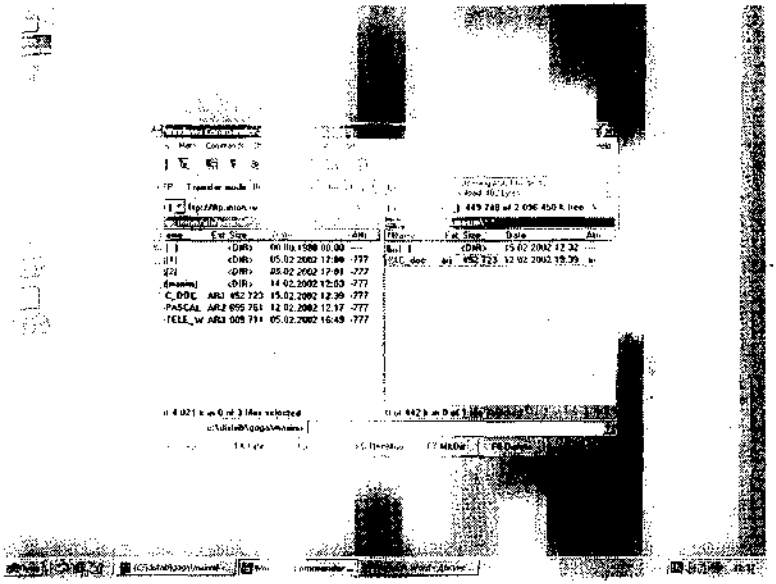


Рис. 4.7. Режим связи с FTP-сервером

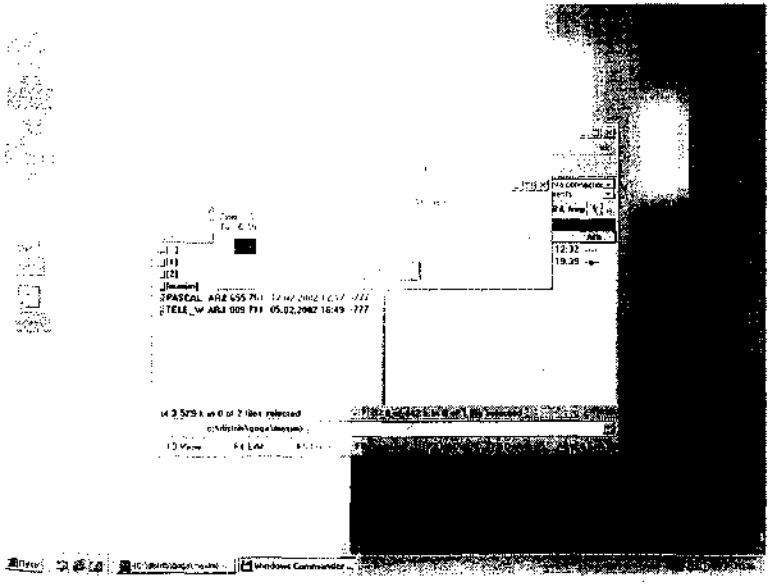


РИС. 4.8. Windows Commander — режим передачи файла (Upload) из локального каталога в удаленный

- поддержка функций перемещения файлов с помощью мыши (буксировка), включая и вывод на печать;
- копирование, перемещение, переименование, удаление «деревьев каталогов», включая удаление непустых каталогов;
- обработка архивов как субдиректорий — автоматически подключаются архиваторы pkzip, arj, lha, rar, uc2 и ace. Встроенная распаковка архивов форматов ARJ, LZH, G2 и TAK. Встроенная упаковка архивов ZIP на основе метода ZLib (Jean-loup Gailly);
- встроенный просмотр (<F3>) файлов любой длины в шестнадцатеричном, двоичном коде, а также ASCII (DOS) или ANSI (Windows);
- конфигурирование запуска внешних программ или внутренних команд;
- поддержка FTP-клиента и т. д.

На рис. 4.6 как раз отображен экран настройки на связь с FTP-сервером. Конфигурации каждой настройки запоминаются в меню *Connection* и включают в себя:

- адрес FTP-сервера (здесь — **ftp.inion.ru**);
- имя пользователя и пароль (здесь — **апопутои$);**

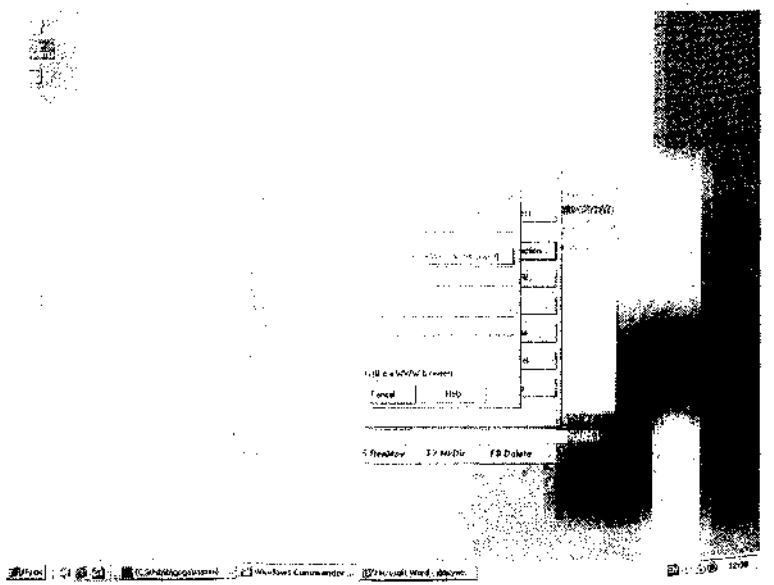


Рис. 4.6. Настройка FTP-клиента Windows Commander на связь с сервером

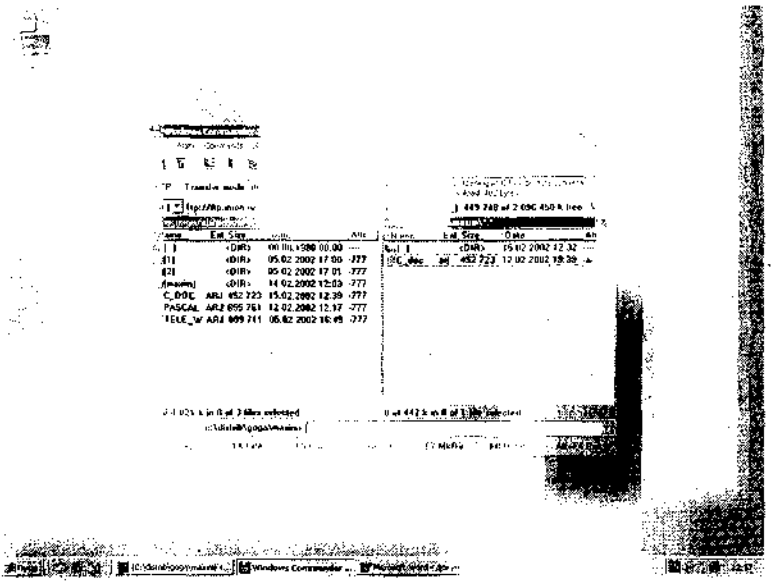


Рис. 4.7. Режим связи с FT

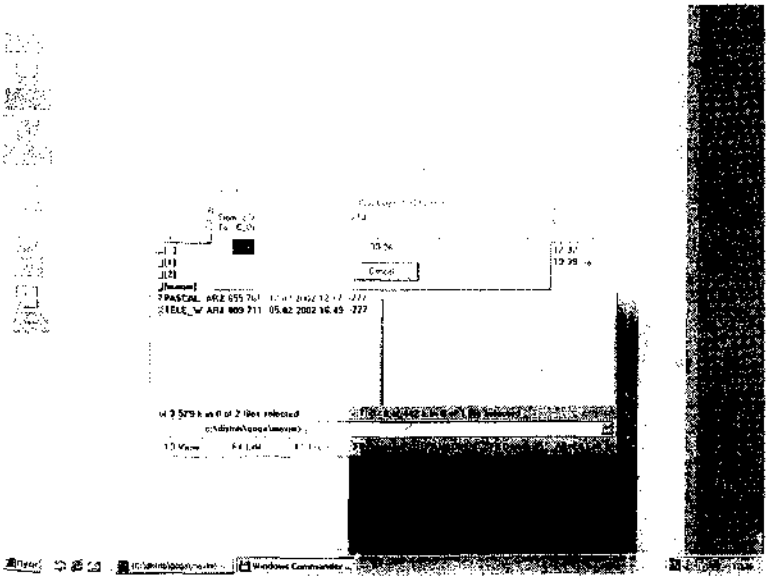


Рис. 4.8. Windows Commander — режим передачи файла (Upload) из локального каталога в удаленный

- имя удаленного каталога в файловой системе FTP (здесь w_inion/irb).

После установления связи на одной из панелей отображается удаленный каталог (рис. 4.8 — слева удаленный каталог, справа — локальный).

Передача файлов в обе стороны (Upload и Download) осуществляется обычным выделением файлов (директорий) и копирования их по команде <F5> (рис. 4.7).

## 4.4. FAR Мападег — текстовая оболочка для Windows 95/98/NT/2000

FAR — это работающая в текстовом режиме программа управления файлами для Windows 95, 98 и Windows NT, с поддержкой *длинных имен файлов* и широким набором операций над файлами и папками.

FAR позволяет работать с архивами. При этом файлы в архивах обрабатываются аналогично файлам в папках. FAR сам преобразует команды в соответствующие вызовы внешних архиваторов. FAR также обеспечивает значительное количество сервисных функций (см. также Приложение 5).

***Параметры командной строки.*** В командной строке при запуске Far Manager могут быть использованы некоторые ключи и параметры (табл. 4.1, табл. П4.1).

*Таблица 4.1.* **Ключи и параметры командной строки Far**

| | | |
|---|---|---|
| /a | Запрет отображения символов с кодами 0–31 и 255 | Может быть полезно при запуске FAR из telnet |
| /ад | Запрет отображения псевдографических символов | |
| /e[<строка>[:<позиция>]] <filename> | Редактирование указанного файла | После /e можно дополнительно указать строку и позицию в строке, устанавливаемые после запуска редактора. Например: far /e70:2 readme |
| /i | Установить маленькую (16x16) иконку для окна консоли FAR | В некоторых конфигурациях эта опция может привести к нестабильной работе |
| /u<username> | Позволяет использовать раздельные настройки для различных пользователей | Например: far /u guest |

*Продолжение табл. 4.1*

| | | |
|---|---|---|
| /v<filename> | Просмотр указанного файла | Если в качестве <filename> использовано -, данные будут читаться с stdin. Например «dir\|far /v -» выведет результат работы команды dir |
| Par<filename>» | Запускает FAR в режиме просмотра содержимого <filename>, если <filename> - архив, или в режиме просмотра содержимого папки, если <filename> - папка | |

***Некоторые общие понятия и операции.*** Прежде чем перейти к описанию команд панелей — основного рабочего инструмента, перечислим некоторые общие понятия.

*Маски файлов.* Маски файлов часто используются в командах PAK для выбора отдельных файлов и папок или их групп. Маски могут включать обычные допустимые в именах файлов символы, '*' и '?', а также специальные выражения:

* — любое количество символов;

? — любой символ;

[c,x — z] — любой символ из находящихся в квадратных скобках. Допускаются и отдельные символы, и их диапазоны.

Например, файлы ftp.exe, fc.exe и f.ext могут быть выбраны с помощью маски f*.ex?, маска *co* выберет и color.ini, и edit.com, маска [c — f,t]*.txt может выбрать config.txt, demo.txt, faq.txt и tips.txt.

Во многих командах FAR можно задать несколько разделенных запятыми масок. Например, чтобы выбрать все документы, можно ввести *.doc,*.txt,*.wri в команде Пометить группу.

*Пометка файлов.* Для обработки файлов и папок панели файлов они могут быть выбраны несколькими различными способами.

<Ins> помечает файл под курсором и перемещает курсор вниз.

<Shift+Клавиши курсора> позволяют перемещать курсор в различных направлениях.

<Num+> и <Num-> выбирают или снимают пометку с группы с использованием одной или нескольких разделенных запятыми масок файлов.

<Num*> инвертирует текущую пометку.

Команда Восстановить пометку (<Ctr+M>) восстанавливает выбранную до этого группу.

<Ctrl+Num+> и <Ctrl+Num -> выбирают или снимают пометку со всех файлов с тем же расширением, что и у файла под курсором.

<Alt+Num+> и <Alt+Num -> выбирают или снимают пометку со всех файлов с тем же именем, что и у файла под курсором.

<Ctrl+Num*> инвертирует текущую пометку, включая папки. Если параметр Пометка папок в диалоге Настроек панели включена, это работает аналогично <Num*>.

<Shift+Num+> и <Shift+Num-> выбирают или снимают пометку со всех файлов.

Если ни один файл не выбран, то будет обработан только файл под курсором.

*Копирование, перенос, переименование и создание связей.* Следующие команды могут быть использованы для копирования, переноса и переименования файлов и папок:

Копировать выбранные файлы — <P5>.

Копировать файл под курсором вне зависимости от пометки — <Shift+F5>.

Переименовать или перенести выбранные файлы — <P6>.

Переименовать или перенести файл под курсором вне зависимости от пометки — <Shift+F6>.

На разделах NTFS также можно создавать жесткие связи файлов с помощью команды <Alt+F6>, так что вы можете иметь несколько различных имен файлов, ссылающихся на одни и те же данные.

Если необходимо создать папку назначения перед копированием, следует добавить к ее имени обратную черту. Также в диалоге копирования можно нажать <P10> для выбора папки из дерева активной файловой панели или <Alt+F10> для выбора из дерева пассивной файловой панели.

Возможность копирования, переноса и переименования файлов для подключаемых модулей зависит от функциональности конкретного модуля.

Если файл, в который производится копирование, уже существует, то он может быть перезаписан, пропущен, либо содержимое копируемого файла может быть дописано в его конец.

Если диск, на который производится копирование или перенос файлов, в ходе операции заполнился, то можно либо отменить операцию, либо заменить диск и выбрать пункт Разделить, после чего копируемый файл будет разделен между дисками. Эта функция доступна только при выключенном параметре «Использовать системную функцию копирования» из диалога Системные параметры.

Параметр «Копировать права доступа» может использоваться только для файловой системы NTFS и позволяет копировать информацию о правах доступа к файлу.

Параметр «Использовать системную функцию копирования» из диалога Системные параметры включает использование функции Windows CopyFileEx (или CopyFile, если CopyPHeEx недоступна) вместо внутренней реализации копирования файлов. Это может быть полезно на NTFS, так как CopyPHeEx выполняет более эффективное распределение дискового пространства и копирует расширенные атрибуты файлов.

*Перетаскивание файлов.* Операции копирования и переноса файлов могут быть выполнены с помощью «перетаскивания» (Drag and Drop, буксировка). Нажмите левую кнопку мыши на исходном файле или папке, перетащите его на другую панель и отпустите кнопку мыши.

Если вы хотите обработать группу файлов или папок, пометьте их перед перетаскиванием, нажмите левую кнопку мыши на исходной панели и перетащите файлы на другую панель.

Вы можете переключаться между копированием и переносом, нажимая правую кнопку мыши во время перетаскивания. Также для переноса файлов вы можете удерживать клавишу <Shift> в момент нажатия левой кнопки мыши.

*Меню выбора диска.* Это меню позволяет сменить текущий диск панели, отсоединиться от сетевого диска или открыть новую панель подключаемого модуля.

Необходимо выбрать пункт меню с соответствующей буквой диска для смены текущего диска или пункт с названием модуля, чтобы создать новую панель модуля. Если панель не является панелью файлов, ее тип будет изменен на панель файлов.

Для отсоединения от сетевого диска можно использовать <Del>. <Ctrl+1> — <Ctrl+8> переключают отображение различной информации:

<Ctrl+1> — тип диска;

<Ctrl+2> — сетевое имя (и путь, ассоциированный с SUBST диском под NT);

<Ctrl+3> — метка диска;

<Ctrl+4> — файловая система;

<Ctrl+5> — общее и свободное место на диске;

<Ctrl+6> — показ параметров сменных дисков;

<Ctrl+7> — показ имен подключаемых модулей;

<Ctrl+8> — показ параметров компакт-дисков.

Настройки меню выбора диска сохраняются в конфигурации PAK.

***Панели.*** (См. также табл. П4.2.). Обычно PAK показывает две панели (левое и правое окна) с различной информацией:

- панель файлов (рис. 4.9, а);
- панель информации (рис. 4.9, в);
- панель дерева папок (рис. 4.10, а);
- панель быстрого просмотра.

Если необходимо изменить режим просмотра панели, следует выбрать желаемый из Меню Панелей. После смены режима просмотра или текущего диска тип любой панели автоматически меняется на панель файлов.

Панель файлов. Отображает содержимое текущей папки. Вы можете выбирать файлы и папки, выполнять различные файловые и архивные операции. По умолчанию в панели файлов используются следующие режимы просмотра:

- краткий — имена файлов выводятся в три колонки;
- средний — имена файлов выводятся в две колонки;
- полный — выводятся имя, размер, дата и время создания (модификации) файла;
- широкий — выводятся имя и размер файла;
- детальный — выводятся имена, размеры, упакованные размеры, время последней модификации, создания, доступа и атрибуты файла.

Упакованные размеры имеют смысл для файлов с атрибутом «Сжатый» на дисках с файловой системой NTFS или для файлов внутри архивов. Владельцы и количество связей файлов тоже применимы только к разделам NTFS. Некоторые файловые системы могут не поддерживать время создания и время доступа к файлу.

Для позиционирования на файл можно воспользоваться операцией быстрого поиска по первым буквам имени. Для этого, удерживая клавишу <Alt>, набирайте имя требуемого файла, пока на него не переместится курсор. С помощью <Ctrl+Enter> можно переместиться на следующее имя, соответствующее введенной строке. Кроме обычных символов, в имени файла также можно использовать символы '*' и '?'.

Панель дерева папок. Отображает структуру папок текущего диска в виде дерева. Это позволяет быстро сменить текущую папку, а также выполнять операции над папками.

PAK запоминает информацию о структуре папок в файле Tree.Par, расположенном в корневой папке каждого диска. Если за-

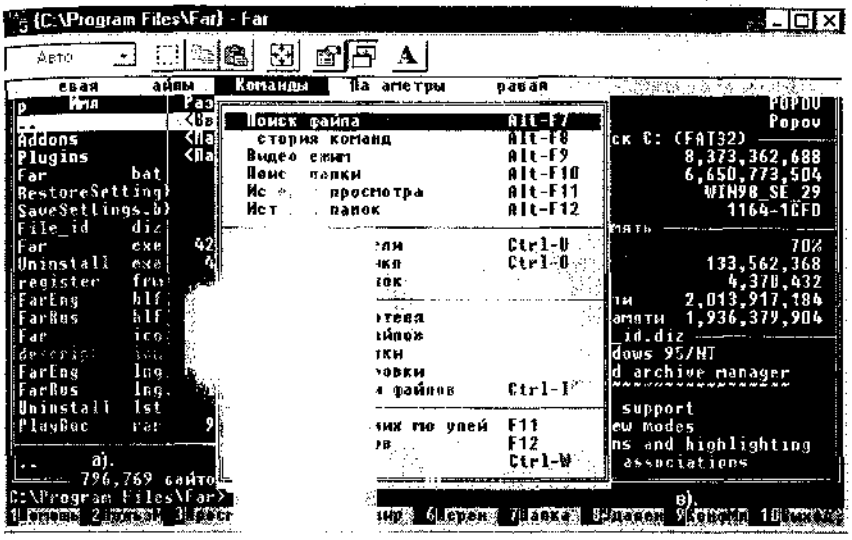



Рис. 4.9. Панель файлов (а), меню команд (б), панель информации (в)

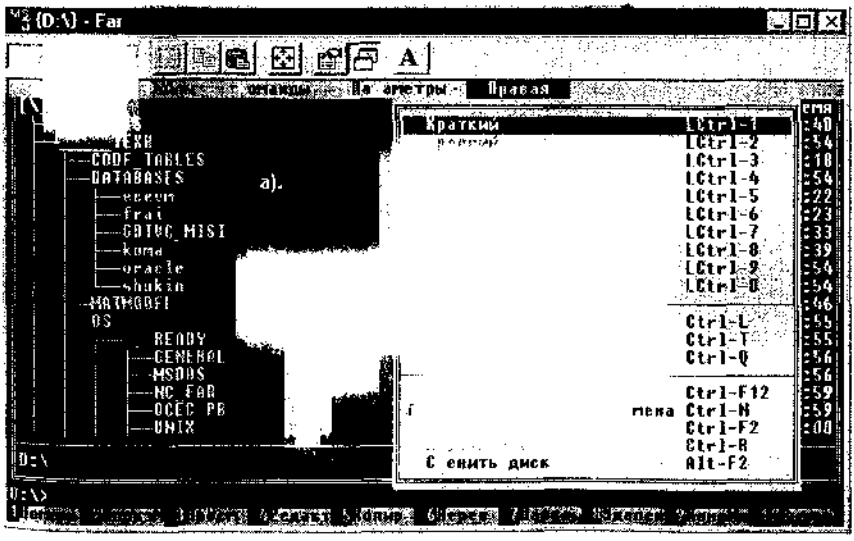



Рис. 4.10. Дерево папок (а) и меню правой панели (б)

пись на диск невозможна, то эта информация сохранятся в скрытой папке Tree.Cache, находящейся в той же папке, что и Рат.exe.

Для позиционирования на папку можно использовать операцию быстрого поиска. Для этого, Удерживая клавишу АН, набирайте имя требуемой папки, пока на нее не переместится курсор. С помощью <Ctrl+Enter> можно переместиться на следующее имя, соответствующее введенной строке.

Панель информации. Панель информации содержит следующие данные:

- сетевые имена компьютера и пользователя;
- имя и тип текущего диска, тип его файловой системы, сетевое имя, общий и свободный размеры, метку тома и серийный номер;
- Уровень загрузки памяти (100 % означает, что использована вся память) общий и свободный размеры физической и виртуальной памяти;
- файл описания папки, можно просмотреть содержимое этого файла в полноэкранном режиме, нажав РЗ или левую кнопку мыши. Для редактирования или создания этого файла нажать <F4> или правую кнопку мыши.

Список возможных имен файлов описания папок может быть задан с помощью команды Файлы описания папок в Меню параметров.

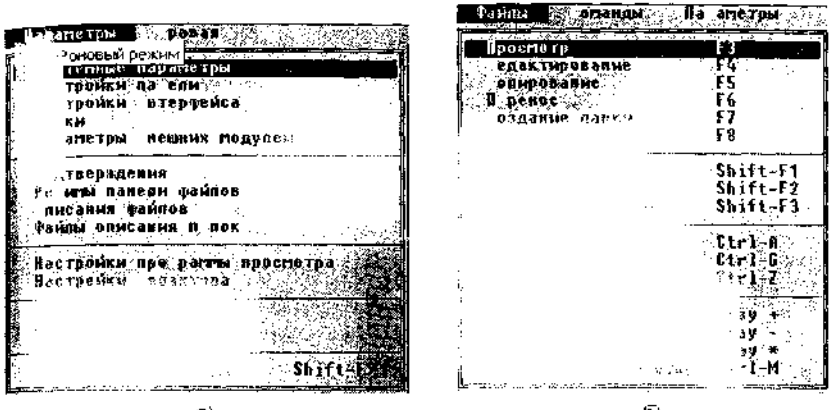
Панель быстрого просмотра используется для получения информации о выбранном элементе панели файлов или дерева папок.

Если выбранный элемент — Файл, то отображается его содержимое. Для известных Windows типов файлов также выводится название типа. Для папок в панели быстрого просмотра сообщается общий размер, общий упакованный размер, количество файлов и вложенных папок, размер кластера текущего диска, реальный размер файлов, включая недоиспользованные Фрагменты кластеров. Общий упакованный размер применим только для дисков с файловой системой NTFS.

Рассмотрим далее основные рубрики меню Рат Manager. Это:

- меню панелей (рис- 4.10, б);
- меню параметров (рис. 4.11, *а);*
- меню файлов (рис. 4.11, *б);*
- меню команд (рис. 4.9, б).

Для активизации меню можно использовать <Р9> или нажать кнопку мыши на верхней строке экрана. Комбинация <Shift+F10> позволяет выбрать последний использованный пункт меню.




а) б)

Рис. 4.11. Меню параметров (*а*) и меню файлов (*б*)

***Меню левой и правой панелей.*** Меню «Левая» и «Правая» позволяют изменить параметры левой и правой панели соответственно. Эти меню включают следующие пункты (рис. 4.10, б):

- краткий — показывать файлы в три колонки;
- средний — показывать файлы в две колонки;
- полный — показывать имя, размер, дату и время файла;
- широкий — показывать имя и размер файла;
- детальный — показывать имя, размер, упакованный размер, время модификации, создания и доступа и атрибуты файла, полноэкранный режим;
- описания — имя и описание файла;
- длинные описания — имя, размер и описание файла, полноэкранный режим;
- владельцы файлов — имя, размер и владелец файла;
- связи файлов — имя, размер и количество жестких связей файлов;
- альтернативный — имя, размер (форматированный с использованием запятых) и дата создания файла;
- панель информации — сменить панель на панель информации;
- дерево папок — сменить панель на дерево папок;
- быстрый просмотр — сменить панель на панель быстрого просмотра;
- режимы сортировки — показать доступные режимы сортировки;

- показывать длинные имена — показывать длинные/короткие имена;
- панельвкл/выкл — показать/спрятать панель;
- перечитать — перечитать содержимое панели;
- сменить диск — сменитьтекущий диск.

***Меню параметров***

- системные параметры — вызывает диалог" настройки системных параметров;
- настройки панели — вызывает диалог настроек панели;
- настройки интерфейса — вызывает диалог настроек интерфейса;
- языки — выбор основного языка и языка помощи, следует использовать команду «сохранить параметры», чтобы сохранить выбранные языки;
- параметр — настройка параметров подключаемых внешних модулей;
- подтверждения — включение или выключение подтверждений для некоторых операций;
- режимы панели файлов — настройка режимов просмотра панели файлов;
- описания файлов — режимы обновления и имена описаний файлов;
- файлы описания — имена или маски файлов, отображаемых папок в панели информации в качестве описания папки;
- настройки программы просмотра — настройки внешней программы просмотра;
- настройки редактора — настройки внешнего и встроенного редактора;
- цвета — изменение цвета различных элементов интерфейса или изменение всей палитры цветов либо на черно-белую, либо на предлагаемую по умолчанию;
- раскраска файлов — редактирование раскраски файлов;
- сохранить параметры — сохранение текущей конфигурации, цветов и внешнего вида экрана.

***Настройка некоторых системных параметров***

Подтверждения. В диалоге Подтверждения можно разрешить или запретить подтверждения для следующих операций:

- перезаписывание файлов назначения во время копирования файлов;
- перезаписывание файлов назначения во время переноса файлов,
- перетаскивание (Drag and Drop) файлов;

- удаление файлов;
- удаление папок;
- выход из PAK.

Системные параметры:

- снимать атрибут R/O с CD файлов — снимать атрибут «только для чтения» с файлов, копируемых с CD;
- удалять в корзину — разрешает удаление файлов с использованием корзины (Recycle Bin);
- использовать системную функцию копирования — использовать функцию Windows CopyFileEx (или CopyFile, если CopyFileEx недоступна) вместо внутренней реализации копирования файлов. Это может быть полезно на NTFS, так как CopyFileEx выполняет более эффективное распределение дискового пространства и копирует расширенные атрибуты файлов;
- копировать открытые для записи файлы — позволяет копировать файлы, которые открыты для записи другими программами. Этот режим удобен, когда нужно скопировать открытый в течение долгого времени файл, но он может быть опасным, если этот файл модифицируется одновременно с копированием;
- создавать папки заглавными буквами — если имя новой папки содержит только строчные буквы и установлена эта опция, то папка будет создана заглавными буквами;
- время бездействия — завершает работу PAK, если в течение указанного интервала не было нажатий клавиш мыши или клавиатуры, PAK ожидал ввода из командной строки и отсутствовали фоновые экраны редактирования или просмотра;
- сохранять историю команд — вызывает сохранение истории команд перед завершением и ее восстановление после запуска PAK;
- сохранять историю папок — вызывает сохранение истории папок перед завершением и ее восстановление после запуска PAK. Для просмотра содержимого истории папок нажать <Alt+F12>;
- сохранять историю просмотра и редактора — вызывает сохранение истории просмотра и редактора перед завершением и ее восстановление после запуска PAK. Для просмотра входящих в этот список файлов нажать <Alt+F11>;
- использовать стандартные типы файлов — если это опция включена, то при нажатии <Enter> на файле, тип которого известен Windows и отсутствует в Ассоциациях файлов PAK, бу-

дет запущена программа Windows, предназначенная для обработки этого типа файлов;

• автозапись конфигурации — если эта опция включена, FAR будет автоматически сохранять конфигурацию. Также будут сохраняться текущие папки обеих панелей.

Настройки панели:

- показывать скрытые и системные файлы — разрешает показ файлов с атрибутами *скрытый* и *системный.* Этот режим также может быть переключен с помощью <Ctrl+H>;
- раскраска файлов — разрешает раскраску файлов;
- автосмена папки — если эта опция включена, то передвижения курсора по дереву папок будут вызывать смену папки в другой панели. Если эта опция выключена, то для смены папки из дерева папок необходимо нажать Enter;
- пометка папок — разрешает пометку папок с использованием <Num+> и <Num*>. В противном случае эти команды работают только с файлами;
- разрешить обратную сортировку — если эта опция включена и текущий режим сортировки файловой панели выбран повторно, то будет установлен режим обратной сортировки;
- показывать заголовки колонок — разрешает показ заголовков колонок панели файлов;
- показывать строку статуса — разрешает показ строки статуса в панели файлов;
- показывать суммарную информацию — разрешает показ суммарной информации в нижней строке панели файлов;
- показывать свободное место — разрешает показ свободного места на текущем диске;
- показывать полосу прокрутки — разрешает показ полосы прокрутки в панели файлов и панели дерева папок;
- показывать количество фоновых экранов — разрешает показ количества фоновых экранов;
- показывать букву режима сортировки — показывать текущий режим сортировки в верхнем левом углу панели;

Раскраска файлов. Диалог Раскраски файлов в Меню параметров позволяет определить группы раскраски файлов. Каждое определение группы включает:

- одну или несколько разделенных запятыми масок файлов;
- атрибуты включения;
- атрибуты исключения;

- цвета обычного имени, помеченного имени, имени под курсором и помеченного имени под курсором. Если вы хотите использовать цвет по умолчанию, установите цвет в «Черный на черном»;
- опционально может быть указан любой символ для обозначения принадлежащих к группе файлов. Он может быть использован как вместе с цветовым выделением, так и вместо него.

Файл принадлежит к группе раскраски, если:

- его имя соответствует хотя бы одной маске;
- он имеет все атрибуты включения;
- он не имеет атрибутов исключения.

Группы раскраски анализируются от начала к концу. Если обнаружено, что файл принадлежит к какой-либо группе, то принадлежность к остальным группам не проверяется.

Настройка режимов просмотра панели файлов. Панель файлов может выводить информацию, используя 10 заданных заранее режимов: краткий, средний, полный, широкий, детальный, описания, длинные описания, владельцы файлов, связи файлов, альтернативный полный. Обычно этого достаточно, но, тем не менее, при желании можно изменить параметры этих режимов или даже полностью заменить их на новые.

Команда Режимы панели файлов из Меню параметров позволяет изменить параметры режимов просмотра. Сначала она предлагает выбрать требуемый режим из списка. В этом списке режим 0 соответствует режиму просмотра, вызываемому по <ЛевыйCtrl+0> (альтернативный полный), режим 1 соответствует краткому режиму (<ЛевыйCtrl+1>), режим 2 соответствует среднему режиму (<ЛевыйCtrl+2>) и так далее. После выбора режима появляется диалог, в котором можно изменить следующие параметры:

- типы колонок — типы колонок кодируются с помощью одной или нескольких букв, разделенных запятыми. Допускаются следующие типы колонок:

  N[M,O,R] — имя файла,

  где M — показывать символы пометки; O — показывать имена без путей (предназначено в основном для подключаемых модулей); R — выравнивать имена по правому краю (эти символы можно комбинировать, например NMR);

  S[C,T] — размер файла,

  где C — форматировать размер файла запятыми; T — использовать 1000 вместо 1024 как делитель, если ширины колонки не хватает для показа полного размера файла;

P[C,T] — упакованный размер файла,
где C — форматировать размер файла запятыми; T — использовать 1000 вместо 1024 как делитель, если ширины колонки не хватает для показа полного размера файла;
D — дата модификации файла;
T — время модификации файла;
DM[B,M] — дата и время модификации файла;
DC[B,M] — дата и время создания файла;
DA[B,M] — дата и время последнего доступа к файлу,
где B — краткий (в стиле Unix) формат времени файла; M — использование текстовых имен месяцев;
A — атрибуты файла;
2 — описание файла;
O — владелец файла;
L IV— количество жестких связей.
Если описание типов колонок содержит более одной колонки имени файла, панель файлов будет отображаться в многоколоночной форме.

- ширина колонок — позволяет изменить ширину колонок панели. Если ширина равна 0, то используется значение по умолчанию. Если ширина колонки с именем, описанием или владельцем равна 0, она будет подсчитана автоматически в зависимости от ширины панели.
  Для правильной работы с различной шириной экрана настоятельно рекомендуется, чтобы в каждом режиме просмотра была хотя бы одна колонка с автоматически вычисляемой шириной.
  Для использования 12-часового формата времени надо увеличить на единицу стандартную ширину колонки времени файла или колонки времени и даты файла. После дальнейшего увеличения в этих колонках также будут показаны секунды и миллисекунды.
  Для показа года в 4-символьном формате нужно увеличить ширину колонки даты на 2.
- типы колонок строки статуса и Ширина колонок строки статуса — аналогично «Типам колонок» и «Ширине колонок», но для строки статуса панели;
- полноэкранный режим — показывать панель во весь экран вместо половины экрана;
- выравнивать расширения файлов — показывать расширения файлов выровненными;

- показывать папки заглавными буквами — показывать все имена папок в верхнем регистре вне зависимости от реального регистра;
- показывать файлы строчными буквами — показывать все имена файлов в нижнем регистре вне зависимости от реального регистра;
- показывать имена файлов из заглавных букв строчными буквами — показывать все имена файлов, которые содержат только заглавные буквы, строчными буквами. По умолчанию эта опция включена.

  Все эти параметры влияют только на способ показа файлов, для их обработки РАК всегда использует настоящий регистр;
- использовать регистро-зависимую сортировку — использовать регистро-зависимую сортировку имен файлов.

Настройки интерфейса:

- часы — показывать часы в верхнем правом углу экрана;
- часы приредактировании и просмотре — показывать часы при редактировании и просмотре файлов;
- мышь — использовать мышь;
- показывать линейку клавиш — показывать назначения функциональных клавиш в нижней строке экрана. Эта опция также может переключаться по <Ctrl+B>;
- всегда показывать меню — показывать меню вверху экрана, даже когда оно неактивно;
- сохранение экрана — запуск программы сохранения экрана после заданного в минутах интервала бездействия;
- история в строках ввода диалогов — сохранять историю в строках ввода некоторых диалогов FAR. Список ранее введенных строк может быть вызван с помощью мыши или <Ctrl+↑> и <Ctrl+↓>. Если вы не хотите вести такую историю, например, по соображениям безопасности, выключите эту опцию;
- установить формат командной строки — изменить формат командной строки РАК. Можно использовать следующие переменные:

  √ $p — текущий путь;

  V $п — буква текущего диска;

  V $g — символ >;

  √ $$ — символ $;
- использовать правый АИ как AltGr — установить эту опцию, если имеются проблемы с использованием комбинаций правой АН для ввода символов в Windows 9x, или отключите ее,

если вы предпочитаете использовать правую Alt для быстрого поиска. Эта опция имеет значение только при работе в Windows 9x и игнорируется под Windows NT;

- показывать общий индикатор копирования — показывать общий индикатор во время выполнения операции копирования. Это может потребовать дополнительного времени перед началом копирования для подсчета общего размера файлов.

Поддержка подключаемых модулей. Внешние подключаемые DLL модули (plugins) могут быть использованы для создания новых команд PAK и поддержки дополнительных файловых систем. Например, работа с архивами, PTP-клиент, временная панель и просмотр сети реализованы с помощью эмулирующих файловые системы модулей.

Все подключаемые модули хранятся в отдельных папках, размещенных в папке 'Plugins', которая находится в одной папке с Par.exe. При обнаружении нового модуля FAR сохраняет информацию о нем и впоследствии загружает его только при необходимости, так что неиспользуемые модули не требуют дополнительной памяти.

Модули могут быть вызваны либо из Меню выбора диска, либо из меню Команды внешних модулей, активизируемого с помощью <P11> или соответствующего пункта Меню команд. <P4> в меню Команды внешних модулей позволяет назначить горячие клавиши для пунктов этого меню, что упрощает их последующий вызов с помощью клавиатурных макрокоманд. Это меню доступно из файловых панелей и (только по <P11>) из встроенной программы просмотра и редактора.

При вызове из программы просмотра и редактора будут показаны не все модули, а только те, которые специально созданы для работы в этом режиме.

Можно установить параметры модулей, используя команду Параметры внешних модулей из Меню параметров.

Модули имеют собственные файлы сообщений и помощи. Можно получить список доступной помощи по модулям, нажав <Shift+F2> в основной помощи PAK.

Если активная панель отображает файловую систему, поддерживаемую внешним модулем, то команда «CD» в командной строке может быть использована для смены текущей папки этой файловой системы. В отличие от «CD», команда «CHDIR» всегда воспринимает указанный параметр как имя реальной папки вне зависимости от типа файловой панели.

*Меню файлов* (см. также табл. П4.3)

- просмотр — просмотр файлов, подсчет размеров папок;
- редактирование — редактирование файлов;

- копирование — копирование файлов и папок;
- перенос — переименование или перенос файлов и папок;
- создание папки — создание новой папки;
- удаление — удаление файлов и папок;
- архивировать — добавить выбранные файлы к архиву;
- распаковать — распаковать выбранные файлы из архива;
- атрибуты файлов — изменитьатрибуты и время файла;
- применить команду — применить команду к выбранным файлам;
- описание файлов — добавить описания к выбранным файлам;
- пометить группу — пометить заданную маской группу файлов;
- снять пометку — снять пометку с группы файлов, соответствующей заданной маске;
- инверсия пометки — инвертировать текущую пометку файлов;
- восстановить пометку — восстановить предыдущую пометку после обработки файлов или операции пометки группы.

Установка атрибутов файлов. Эта команда позволяет изменить атрибуты и время как у отдельных файлов* так и у групп файлов и папок. Если вам не нужно обрабатывать файлы во вложенных папках, отмените параметр Обрабатывать вложенные папки. Атрибут Сжатый может быть изменен только на дисках с файловой системой NTFS.

Поддерживаются три различных времени файла;

- время последней модификации;
- время создания файла;
- время последнего доступа.

Для дисков с файловой системой PAT часы, минуты и секунды времени последнего доступа всегда равны нулю.

Если вы не хотите изменять время файла, оставьте соответствующее поле пустым.

Кнопка Текущее позволяет заполнить поля времени файла текущим временем.

Описания файлов. Описания могут быть использованы для того, чтобы связать с файлом текстовую информацию. Описания файлов текущей папки хранятся в этой папке в специальном файле — списке описаний. В нем в начале каждой строки содержится имя описываемого файла и отделенный от него пробелами текст описания.

Описания можно посмотреть в соответствующих режимах просмотра панели файлов. По умолчанию этими режимами являются Описания и Длинные описания.

Команда Описание файлов (<Ctrl+Z>) из Меню файлов предназначена для добавления описаний к выбранным файлам.

Имена списков описаний могут быть изменены в диалоге Описания файлов из меню Меню параметров. В этом диалоге также можно установить режим обновления локальных описаний. Обновление может быть запрещено совсем, разрешено, только если текущий режим просмотра файловой панели показывает описания, или разрешено всегда. По умолчанию FAR устанавливает атрибут «Hidden» на созданные списки описаний, но вы можете это запретить, выключив опцию «Устанавливать атрибут "Hidden" на новые списки описаний» в этом же диалоге. Также здесь вы можете указать позицию для выравнивания новых описаний в списке описаний.

Если это разрешено в конфигурации, PAK обновляет описания файлов при копировании, переносе и удалении файлов. Но если команда обрабатывает и часть файлов во вложенных папках, то для этих файлов описания не обновляются.

***Меню команд*** (см. табл. П4.4—П4.7)

- поиск файла — поиск в дереве папок файлов, удовлетворяющих заданной маске;
- история команд — показать предыдущие команды;
- видеорежим — выбрать количество строк на экране;
- поиск папки — поиск папки в дереве папок;
- история просмотра — показать историю просмотра и редактирования файлов;
- история папок — показать историю смены папок. Элементы истории просмотра и истории смены папок после выбора передвигаются в конец списка. Необходимо использовать <Shift+Enter>, чтобы выбрать элемент без смены его позиции;
- поменять панели — поменять левую и правую панели местами;
- панели вкл/выкл — показать/спрятать обе панели;
- сравнение папок — сравнить содержимое папок;
- меню пользователя — позволяет редактировать главное или местное меню пользователя. Для вставки пункта используется <Ins>, для удаления — <Del>, для редактирования — <P4>;
- ассоциации файлов — показывает список ассоциаций файлов. для вставки новой ассоциации может использоваться <Ins>, для удаления — <Del>, для редактирования — <P4>;
- ссылки на папки — показывает текущие ссылки на папки;
- группы сортировки — позволяет редактировать задаваемые пользователем группы сортировки;
- фильтр панели — позволяет управлять содержимым панели файлов;

• список экранов — показывает список открытых экранов;

• список задач — показывает список активных задач.

*Некоторые команды*

Поиск файла. Эта команда предназначена для поиска одного или нескольких файлов и папок в дереве папок, в соответствии с одной или несколькими разделенными запятыми масками. Также она может быть использована с файловыми системами, поддерживаемыми с помощью внешних модулей (Plugins). Дополнительно может быть указан текст, который должен содержаться в разыскиваемых файлах. В этом случае параметр Учитывать регистр может быть использован для проведения поиска текста с учетом регистра. С помощью кнопки Таблица можно изменить таблицу символов, используемую для поиска текста. Параметр Использовать все таблицы символов заставляет PAK использовать все доступные ему таблицы для поиска текста в файлах с различной кодировкой. Для поиска файлов и в архивах нужно установить опцию Искать в архивах. В то же время она существенно замедляет выполнение операции и не позволяет выполнять поиск во вложенных архивах.

Поиск может выполняться на всех дисках, кроме сменных, во всех папках, начиная с корневой, начиная с текущей папки, только в текущей папке или в отмеченных папках. Область поиска сохраняется в конфигурации.

Во время или после завершения поиска можно использовать клавиши управления курсором для передвижения по списку файлов и кнопки для выполнения требуемых действий.

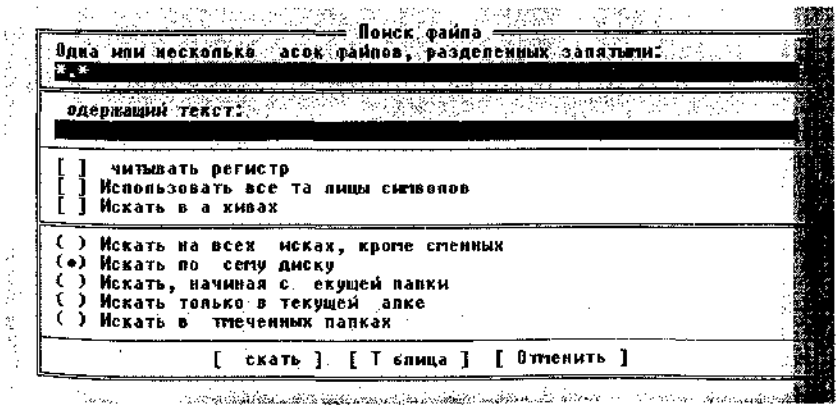



Рис. 4.12. Окно команды «Поиск файла»

Имена списков описаний могут быть изменены в диалоге Описания файлов из меню Меню параметров. В этом диалоге также можно установить режим обновления локальных описаний. Обновление может быть запрещено совсем, разрешено, только если текущий режим просмотра файловой панели показывает описания, или разрешено всегда. По умолчанию FAR устанавливает атрибут «Hidden» на созданные списки описаний, но вы можете это запретить, выключив опцию «Устанавливать атрибут "ШсШеп" на новые списки описаний» в этом же диалоге. Также здесь вы можете указать позицию для выравнивания новых описаний в списке описаний.

Если это разрешено в конфигурации, FAR обновляет описания файлов при копировании, переносе и удалении файлов. Но если команда обрабатывает и часть файлов во вложенных папках, то для этих файлов описания не обновляются.

***Меню команд*** (см. табл. П4.4—П4.7)

- поиск файла — поиск в дереве папок файлов, удовлетворяющих заданной маске;
- история команд — показать предыдущие команды;
- видеорежим — выбрать количество строк на экране;
- поиск папки — поиск папки в дереве папок;
- история просмотра — показать историю просмотра и редактирования файлов;
- история папок — показать историю смены папок. Элементы истории просмотра и истории смены папок после выбора передвигаются в конец списка. Необходимо использовать <Shift+Enter>, чтобы выбрать элемент без смены его позиции;
- поменять панели — поменять левую и правую панели местами;
- панели вкл/выкл — показать/спрятать обе панели;
- сравнение папок — сравнить содержимое папок;
- меню пользователя — позволяет редактировать главное или местное меню пользователя. Для вставки пункта используется <Ins>, для удаления — <Del>, для редактирования — <P4>;
- ассоциации файлов — показывает список ассоциаций файлов. для вставки новой ассоциации может использоваться <Ins>, для удаления — <Del>, для редактирования — <P4>;
- ссылки на папки — показывает текущие ссылки на папки;
- группы сортировки — позволяет редактировать задаваемые пользователем группы сортировки;
- фильтр панели — позволяет управлять содержимым панели файлов;

• список экранов — показывает список открытых экранов;

• список задач — показывает список активных задач.

*Некоторые команды*

Поиск файла. Эта команда предназначена для поиска одного или нескольких файлов и папок в дереве папок, в соответствии с одной или несколькими разделенными запятыми масками. Также она может быть использована с файловыми системами, поддерживаемыми с помощью внешних модулей (Plugins). Дополнительно может быть указан текст, который должен содержаться в разыскиваемых файлах. В этом случае параметр Учитывать регистр может быть использован для проведения поиска текста с учетом регистра. С помощью кнопки Таблица можно изменить таблицу символов, используемую для поиска текста. Параметр Использовать все таблицы символов заставляет PAK использовать все доступные ему таблицы для поиска текста в файлах с различной кодировкой. Для поиска файлов и в архивах нужно установить опцию Искать в архивах. В то же время она существенно замедляет выполнение операции и не позволяет выполнять поиск во вложенных архивах.

Поиск может выполняться на всех дисках, кроме сменных, во всех папках, начиная с корневой, начиная с текущей папки, только в текущей папке или в отмеченных папках. Область поиска сохраняется в конфигурации.

Во время или после завершения поиска можно использовать клавиши управления курсором для передвижения по списку файлов и кнопки для выполнения требуемых действий.

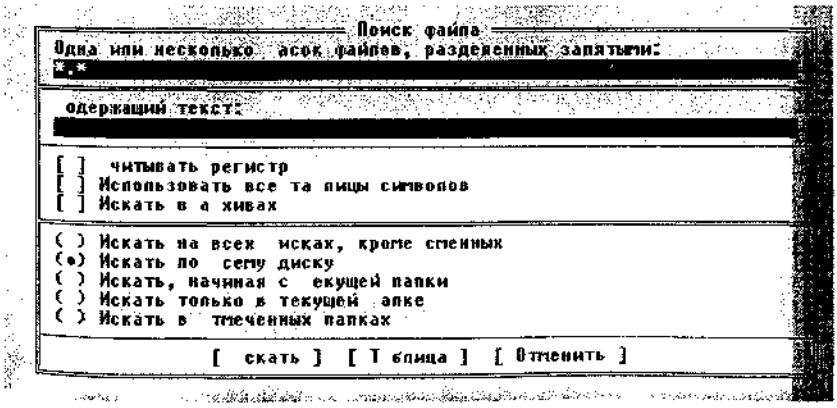



Рис. 4.12. Окно команды «Поиск файла»

Имена списков описаний могут быть изменены в диалоге Описания файлов из меню Меню параметров. В этом диалоге также можно установить режим обновления локальных описаний. Обновление может быть запрещено совсем, разрешено, только если текущий режим просмотра файловой панели показывает описания, или разрешено всегда. По умолчанию FAR устанавливает атрибут «Hidden» на созданные списки описаний, но вы можете это запретить, выключив опцию «Устанавливать атрибут "Hidden" на новые списки описаний» в этом же диалоге. Также здесь вы можете указать позицию для выравнивания новых описаний в списке описаний.

Если это разрешено в конфигурации, PAK обновляет описания файлов при копировании, переносе и удалении файлов. Но если команда обрабатывает и часть файлов во вложенных папках, то для этих файлов описания не обновляются.

***Меню команд*** (см. табл. П4.4—П4.7)

- поиск файла — поиск в дереве папок файлов, удовлетворяющих заданной маске;
- история команд — показать предыдущие команды;
- видеорежим — выбрать количество строк на экране;
- поиск папки — поиск папки в дереве папок;
- история просмотра — показать историю просмотра и редактирования файлов;
- история папок — показать историю смены папок. Элементы истории просмотра и истории смены папок после выбора передвигаются в конец списка. Необходимо использовать <Shift+Enter>, чтобы выбрать элемент без смены его позиции;
- поменять панели — поменять левую и правую панели местами;
- панели вкл/выкл — показать/спрятать обе панели;
- сравнение папок — сравнить содержимое папок;
- меню пользователя — позволяет редактировать главное или местное меню пользователя. Для вставки пункта используется <Ins>, для удаления — <Del>, для редактирования — <P4>;
- ассоциации файлов — показывает список ассоциаций файлов. для вставки новой ассоциации может использоваться <Ins>, для удаления — <Del>, для редактирования — <P4>;
- ссылки на папки — показывает текущие ссылки на папки;
- группы сортировки — позволяет редактировать задаваемые пользователем группы сортировки;
- фильтр панели — позволяет управлять содержимым панели файлов;

- список экранов — показывает список открытых экранов;
- список задач — показывает список активных задач.

*Некоторые команды*

Поиск файла. Эта команда предназначена для поиска одного или нескольких файлов и папок в дереве папок, в соответствии с одной или несколькими разделенными запятыми масками. Также она может быть использована с файловыми системами, поддерживаемыми с помощью внешних модулей (Plugins). Дополнительно может быть указан текст, который должен содержаться в разыскиваемых файлах. В этом случае параметр Учитывать регистр может быть использован для проведения поиска текста с учетом регистра. С помощью кнопки Таблица можно изменить таблицу символов, используемую для поиска текста. Параметр Использовать все таблицы символов заставляет PAK использовать все доступные ему таблицы для поиска текста в файлах с различной кодировкой. Для поиска файлов и в архивах нужно установить опцию Искать в архивах. В то же время она существенно замедляет выполнение операции и не позволяет выполнять поиск во вложенных архивах.

Поиск может выполняться на всех дисках, кроме сменных, во всех папках, начиная с корневой, начиная с текущей папки, только в текущей папке или в отмеченных папках. Область поиска сохраняется в конфигурации.

Во время или после завершения поиска можно использовать клавиши управления курсором для передвижения по списку файлов и кнопки для выполнения требуемых действий.

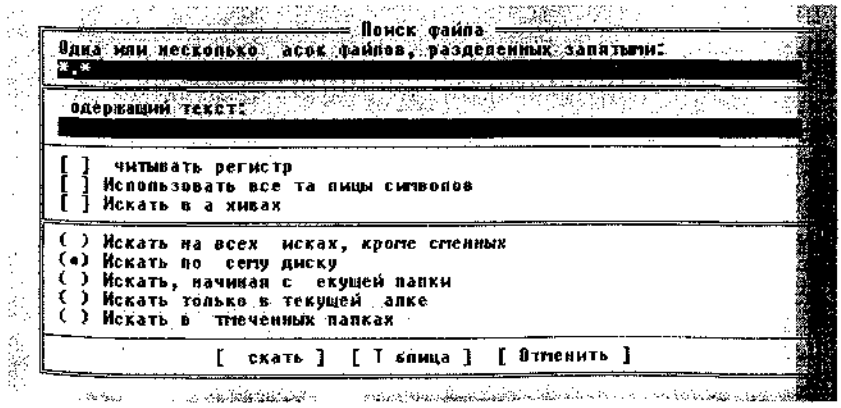



Рис. 4.12. Окно команды «Поиск файла»

Имена списков описаний могут быть изменены в диалоге Описания файлов из меню Меню параметров. В этом диалоге также можно установить режим обновления локальных описаний. Обновление может быть запрещено совсем, разрешено, только если текущий режим просмотра файловой панели показывает описания, или разрешено всегда. По умолчанию РАЯ устанавливает атрибут «ШсШеп» на созданные списки описаний, но вы можете это запретить, выключив опцию «Устанавливать атрибут "ШсШеп" на новые списки описаний» в этом же диалоге. Также здесь вы можете указать позицию для выравнивания новых описаний в списке описаний.

Если это разрешено в конфигурации, РАК обновляет описания файлов при копировании, переносе и удалении файлов. Но если команда обрабатывает и часть файлов во вложенных папках, то для этих файлов описания не обновляются.

***Меню команд*** (см. табл. П4.4—П4.7)

- поиск файла — поиск в дереве папок файлов, удовлетворяющих заданной маске;
- история команд — показать предыдущие команды;
- видеорежим — выбрать количество строк на экране;
- поиск папки — поиск папки в дереве папок;
- история просмотра — показать историю просмотра и редактирования файлов;
- история папок — показать историю смены папок. Элементы истории просмотра и истории смены папок после выбора передвигаются в конец списка. Необходимо использовать <Shift+Enter>, чтобы выбрать элемент без смены его позиции;
- поменять панели — поменять левую и правую панели местами;
- панели вкл/выкл — показать/спрятать обе панели;
- сравнение папок — сравнить содержимое папок;
- меню пользователя — позволяет редактировать главное или местное меню пользователя. Для вставки пункта используется <Ins>, для удаления — <Del>, для редактирования — <P4>;
- ассоциации файлов — показывает список ассоциаций файлов. для вставки новой ассоциации может использоваться <Ins>, для удаления — <Del>, для редактирования — <P4>;
- ссылки на папки — показывает текущие ссылки на папки;
- группы сортировки — позволяет редактировать задаваемые пользователем группы сортировки;
- фильтр панели — позволяет управлять содержимым панели файлов;

- список экранов — показывает список открытых экранов;
- список задач — показывает список активных задач.

*Некоторые команды*

Поиск файла. Эта команда предназначена для поиска одного или нескольких файлов и папок в дереве папок, в соответствии с одной или несколькими разделенными запятыми масками. Также она может быть использована с файловыми системами, поддерживаемыми с помощью внешних модулей (Plugins). Дополнительно может быть указан текст, который должен содержаться в разыскиваемых файлах. В этом случае параметр Учитывать регистр может быть использован для проведения поиска текста с учетом регистра. С помощью кнопки Таблица можно изменить таблицу символов, используемую для поиска текста. Параметр Использовать все таблицы символов заставляет PAЯ использовать все доступные ему таблицы для поиска текста в файлах с различной кодировкой. Для поиска файлов и в архивах нужно установить опцию Искать в архивах. В то же время она существенно замедляет выполнение операции и не позволяет выполнять поиск во вложенных архивах.

Поиск может выполняться на всех дисках, кроме сменных, во всех папках, начиная с корнёвой, начиная с текущей папки, только в текущей папке или в отмеченных папках. Область поиска сохраняется в конфигурации.

Во время или после завершения поиска можно использовать клавиши управления курсором для передвижения по списку файлов и кнопки для выполнения требуемых действий.

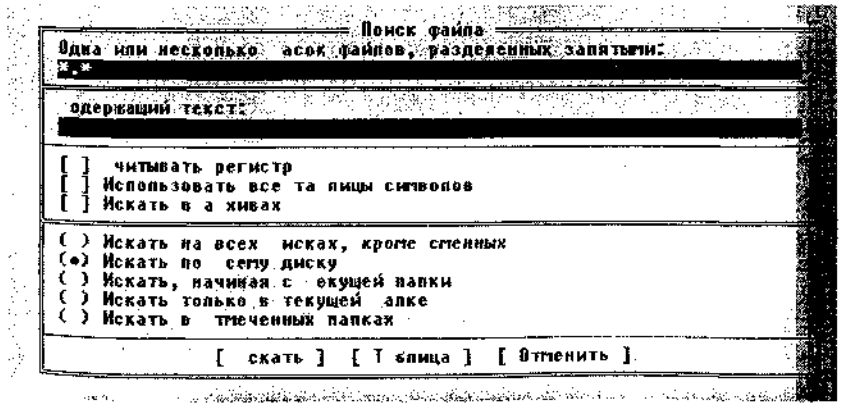



Рис. 4.12. Окно команды «Поиск файла»

Имена списков описаний могут быть изменены в диалоге Описания файлов из меню Меню параметров. В этом диалоге также можно установить режим обновления локальных описаний. Обновление может быть запрещено совсем, разрешено, только если текущий режим просмотра файловой панели показывает описания, или разрешено всегда. По умолчанию РАЯ устанавливает атрибут «Hidden» на созданные списки описаний, но вы можете это запретить, выключив опцию «Устанавливать атрибут "Hidden" на новые списки описаний» в этом же диалоге. Также здесь вы можете указать позицию для выравнивания новых описаний в списке описаний.

Если это разрешено в конфигурации, РАК обновляет описания файлов при копировании, переносе и удалении файлов. Но если команда обрабатывает и часть файлов во вложенных папках, то для этих файлов описания не обновляются.

***Меню команд*** (см. табл. П4.4—П4.7)

- поиск файла — поиск в дереве папок файлов, удовлетворяющих заданной маске;
- история команд — показать предыдущие команды;
- видеорежим — выбрать количество строк на экране;
- поиск папки — поиск папки в дереве папок;
- история просмотра — показать историю просмотра и редактирования файлов;
- история папок — показать историю смены папок. Элементы истории просмотра и истории смены папок после выбора передвигаются в конец списка. Необходимо использовать <Shift+Enter>, чтобы выбрать элемент без смены его позиции;
- поменять панели — поменять левую и правую панели местами;
- панели вкл/выкл — показать/спрятать обе панели;
- сравнение папок — сравнить содержимое папок;
- меню пользователя — позволяет редактировать главное или местное меню пользователя. Для вставки пункта используется <Ins>, для удаления — <Del>, для редактирования — <P4>;
- ассоциации файлов — показывает список ассоциаций файлов. для вставки новой ассоциации может использоваться <Ins>, для удаления — <Del>, для редактирования — <P4>;
- ссылки на папки — показывает текущие ссылки на папки;
- группы сортировки — позволяет редактировать задаваемые пользователем группы сортировки;
- фильтр панели — позволяет управлять содержимым панели файлов;

- список экранов — показывает список открытых экранов;
- список задач — показывает список активных задач.

*Некоторые команды*

Поиск файла. Эта команда предназначена для поиска одного или нескольких файлов и папок в дереве папок, в соответствии с одной или несколькими разделенными запятыми масками. Также она может быть использована с файловыми системами, поддерживаемыми с помощью внешних модулей (Plugins). Дополнительно может быть указан текст, который должен содержаться в разыскиваемых файлах. В этом случае параметр Учитывать регистр может быть использован для проведения поиска текста с учетом регистра. С помощью кнопки Таблица можно изменить таблицу символов, используемую для поиска текста. Параметр Использовать все таблицы символов заставляет РАЯ использовать все доступные ему таблицы для поиска текста в файлах с различной кодировкой. Для поиска файлов и в архивах нужно установить опцию Искать в архивах. В то же время она существенно замедляет выполнение операции и не позволяет выполнять поиск во вложенных архивах.

Поиск может выполняться на всех дисках, кроме сменных, во всех папках, начиная с корнёвой, начиная с текущей папки, только в текущей папке или в отмеченных папках. Область поиска сохраняется в конфигурации.

Во время или после завершения поиска можно использовать клавиши управления курсором для передвижения по списку файлов и кнопки для выполнения требуемых действий.

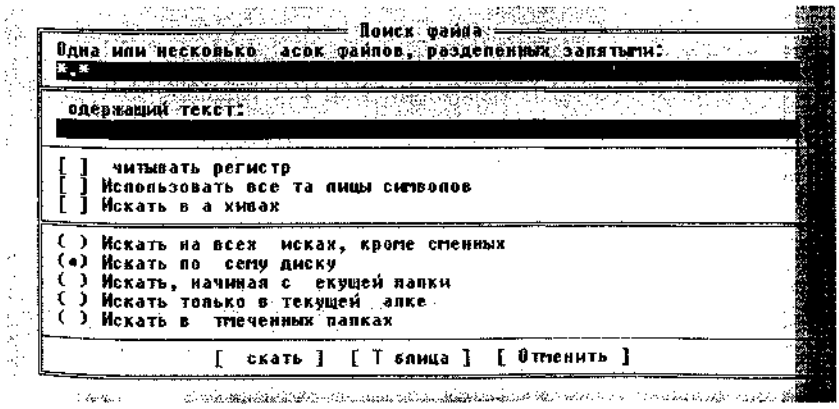



Рис. 4.12. Окно команды «Поиск файла»

Во время или после завершения поиска доступны следующи кнопки:

- новый поиск — начать новую операцию поиска;
- перейти — прервать поиск, сменить текущую папку и поместить курсор на выбранный файл;
- смотреть — просмотр выбранного файла, если поиск не завершен, он будет возобновлен по окончании просмотра;
- панель — создать временную панель и заполнить ее найденными файлами;
- стоп — прервать поиск (доступна во время поиска);
- отмена — закрыть диалог поиска.

Для просмотра и редактирования найденных файлов могут быть использованы <F3> и <F4>, но редактирование и просмотр не поддерживаются для файловых систем подключаемых модулей.

Поиск папки. Эта команда предназначена для быстрого поиска нужной папки в дереве папок. Для выбора папки можно использовать клавиши управления курсором или набрать несколько начальных символов имени папки. Нажать <Enter> для перехода в выбранную папку. <Ctrl+R> и <F2> позволяют перечитать дерево папок.

История команд. История команд показывает список выполненных ранее команд. Необходимо выбрать команду и нажать <Enter>, чтобы выполнить ее еще раз, <Shift+Enter>, чтобы выполнить ее в

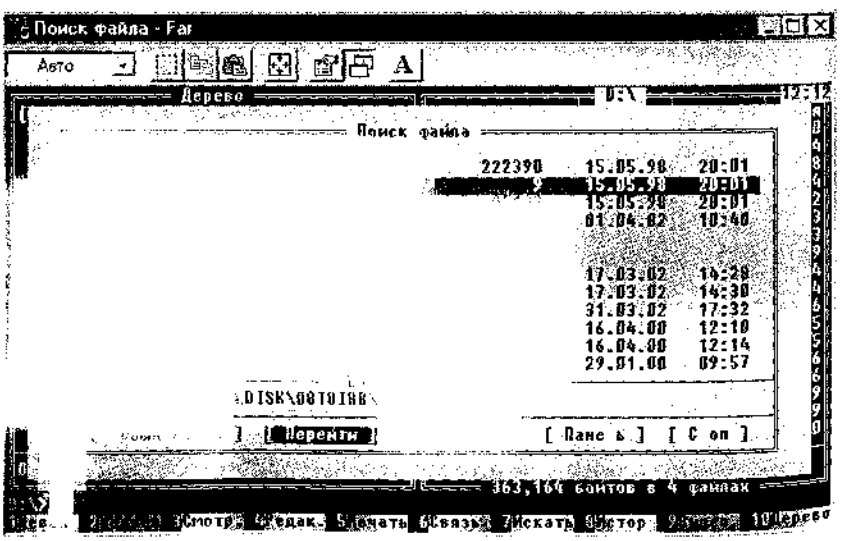



Рис. 4.13. Экран в процессе поиска файлов

отдельном окне или <Ctrl+Enter>, чтобы поместить ее в командную строку для редактирования. Для выбора команды, кроме клавиш управления курсором и <Enter>, можно использовать подсвеченные буквы. <Del> очищает историю команд. Кроме того, для перехода к предыдущей или следующей команде непосредственно из командной строки можно использовать клавиши <Clrl+E> или <Ctrl+X> соответственно.

Для сохранения истории команд перед выходом необходимо использовать соответствующую опцию из диалога системных параметров.

Список задач. Список задач показывает активные на данный момент задачи. Каждая строка списка содержит заголовок окна задачи. Из списка задач можно переключиться на окно задачи или удалить задачу, используя клавишу <Del>. Эта операция выполняется немедленно и любая несохраненная информация данной задачи будет потеряна. Поэтому удаление задач должно использоваться только при необходимости, например, если программа перестала отвечать на запросы пользователя. Список задач может быть вызван либо из Меню команд, либо используя <Ctrl+W>. В последнем случае список задач также может быть вызван из программы просмотра или редактора.

Сравнение папок. Команду сравнения папок можно выполнять, только когда обе панели на экране являются панелями файлов. Она сравнивает содержимое отображаемых в этих панелях папок. Файлы, которые присутствуют только в одной панели, или файлы, чья дата модификации более свежая, чем у файлов с тем же именем в другой панели, становятся помеченными. Вложенные папки не сравниваются. Для сравнения файлов используется их имя, дата и время, но не содержимое.

Меню пользователя. Меню пользователя предназначено для упрощения выполнения часто используемых операций. Оно содержит заданные пользователем команды и последовательности команд, которые могут быть выполнены с использованием этого меню. Меню пользователя может включать вложенные меню. Специальные символы, используемые при Ассоциации файлов, поддерживаются и в командах, и в заголовках команд меню. Составной символ **!?<title>?<init>!** может быть использован для ввода дополнительных параметров непосредственно перед выполнением команды. Для редактирования или создания главного или местного меню пользователя применяется команда Меню пользователя из Меню команд. Главное Меню пользователя может быть только одно. Главное меню вызывается в том случае, если для текущей папки отсутст-

вует местное меню. Местное меню может быть расположено в любой папке.

Для выполнения команды из Меню пользователя нужно выбрать ее с помощью клавиш управления курсором и нажать <Enter>. Также можно использовать назначенную для данного пункта меню горячую клавишу.

Можно удалить вложенное меню или пункт меню, используя клавишу <Del>, вставить новое вложенное меню или пункт меню с помощью <Ins> и редактировать существующее вложенное меню или пункт меню с помощью <P4>, <Alt+F4> для редактирования меню в виде текстового файла. В качестве горячих клавиш для обращения к пунктам меню могут использоваться цифры, буквы и функциональные клавиши (F1—F12). Если использованы <P1> и <P4>, их первоначальные функции теряются. В этом случае для редактирования меню может применяться <Shift+F4>.

При редактировании или создании пункта меню нужно ввести горячую клавишу для быстрого доступа к этому пункту, заголовок пункта, который будет отображаться в меню, и последовательность команд для выполнения в случае выбора данного пункта меню.

При редактировании или создании вложенного меню достаточно ввести горячую клавишу и заголовок вложенного меню. Местные меню хранятся в текстовых файлах *FarMenu.Ini.*

Главное меню по умолчанию хранится в Реестре, но его также можно держать в файле. Если создать местное меню в папке PAK, то оно будет использовано вместо хранящегося в Реестре главного меню.

Ассоциации файлов. PAK позволяет задать три команды, ассоциированные с определенным типом файла, заданным маской:

- команда запуска — выполняется при нажатии <Enter>;
- команда просмотра — выполняется при нажатии <P3>;
- команда редактирования — выполняется при нажатии <P4>.

Ассоциацию можно описать в поле Описание ассоциации.

Новые ассоциации добавляются с помощью команды Ассоциации файлов в Меню команд.

В ассоциированных командах могут использоваться специальные символы:

| | |
|---|---|
| !! | символ «!» |
| ! | Длинное имя файла без расширения |
| !~ | Короткое имя файла без расширения |

*Продолжение таблицы специальных символов*

| | |
|---|---|
| !.! | Длинное имя файла с расширением |
| !-! | Короткое имя файла с расширением |
| !+! | Аналогично !-!, но если длинное имя файла утеряно после выполнения команды, РАЯ восстановит его |
| !@@! | Имя файла, содержащего имена помеченных файлов |
| !$! | Имя файла, содержащего короткие имена помеченных файлов |
| !: | Текущий диск |
| !\ | Текущий путь |
| !/ | Короткое имя текущего пути |
| !?<title>?<init>! | При выполнении команды этот символ заменяется данными, введенными пользователем. <title> и <init> - заголовок и исходный текст строки редактирования |

Допускается использование нескольких таких символов в одной строке, например:

grep !?Search for:?! !?In:?*.*!|c:\far\far.exe -V -

Префикс «!<<», указанный перед символом ассоциации файла, заставляет его ссылаться на пассивную панель. Например, !<<!.! обозначает имя текущего файла на пассивной панели.

***Примечания:***

*1. Если для данного файла отсутствуют ассоциированные команды запуска и установлен параметр Использовать стандартные типы в Системных параметрах, то РАК пытается использовать ассоциации Windowsдля запуска этого типа файлов.*

*2. Вы можно задать несколько ассоциаций для одного типа файлов и выбирать желаемую ассоциацию из меню.*

Используя пункт Применить команду из Меню файлов, можно применить команду к каждому помеченному файлу. Для обозначения имени файла должны использоваться те же символы, что и в Ассоциацияхфайлов.

Например, 'type !.!' будет выводить на экран все помеченные файлы по очереди.

Ссылки на папки. Ссылки на папки позволяют обеспечить быстрый доступ к часто используемым папкам. Для создания ссылки на текущую папку нужно нажать <Ctrl+Shift+N>>, где N - '0'..'9'. После этого, чтобы перейти в папку, записанную в ссылке, достаточно Нажать <ПравыйCtrl+N>. Если <ПравыйCtrl+N> нажат в строке Редактирования, то путь ссылки будет вставлен в эту строку.

Пункт Ссылки на папки в Меню команд позволяет просматривать, устанавливать, редактировать и удалять ссылки на папки.

Группы сортировки. Группы сортировки могут применяться в панели файлов совместно с сортировкой *по имени или по расширению.* Они активизируются при нажатии <Shift+F11> и позволяют задать правила сортировки файлов, дополняющие уже действующие.

Каждая группа сортировки состоит из одной или нескольких разделенных запятыми масок файлов. Если позиция одной группы сортировки в общем списке групп выше, чем у другой группы, то при сортировке по возрастанию все принадлежащие к этой группе файлы будут выше, чем принадлежащие к другой группе.

Команда Группы сортировки из Меню команд позволяет удалять, создавать и редактировать группы сортировки, используя клавиши <Del>, <Ins> и <P4>. Группы, находящиеся выше разделителя меню, относятся к началу файловой панели, и все файлы, попавшие в эти группы, будут расположены выше не попавших в них файлов. Группы, находящиеся ниже разделителя меню, относятся к концу файловой панели, и все файлы, попавшие в эти группы, будут расположены ниже не попавших в них файлов.

Фильтр файловой панели. С помощью фильтра можно определить набор типов файлов, которые будут показываться в панели файлов. Меню фильтра состоит из двух частей. В верхней части расположены пользовательские фильтры. С помощью клавиш <Ins>, <Del> и <P4> можно добавлять, удалять и редактировать их. Каждый пользовательский фильтр включает необязательный заголовок и маску файлов, либо несколько масок файлов, разделенных запятыми. В нижней части меню фильтра находятся маски всех файлов, содержащихся в данный момент в активной панели файлов.

Для выбора элементов меню фильтров могут использоваться клавиши <5pace>, <+> и <->. Элементы, выбранные с помощью пробела или <+>, помечаются символом <+>. Если такие элементы присутствуют, то будут показаны только удовлетворяющие им файлы. Элементы, выбранные с помощью <->, помечаются символом <->, и все удовлетворяющие им файлы будут исключены из панели файлов.

Пометка пользовательских фильтров сохраняется в конфигурации.

Когда фильтр используется в панели, это показывается символом <*> после буквы режима сортировки в верхнем левом углу панели.

***Прочие инструментальные возможности.*** Переключение между экранами. PAK позволяет открыть несколько копий встроенной программы просмотра и редактора. Следует использовать

<Ctrl+Tab>, <Ctrl+Shift+Tab> или <P12> для переключения между панелями и экранами с этими копиями. <Ctrl+Tab> переключает на следующий экран, <Ctrl+Shift+Tab> на предыдущий, <P12> выводит список всех доступных экранов.

Количество фоновых экранов редактирования и просмотра отображается в верхнем левом углу левой панели. Можно запретить показ количества экранов, используя диалог Настройки панели.

Клавиатурные макрокоманды. Клавиатурные макрокоманды могут быть использованы для переопределения стандартных клавиш или комбинаций клавиш PAK или для создания новых клавиатурных команд. Для задания макрокоманды необходимо нажать <Ctrl+.> (Ctrl и клавишу с точкой), желаемую последовательность клавиш, опять <Ctrl+.> и клавишу, либо комбинацию клавиш, на которую будет назначена эта макрокоманда. Во время записи макрокоманды в верхнем левом углу экрана выводится символ 'R'.

Чтобы удалить макрокоманду и вернуть клавише ее первоначальную функцию, следует дважды нажать <Ctrl+.> и затем клавишу, на которую назначена макрокоманда.

Дополнительно к стандартным комбинациям клавиш PAK можно назначать макрокоманды на <Ctrl+Shift+буква>, <Ctrl+Alt+буква> и <Alt+Shift+буква>.

PAK поддерживает несколько независимых наборов макрокоманд: макрокоманды оболочки, программы просмотра, редактора, а также некоторые другие типы макрокоманд. Наборы макрокоманд сохраняются по команде Сохранить параметры из Меню параметров.

Для задания дополнительных параметров макрокоманды необходимо начать или завершить ее запись с помощью <Ctrl+Shift+.> вместо <Ctrl+.> и выбрать желаемые опции в появившемся диалоге.

***Встроенная программа просмотра.*** Возможности программы просмотра достаточно очевидны из списка команд, приведенных в табл. 4.2.

*Таблица 4.2.* **Основные команды программы просмотра**

| Клавиши | Функция | Клавиши | Функция |
|---|---|---|---|
| <← > | Символ влево | <F7> | Поиск |
| <→> | Символ вправо | <Shift+F7>, <5pace> | Искать дальше |
| <T> | Строку вверх | <P8> | Переключить режим просмотра текста DOS/Windows |

*Продолжение табл. 4.*

| Клавиши | Функция | Клавиши | Функция |
|---|---|---|---|
| <↓> | Строку вниз | <Shift-F8> | Выбор пользовательской таблицы символов |
| <Ctrl+←> | 20 символов влево | <Alt-F8> | Изменить текущую позицию |
| <Ctrl+→> | 20 символов вправо | <Num5>,<Р3>, <Р10>, <Е5С> | Выход |
| <PgUp> | Страницу вверх | <Р11> | Вызвать меню «Команды внешних модулей» |
| <PgDn> | Страницу вниз | <+> | Перейти к следующему файлу |
| <Home> | В начало файла | <-> | Перейти к предыдущему файлу |
| <End> | В конец файла | <Ctrl+O> | Показать пользовательский экран |
| <F1> | Помощь | <Alt+BS>, <Ctrl+Z> | Возврат к предыдущей позиции |
| <F2> | Переключить свертку строк | <ПравыйCtrl+N> | Сохранить текущую позицию (N должно быть равно «0»..«9») |
| <Р4> | Переключение между текстовым и шестнадцатеричным режимом | <ЛевыйCtrl+N> | Восстановить сохраненную позицию |
| <Р6> | Переключиться в редактор | | |

***Примечания:***

*1. Пользовательские таблицы символов находятся в папке «Addons\Tables» папки РАК в виде.reg файлов. Перед использованием любой из таблиц ее нужно установить: для этого нажать <Shift+Enter>на соответствующем файле.*

*2. Для вызова диалога поиска достаточно просто начать вводить предназначенный для поиска текст.*

*Настройки программы просмотра.* В этом диалоге можно изменить параметры программы просмотра.

*А. Внешняя программа просмотра:*

- запускать внешнюю программу просмотра по <Р3>;
- запускать внешнюю программу просмотра по <Alt+F3>;
- команда просмотра — команда для запуска внешней программы просмотра, для указания имени просматриваемого файл; использовать специальные символы, описанные в ассоциациях файлов.

Если внешняя программа просмотра назначена на клавиш; <Р3>, она будет запускаться только в том случае, если ассоциированная программа просмотра для данного типа файлов отсутствует.

*Б. Встроенная программа просмотра:*

- сохранять позицию файла — сохранять и восстанавливать позицию в недавно просмотренных файлах. Эта опция также вы-

зывает сохранение таблицы символов, использованной при просмотре файла, в случае если эта таблица была установлена пользователем вручную;

- автоопределение таблицы символов — если приходится пользоваться несколькими таблицами символов и установлена таблица с распределением частот символов для выбранного языка, можно включить эту опцию для автоопределения таблицы просматриваемого файла. Очевидно, что корректное определение не гарантируется, особенно для маленьких или нетипичных текстовых файлов. Некоторые таблицы символов помещены в папке «Addons\Tables» дистрибутива PAK;
- размер табуляции — количество пробелов при показе символа табуляции.

***Встроенный редактор.*** Команды управления встроенным редактором приведены в табл. 4.3 и 4.4.

*Таблица 4.3.* **Команды управления курсором и удаления**

| Команды управления курсором | | Удаление | |
|---|---|---|---|
| <←> | Символ влево | <Del> | Удалить символ (также может удалить блок в зависимости от настроек редактора) |
| <→> | Символ вправо | <B5> | Удалить символ слева |
| <T> | Строку вверх | <Ctrl+Y> | Удалить строку |
| <!> | Строку вниз | <Ctrl+K>, <Alt+D> | Удалить до конца строки |
| <Ctrl+←> | Слово влево | <Ctrl+BS> | Удалить слово слева |
| <Ctrl+→> | Слово вправо | <Ctrl+T>, <Ctrl+Del> | Удалить слово справа |
| <Ctrl+↑> | Прокрутка экрана вверх | | |
| <Ctrl+↓> | Прокрутка экрана вниз | | |
| <PgUp> | Страницу вверх | | |
| <PgDn> | Страницу вниз | | |
| <Home> | В начало строки | | |
| <End> | В конец строки | | |
| <Ctrl+Home> | В начало файла | | |
| <Ctrl+End> | В конец файла | | |
| <Ctrl+N> | В начало экрана | | |
| <Ctrl+E> | В конец экрана | | |

<Shift+F2>. Для переключения между полноэкранным режимом отображения помощи и выводом текста в окне можно использовать <F5>.

## 4,5. Программная оболочка Dosshell

Программа Dosshell представляет собой текстовую оболочку, предназначенную для работы с DOS. В оболочке Dosshell можно запускать прикладные программы на исполнение, осуществлять операции с файлами и дисками. Возможности Dosshell шире, чем PCTools, а ряд операций выполняет только Dosshell. При копировании файлов используется технология Drag and Drop («переместить и отпустить») [17].

Оболочка представляет собой программный пакет, и для полного использования ее возможностей все составляющие пакета должны присутствовать в том каталоге, из которого запускается исполняющая программа оболочки. В состав пакета входят следующие файлы:

DQSSHELLCOM, DOSSHELL.EXE, DOSSHELLGRB, DOSSHELL.HLP, DOSSHELL.INI, DOSSHELLSWP, DOSSHELL.VID, DOSSWAP.EXE

***Запуск оболочки.*** Запуск оболочки MS-DOS Shell, включенной в операционную систему MS-DOS, в простейшем случае осуществляется командой

```
[dk:nk*\]DOSSHELL,
```

которую для автоматизации вызова можно включить в стартовый файл AUTOEXEC.BAT.

Предусмотрена возможность запуска оболочки MS-DOS Shell с одновременной передачей ей параметров настройки режимов. Для запуска оболочки в текстовом режиме используется следующая команда:

```
[dk:nk\]DOSSHELL [/7[:разрешение[n]]] [/B]
```

Для запуска оболочки в графическом режиме используется следующая команда:

```
[dK:nK\]DOSSHELL [/C[:разрешение[n]]] [/B]
```

*Параметры*

*дк:пк* — диск и путь к файлу DOSSHELL.COM или DOSSHELL.EXE, если этот путь не является текущим и не описан в команде PATH,

*разрешение* — определяет группу разрешения экрана. Допустимы следующие значения:

L — низкое разрешение;

M — среднее разрешение;

H — высокое разрешение.

Значение по умолчанию определяется аппаратными возможностями компьютера.

*n* — подпараметр, определяющий разрешение экрана внутри группы. Например, разрешение 43 строк на экран описывается обозначением H1; разрешение 50 строк на экран описывается обозначением H2.

*Ключи*

/T — MS-DOS Shell запускается в текстовом режиме;

/O — MS-DOS Shell запускается в графическом режиме;

/B — MS-DOS Shell запускается в режиме вывода черно-белого изображения.

***Замечания***

*1. Для работы оболочки MS-DOS Shell компьютер должен иметь по крайней мере 384 Кбайт свободной обычной памяти.*

*2. Не запускайте Microsoft Windows из оболочки MS-DOS Shell. Если требуется использовать оба программных продукта, запустите Windows, а затем из среды Windows запустите MS-DOS Shell.*

*3. Режим работы оболочки MS-DOS Shell может быть изменен после ее активизации с помощью меню.*

*4. Текущие установки оболочки MS-DOS Shell хранятся в файле DOSSHELL.INI, создаваемом автоматически. Каждое изменение установок отображается в этом файле, который по умолчанию заводится в том же каталоге, в котором расположены программы оболочки. С помощью переменной окружения DOSSHELL можно задать диск и каталог, в котором будет храниться и обновляться файл DOSSHELL.INI (если перед сеансом работы с оболочкой MS-DOS Shell файл DOSSHELL.INI перенести в этот каталог).*

*5. При запуске прикладных или системных программ из оболочки MS-DOS Shell временные файлы заводятся в том же каталоге, в котором хранится файл DOSSHELL.EXE. С помощью переменной окружения TEMP можно задать местоположение временных файлов.*

***Исходный кадр оболочки MS-DOS Shell.*** После вызова оболочки MS-DOS Shell на экране появляется информационный кадр, показанный на рис. 4.14.

Начальный информационный кадр включает 6 областей (окон):

- строку основного меню;
- линейку дисков;
- изображение дерева каталогов выбранного диска;

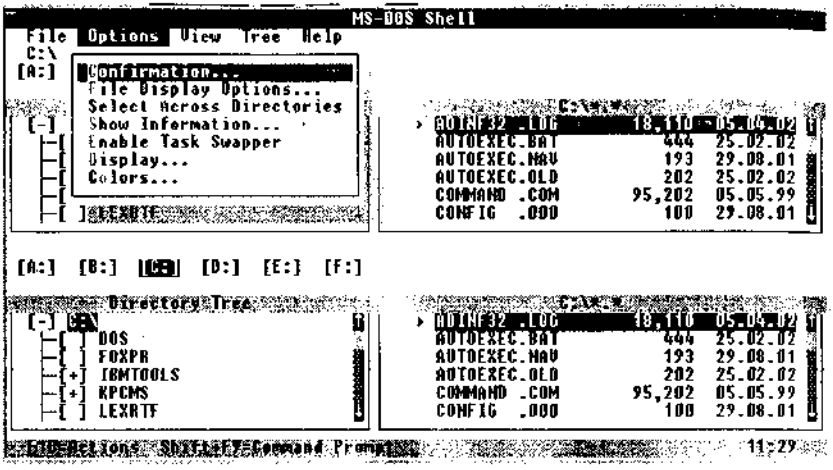



Рис. 4.14. Начальный информационный кадр оболочки MS-DOS Shell

- список файлов выбранного каталога;
- список программ (программных групп и программных элементов);
- список активных задач.

В любой момент одно из окон является активным и выделяется подсветкой заголовка. Смена активного окна осуществляется последовательным нажатием клавиши <TaЬ> или щелчком левой клавиши мыши на заголовке требуемого окна. После загрузки оболочки активной становится линейка имен дисков.

Линейка меню в верхней части экрана File Options View Tree Help служит для задания режимов работы и активизации опций оболочки. Активизация требуемого пункта меню осуществляется следующим образом:

- активизацией всей линейки меню клавишами <Alt> или <P10>, переводом курсора с помощью клавиш <←>, <→> на требуемый пункт и нажатием клавиши <Enter>;
- активизацией всей линейки меню клавишами <Alt> или <P10> и нажатием клавиши с выделенной буквой требуемого пункта меню (P для File, O для Options, V для View, T для Tree и H для Help);
- щелчком левой клавиши мыши на требуемом пункте меню.

Линейка имен дисков [A:] [B:] [C:] [D:] позволяет сменить диск, информация из которого выводится на экран. Смена диска осуществ-

ляется переводом курсора с помощью клавиш <←>, <→> на обозначение требуемого диска и нажатием клавиши <Enter> (при активной линейке дисков) или щелчком левой клавиши мыши на обозначении требуемого диска (при любом активном окне).

Левая верхняя четверть информационного кадра служит для изображения дерева каталогов выбранного диска, а правая верхняя — для вывода списка файлов выбранного каталога.

Смена выбранного каталога осуществляется клавишами <T>, <↓> или щелчком левой клавиши мыши на требуемом каталоге, вывод на экран ветви, нижележащей относительно выбранного каталога — клавишей <+>, или щелчком левой клавиши мыши на знаке «+» развертываемого каталога. Свертывание ветви каталога осуществляется нажатием клавиши <->, когда выбранным является свертываемый каталог, или щелчком левой клавиши мыши на знаке «-» свертываемого каталога (при любом выбранном каталоге).

***Операции с файлами.*** Любая операция с файлом требует, чтобы этот файл был выделен. Групповые операции требуют выделения группы файлов. Выделение файла осуществляется активизацией окна со списком файлов с помощью клавиши <TaЬ> и переводом курсора с помощью клавиш <↑ >, <↓> на выбираемый файл. Щелчок левой клавиши мыши* на требуемом файле активизирует окно файлов и выделяет файл.

Для выделения всех файлов выбранного каталога можно воспользоваться пунктом меню File-Select All или нажать <Ctrl+/>.

Для выделения группы последовательных файлов следует при нажатой клавише <Shift> нажимать клавиши <T> или <↓>. Каждое нажатие будет добавлять очередной файл к группе выделенных. Для выделения группы последовательных файлов с помощью мыши следует щелкнуть левой клавишей мыши на первом файле и при нажатой клавише <Shift> щелкнуть левой клавишей мыши на последнем из выделяемых файлов.

Для выделения группы произвольных (не последовательных в списке) файлов с помощью клавиатуры следует, выделив первый файл, перейти в режим добавления нажатием <Shift+F8> (повторное нажатие <Shift+F8> выключает этот режим). Далее устанавливая курсор с помощью клавиш <T>, <↓> на требуемые файлы, включать их в группу выделенных клавишей пробела (повторное нажатие клавиши пробела отменяет выделение данного файла).

Для выделения группы произвольных файлов с помощью мыши следует при нажатой клавише <Ctrl> щелкать левой клавишей мыши на именах требуемых файлов (отмена выделения выполняет-

ся повторным щелчком левой клавиши мыши на имени файла при нажатой клавише <Ctrl>).

Для того чтобы создать группу выделенных файлов из разных каталогов, следует включить экранную клавишу Options-Select Acro55 Directories. При повторной активизации меню Options в начале строки Select Across Directories будет стоять метка, свидетельствующая о включенном состоянии режима.

Оболочка MS-DOS Shell предоставляет средства поиска файлов по имени или по шаблону групповой операции как в пределах выбранного каталога, так и на всем выбранном диске. Поиск файлов активизируется с помощью пункта меню File-Search. На экран выводится диалоговая рамка, в которой можно указать имя искомого файла или шаблон групповой операции (по умолчанию действует шаблон *.*), а также включить или выключить поиск по всему диску экранной клавишей Search entire disk.

Копирование выделенных файлов осуществляется с помощью пункта меню File-Copy. На экран выводится диалоговая рамка, в которой следует указать спецификацию файла-приемника.

Перемещение выделенных файлов (т. е. копирование с одновременным удалением файлов-источников) осуществляется с помощью пункта меню File-Move.

Удаление выделенных файлов осуществляется с помощью пункта меню File-Delete или нажатием клавиши <Del>. Если включена экранная клавиша Option-Confirmation-Confirm оп Delete, на экран выводится предупреждающее сообщение с требованием подтверждения операции. Если экранная клавиша Option-Confirmation-Confirm оп Delete выключена, предупреждающее сообщение не выводится.

Пункт меню File-Rename позволяет переименовывать выделенные файлы (рис. 4.15). Если выделена группа файлов, диалоговая рамка переименования, куда следует вписать новое имя файла, выводится отдельно для каждого переименовываемого файла.

Содержимое любого (не только текстового) файла можно просмотреть на экране с помощью пункта меню View-File Contents или посредством нажатия клавиши <P9>. Текстовые файлы выводятся в текстовом режиме, двоичные файлы — в шестнадцатеричном, однако и те, и другие файлы можно наблюдать в обоих режимах. Переключение режимов осуществляется той же клавишей <P9>. Прокрутка содержимого длинных файлов выполняется клавишами <T>, <↓>, а постраничная прокрутка — клавишами <PgUp>, <PgDn>. Для выхода из режима просмотра достаточно нажать клавишу <E$c>.

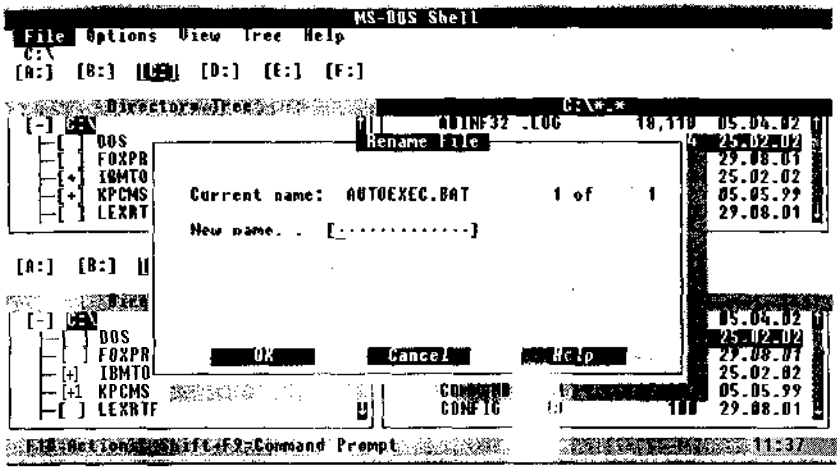



Рис. 4.15. Переименование файла

Оболочка MS-DOS Shell позволяет изменять атрибуты R (только для чтения), 8 (системный), H (скрытый) и A (архивный) выделенных файлов.

Если до запуска оболочки MS-DOS Shell загружена резидентная программа MS-DOS PRINT, из оболочки можно выполнять печать файлов на принтере. При этом должно быть активным окно со списком файлов. В нем следует выбрать файлы, подлежащие выводу на печать, после чего активизировать пункт меню File-Print.

Любое расширение имени файла может быть связано с некоторой программой, так чтобы открытие конкретного файла с таким расширением автоматически запускало связанную программу для обработки именно этого файла. Например, файлы *с* расширениями TXT или DOC можно связать с текстовыми редакторами, а файлы с расширениями ZIP или ARJ — с соответствующими архиваторами. Связь расширений с программами осуществляется с помощью пункта меню File-Associate. После установления связи нажатие клавиши <Enter> или двойной щелчок левой клавиши мыши на имени файла запускает связанную программу для этого файла.

В состав информационного кадра оболочки MS-DOS Shell входит изображение дерева каталогов выбранного диска и списка файлов выбранного каталога этого диска. Формат информационного кадра можно изменять с помощью пункта View основного меню.

Пункт меню View-Dual PHe Lists позволяет вывести на экран одновременно два дерева каталогов со списками файлов. Каталоги могут принадлежать одному или разным дискам.

Пункт меню View-All Files позволяет вывести на экран список всех файлов выбранного диска, включая системные файлы. Работая в таком формате, можно с помощью линейки дисков выполнять смену выбранного диска.

Перейти из режима списка всех файлов в режим изображения каталогов и файлов можно с помощью пункта меню View-Single PHe List (дерево каталогов и список файлов занимают весь экран). В режим списка программ переход осуществляется с помощью пункта View-Program List (список программных групп и программных элементов занимает весь экран), в режим двух каталогов с помощью пункта View-Dual PHe Lists, а в исходный режим всех четырех окон — с помощью пункта View-Program/File Lists.

Поскольку оболочка MS-DOS Shell хранит информацию об информационном кадре в памяти, изменение содержимого диска (например, создание новых файлов) может не найти отражения в информационном кадре. Для обновления информационного кадра после модификации содержимого диска следует воспользоваться пунктом View-Refresh или клавишей <P5>.

***Операции с деревом каталогов.*** Для смены диска, информация из которого выводится в информационный кадр с помощью клавиатуры, можно при нажатой клавише <Ctrl> нажать букву, соответствующую обозначению диска. В режиме двух каталогов смена диска осуществляется в выбранной половине информационного кадра. Другой способ смены диска — с помощью клавиши <TaЬ> выбрать линейку дисков, затем клавишами <←>, <→> выбрать требуемый диск и нажать <Enter>.

Для смены диска с помощью мыши достаточно щелкнуть левой клавишей мыши на обозначении требуемого диска. Двойной щелчок левой клавиши мыши приводит к смене диска с одновременным обновлением информационного кадра.

Для вывода на экран дополнительной информации о выбранных диске, каталоге и файле следует активизировать пункт меню Options-Show Information. Выход из этого кадра осуществляется выбором экранной клавиши Close или нажатием <E$c>. Вывод дополнительной информации возможен только, если активными являются окна с каталогами или файлами (не с программами).

Если выбран некоторый каталог дерева каталогов, в правое окно выводится список файлов выбранного каталога. Смена выбранного

каталога осуществляется клавишами <T>, <I> или щелчком левой клавиши мыши на требуемом каталоге.

Для вывода в информационный кадр ветви, нижележащей относительно выбранного каталога, следует воспользоваться пунктом меню Tree-Expand One Level или нажать клавишу <+>. Развертывание каталога с помощью мыши выполняется щелчком левой клавиши мыши на знаке «+» слева от имени каталога.

Вывод на экран каталогов всех уровней, нижележащих относительно выбранного, осуществляется нажатием клавиши <*> или с помощью пункта меню Tree-Expand Branch. Вывод на экран всего дерева каталогов выбранного диска осуществляется нажатием клавиш <Ctrl+*> или с помощью пункта меню Tree-Expand АИ.

Свертывание нижележащих уровней каталогов относительно выбранного осуществляется нажатием клавиши <-> или с помощью пункта меню Tree-Collapse ВгапсЬ. Свернуть любую ветвь каталогов можно с помощью мыши, щелкнув левой клавишей на знаке «-» слева от свертываемого каталога.

Для создания нового каталога следует выбрать тот каталог, в котором предполагается создать новый, после чего выбрать пункт меню File-Create Directory. На экран выводится диалоговая рамка, в которую следует вписать имя создаваемого каталога.

Удаление выбранного каталога выполняется с помощью пункта меню File-Delete или просто нажатием клавиши <Del>. Если включена экранная клавиша Option-Confirmation-Confirm оп Delete, на экран выводится предупреждающее сообщение с требованием подтверждения операции. Если экранная клавиша Option-Confirmation-Confirm оп Delete выключена, предупреждающее сообщение не выводится. Удалить можно только пустой каталог.

С помощью пункта меню File-Rename можно переименовать выбранный каталог.

*Запуск программ.* Запустить программу можно как из списка файлов, так и из списка программ. Для запуска программы из списка файлов с помощью клавиатуры следует выделить программный файл или связанный с ним файл данных и нажать клавишу <Enter> или выбрать пункт меню File-Open. Для запуска программы из списка файлов с помощью мыши следует дважды щелкнуть левой клавишей мыши на имени программного файла или связанного с ним файла данных.

Для запуска программы из списка программ с помощью клавиатуры следует открыть группу, содержащую требуемый программный элемент, выделить этот элемент и нажать клавишу <Enter> или выбрать пункт меню File-Open. Для запуска программы из списка про-

грамм с помощью мыши следует дважды щелкнуть левой клавишей мыши на требуемом программном элементе.

Для запуска программы с командной строки DOS из главной группы списка программ следует открыть главную группу Main и активизировать программный элемент Command Prompt. Оболочка загружает вторую копию командного процессора и передает ему управление. Пользователь может вводить с клавиатуры любые командные строки DOS, запуская системные или прикладные программы (в том числе и неоднократно). Для возвращения в оболочку MS-DOS Shell следует ввести с клавиатуры команду EXIT.

Для запуска программы, отсутствующей в настоящий момент в информационном кадре, можно воспользоваться пунктом меню File-Run. При активизации этого пункта на экран выводится диалоговая рамка, в которой можно напечатать любую требуемую командную строку DOS.

*Запуск программ в многозадачном режиме.* При активизации механизма свопинга (выгрузки и загрузки выполняемых программ) возникает возможность одновременной работы с несколькими программами. Каждая программа, запускаемая в этом режиме, добавляется в список активных программ. В процессе выполнения любой активной программы можно выйти на время в оболочку MS-DOS 8hell или перейти в любую другую активную программу. Описываемая процедура имеет смысл только для программ, работа с которыми осуществляется длительное время в интерактивном режиме диалога с пользователем. К таким программам относятся, например, текстовые редакторы и процессоры, администраторы баз данных и электронных таблиц, системы программирования с полноэкранным интерфейсом и др.

Для включения механизма свопинга следует выбрать пункт меню Options-Enable Task Swapper. Перед обозначением пункта появляется метка, свидетельствующая об активизации многозадачного режима, а в нижней правой части информационного кадра появляется окно списка активных задач (пока пустое).

Имеется два способа запуска программ в многозадачном режиме: с немедленным переходом из оболочки в запускаемую программу и без выхода из оболочки. Последний режим удобен в тех случаях, когда пользователь хочет сначала запустить целую группу программ и лишь затем начать работать с ними в требуемой последовательности.

Запуск программы с немедленным переходом в нее осуществляется обычным образом, из списка файлов или программных эле-

ментов. На экран выводится информационный кадр запущенной программы, а ее имя заносится оболочкой в список активных задач.

Запуск программы без выхода из оболочки осуществляется нажатием при выделенном имени программы комбинации клавиш <Shift+Enter> или двойным щелчком мыши на имени программы при нажатой клавише <Shift>. Таким образом можно занести в список активных задач несколько программ, оставаясь в оболочке MS-DOS Shell. После этого запуск требуемой программы из списка активных программ осуществляется выбором ее имени и нажатием клавиши <Enter> или двойным щелчком левой клавиши мыши на имени программы.

Возврат из текущей программы в оболочку MS-DOS Shell осуществляется нажатием <Ctrl+Esc>.

Завершение работы с оболочкой MS-DOS Shell возможно только после завершения всех активных программ их внутренними средствами.

*Переключение между программами в многозадачном режиме.* Для переключения из активной задачи в оболочку MS-DOS Shell следует нажать <Ctrl+Esc>. Для обратного переключения из оболочки MS-DOS Shell в активную задачу следует выбрать ее в списке активных задач.

Циклическое переключение программ осуществляется клавишей <TaЬ> при нажатой клавише <Alt>. При каждом нажатии <TaЬ> на экране появляется кадр с именем очередной программы. Для ее активизации достаточно отпустить клавишу <Alt>. Циклическое переключение в обратном направлении осуществляется клавишей <TaЬ> при нажатых клавишах <Shift> и <Alt>.

При наличии нескольких активных задач нажатие <Alt+Esc> осуществляет переключение на следующую программу из списка, а нажатие <Shift+Alt+Esc> — на предыдущую (в число активных задач включается и оболочка MS-DOS Shell). При наличии двух активных задач переключение из одной в другую осуществляется нажатием <A11+TaЬ>.

С помощью пункта меню File-New-Add Рrодгат или File-Properties программе можно назначить горячие клавиши запуска из числа сочетаний <Ctrl+Буква> <Alt+Буква> или <Shift+Буква> (см. раздел «Работа с программными группами и программными элементами».

*Использование дисковых утилит.* В состав главной группы списка групп и программ входит группа Disk Utilities (дисковые утилиты). Программные элементы этой группы позволяют вызывать некоторые часто используемые команды и утилиты MS-DOS.

***Использование интерактивного справочника MS-DOS Shell.*** Оболочка MS-DOS Shell содержит подробный интерактивный справочник по всем элементам оболочки и правилам работы с ней. Доступ к справочнику осуществляется с помощью пункта верхнего меню Help. Справочник построен в виде ссылочной базы данных, что дает возможность пользователю быстро найти необходимую ему информацию, а также дополнительные сведения и пояснения.

Пункт верхнего меню Help предназначен для более или менее систематического изучения оболочки. При необходимости быстро получить справку по используемому в настоящий момент средству оболочки следует нажать клавишу <P1>. На экран будет выведен раздел справочника по конкретному средству (так называемый контекстный поиск).

В нижней части информационного кадра справочника располагаются 5 экранных клавиш. Для активизации экранных клавиш следует нажать клавишу <TaЬ>, которая последовательно выделяет заголовки тем, входящих в текст справки (если они есть), и затем экранные клавиши. Если курсор находится на одной из экранных клавищ, переходить по ним можно с помощью клавиш <←>, <→>. Нажатие клавиши <Enter> активизирует выделенную экранную клавишу. Активизация любого заголовка, входящего в текущий информационный кадр, или любой экранной клавиши выполняется также щелчком левой клавиши мыши на соответствующем поле.

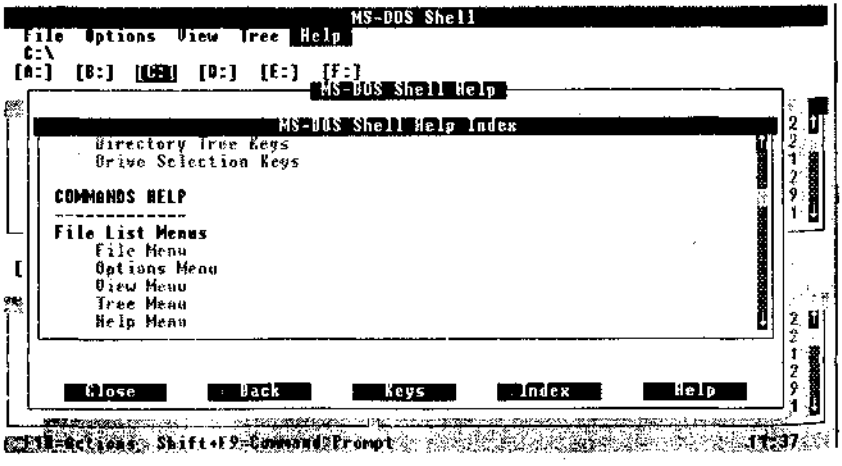



Рис. 4.16. Указатель к содержанию MSDOSShell Help

Назначение клавиш:

Close — закрытие текущего окна справочника и возврат в оболочку MS-DOS Shell. Для того же служит клавиша <Esc>;

Back — возврат в предыдущий информационный кадр справочника;

Index — вывод на экран списка тем интерактивного справочника оболочки MS-DOS Shell;

Keys — вывод на экран раздела справочника о горячих и управляющих клавишах оболочки MS-DOS Shell;

Help — вывод на экран раздела справочника о работе с интерактивным справочником оболочки MS-DOS Shell.

Полный список команд MSDOSShell приведен в Приложении 5.

## Вопросы и упражнения к главе 4

1. **Что такое программные оболочки и какие функции они выполняют?**
2. **Какие вы знаете оболочки для** DOS **и для** Windows? **Что общего между ними?**
3. **Какие оболочки относятся к «нортонообразным», а какие нет?**
4. **Какие функции выполняет** Windows Commander?
5. **Какие функции выполняет** Norton Commander **для** Windows?
6. **Как создать каталог и файл в** Norton Commander?
7. **Укажите назначение основных функциональных клавиш** NC.
8. **Как убрать (восстановить) панели в** Norton Commander?
9. **От чего зависит действие** Norton Commander **при нажатии** <Enter>?
10. **Зачем служит файл** nc.ext **и какова его структура?**
11. **Зачем служит файл** пс.тпи **и какова его структура?**
12. **Какие пункты управляющего меню есть у** Norton Commander?
13. **Какие дисковые функции имеются у** Norton Commander?
14. **Как изменить атрибут файла (группы файлов)?**
15. **Как сделать видимыми (или невидимыми) скрытые и системные файлы?**
16. **Что такое программа** PCTools **и какие функции она выполняет?**
17. **Как запустить программу** PCTools **и как из нее выйти?**
18. **Какие меню имеет программа** PCTools **и чем они отличаются?**
19. **Просмотрите дерево** каталогов **с помощью** pctools **и скопируйте группу файлов в каталог** PRIMER.
20. **Просмотрите план диска и карту дискеты с помощью** pctools.
21. **Что такое программа** Dosshell **и какие функции она выполняет?**
22. **Как запустить программу** Dosshell **и как из нее выйти?**
23. **Чем программы** Dosshell **и** PCTools **отличаются друг от друга?**

## Задания

1. Уберите, а затем восстановите панели NC и поменяйте их местами.
2. Создайте каталог TE5T и в нем создайте текстовый файл GOGA.txt. Затем удалите этот файл и каталог.
3. Произведите сортировку файлов по имени, расширению, дате создания и размеру в каталоге WINDOWS.
4. Найдите на диске файл nc.exe.
5. Найдите на диске каталоги DOS, WINDOWS и 5У5TEM. Выделите в них один файл, группу файлов, все файлы, а затем отмените выделение.
6. Вызовите помощь, затем меню пользователя и управляющее меню NC.
7. Выведите дерево каталогов диска C: и перейдите в каталог NC.
8. Получите всю информацию о диске и дискете. Определите, сколько свободного места осталось на диске и дискете.
9. Отредактируйте файл nc.ext и свяжите файлы с соответствующими приложениями. Проверьте работу нового файла nc.ext.
10. Создайте локальное меню и отредактируйте файл nc.mnu. Запустите программы из этого меню.
11. Отформатируйте дискету, создайте метку на дискете, протестируйте дискету с помощью pctools.
12. Восстановите стертый файл с помощью pctools.
13. Произведите поиск файла win.com на диске с помощью pctools.
14. Скопируйте группу файлов с диска на дискету, просмотрите их, а затем удалите с помощью pctools.
15. Создайте и отредактируйте текстовый файл с помощью программы PCTools.
16. Запустите Par, произведите копирование файлов и каталогов и архивацию файлов.
17. Запустите Windows Commander, произведите копирование файлов и каталогов и распаковку архива.
18. Запустите Norton Commander для Windows и запустите основные приложения Windows.
19. Запустите Par Мападег и осуществите основные операции с файлами и каталогами (копирование, сортировка, переименование, перенос, удаление), запустите прикладные программы на исполнение.
20. Просмотрите дерево каталогов с помощью программы Dosshell и скопируйте группу файлов в каталог TEST_PRIMER.
21. Просмотрите карту диска, карту файлов и карту дискеты с помощью программы Dosshell.
22. Отформатируйте дискету, создайте метку на дискете, протестируйте дискету с помощью программы Dosshell.

23. Запустите на исполнение любую программу с помощью программы Dosshell.
24. Произведите поиск файла win.com на диске с помощью программы Dosshell.
25. Скопируйте группу файлов с диска на дискету, просмотрите их, а затем удалите с помощью программы Dosshell.
26. Создайте и отредактируйте текстовый файл с помощью программы Dosshell.

# Заключение

Операционные системы ЭВМ развиваются и модифицируются в общем контексте развития технических и программных средств. Постоянной средой этого развития является сосуществование по меньшей мере 3 уровней организации информационно-вычислительных процессов — *аппаратурного, программного, информационного.* Они образуют некоторые *слои, страты* информационных технологий, которые взаимозаменяемы в определенных пределах, например функция $y = e^x$ может быть вычислена:

1 — *аппаратурно* (усилитель с нелинейными обратными связями);

2 — *программно* (вычисление суммы ряда для $e^x$);

3 — на основе *базы данных,* в качестве таковой здесь может фигурировать сборник математических таблиц, содержащих соответствующие значения y и x.

В рамках программного обеспечения следует в свою очередь упомянуть известные *подслои — операционные системы, средства разработки приложений, собственно приложения.*

Необходимо отметить то не всегда очевидное обстоятельство, что перечисленные слои технических и программных средств сложились в результате длительной (по масштабам информатики!) эволюции*). Они приспособились друг к другу и взаимодействуют так же, как живые организмы в земной биосфере. Есть растения, которые связывают углекислый газ и выдают кислород, а также производят жиры, крахмал, клетчатку, есть травоядные, которые едят растения, есть хищники, которые едят травоядных, есть человек (венец творения), который ест всех. Но бесперспективны попытки заставить человека усваивать продукт вторичный (см. Вл. Войнович, «Москва, 2042»), и также могут оказаться бесперспективными работы, разрушающие указанную структуру и объединяющие функции

*) *WashingtonPost, 1984 г.:* «В 1953 г. ЭВМ с памятью 64К стоила 1 млн долл., сейчас она стоит менее 1 тыс. долл. Если бы автомобили развивались в течение поледних 20 лет теми же темпами, как компьютеры, то сегодня Роллс-Ройс стоил бы $3.0, проходил бы миллион миль на галлоне бензина, развивал бы мощность лайнера Queen Elisabeth, и 2 автомобиля помещались бы на кончике пера».

различных слоев, хотя они постоянно идут и их результаты даже иногда попадают на рынок (программных средств).

Если не учитывать вероятность «технологических революций» (отказ от фон-неймановских машин, например), то основные направления развития информационных технологий следует ожидать в «диффузии» процессов обработки информации между различными слоями (аппаратурный, программный, информационный) и подслоями программного слоя (операционная система, СУБД и пр.). Самый расхожий пример (ему лет 30) — *макрокоманды* и *микропрограммы.* В первом случае речь идет о программной эмуляции команд процессора (иногда эмуляция системы команд одной машины на другой), во втором — «зашивке программ в железо». Более поздние примеры — эмуляция терминалов, эмуляция протоколов модемов и пр.

Следующий пример (в подслоях) — операционные (ОС) и информационные системы (ИС). Обычные ОС (ОЗ/360, RSX, MS-DOS, Win/95/98/NT) достаточно эффективно обеспечивают *доставку файла* прикладной программе (осуществляют *связь имени с адресом* и не интересуясь содержимым, возлагают эту проблему на пользователя). ИС (СУБД, АИПС) реализует *доставку записи (или строки) файла* в прикладную программу (осуществляет связь *содержания с адресом*).

Тем не менее известны ОС, включающие элементы ИС, например ОС UNIX включает команды **sort** (сортировка текстов), **grep** (контекстный поиск в файле), **awk** (генерация отчетов) и пр. (соединение отношений, например). Достаточно распространенная ОС PICK содержит язык запросов (ЯЗ) ENGLISH, возможны и другие примеры (например, индексные методы доступа в ОЗ/360).

Однако это не более чем внешнее сходство, поскольку ИС обеспечивают выполнение всех этих функций за счет вспомогательных файлов и процессов (индексирование, хэширование, динамически отслеживающих изменение данных), в то время как ОС — за счет прямого сканирования содержания (заведомо неэффективного для больших файлов).

Кроме данного *вертикального* взаимопроникновения компонентов, может быть рассмотрено горизонтальное *(клиент-серверное),* когда функции переработки информации, например, перемещаются от центрального процессора к внешним устройствам ЭВМ:

- интеллектуальный терминал (берет на себя функции предобработки и синтаксического анализа запроса пользователя);
- интеллектуальный принтер (управляется некоторым языком, начиная от ESC-последовательностей, стандарта Epson и за-

канчивая языками управления плоттерами и принтерами Hewlett-Packard);

- интеллектуальный видеомонитор (карты-ускорители алгоритмов обработки изображений, описанных на специализированных языках);
- интеллектуальный контроллер магнитного диска (машина баз данных, data base machine, управляется директивами **find**, **select**, **sort** и передает в центральный процессор только релевантную информацию). Идея, весьма популярная в 80-е годы, но не получившая в последующем распространения, сколько-нибудь заслуживающего внимания.

Данные процессы *диффузии* (как и вертикальные) двунаправлены — достаточно вспомнить алфавитно-цифровые принтеры и терминалы (аппаратная, даже механическая генерация символов) и графические принтеры и терминал (программная генерация). Хороший пример неудачного возложения функций процессора на терминал — протокол устройства IBM 3270 (пользователь продолжает работать, не зная, что хост-машина, например, выключилась). Возврат назад — передача ряда функций терминала (обработка ввода и вывода) центральному процессору — приводит к ANSI-терминалам, наиболее распространенным сегодня.

Рассмотренные двунаправленные горизонтальные и вертикальные перемещения элементов технологий вызывают аналогию с *океанской волной,* которая, устрашающе выглядя на поверхности, переносит потоки энергии, но частицы воды под ней описывают круговые траектории, в принципе оставаясь на месте.

# Литература


1. Бек Л. Введение" в системное программирование. М.: Мир, 1988. 448 с.

2. Дегтярев Е. К., Введение в UNIX. М.: МП «Память», 1992. 128 с.

3. Дьяконов В. Ю., Китов В. А., Калинчев И. А. Системное программирование. Учебное пособие для втузов / Иод ред. А. Л. Горелика. М.: Высшая школа, 1990. 221 с.: ил.

4. Калверт Ч. Программирование в Windows 95. Освой самостоятельно. М.: Восточная Книжная Компания, 1996. 1008 с.

5. Керниган Б. В., Пайк Р. UNIX — универсальная среда программирования. М.: Финансы и статистика, 1992. 304 с.

6. Мэсфилд Р. Windows 95 для занятых, 1998.

7. Петерсен Р. LINUX: руководство по операционной системе. Киев, Издательская группа BHV, 1997. 688 с.

8. Петроченков А. А. Компьютер и периферия, М., 1995.

9. Попов И. И. Автоматизированные информационные системы (по областям применения). Учебное пособие / Под общей редакцией К. И. Курбакова. М.: Изд-во Рос. экон. акад., 1998. 103 с.

10. Попов И. И., Храмцов П. Б. Мировые информационные ресурсы и сети (методы доступа к ним): Учебник / Под ред. К. И. Курбакова. М.: Изд-во Рос. экон. акад., 1998. 145 с.

11. Потапкин А. В. Операционная система Windows 95, 1997.

12. Робачевский А. М. Операционная система UNIX. СПб.: BHV Санкт-Петербург, 1997. 528 с.

13. Стинсон К. Эффективная работа в Windows 95, 1996.

14. Титаренко С. П. Управление процессами в современных операционных системах ЭВМ. Учебное пособие. Изд. Белгородской государственной технологической академии строительных материалов, 1999. 40 с.

15. Фигурнов В. Э. «IBM PC для пользователя», М.: Финансы и статистика, 1990.

16. Фойц С. Windows 3.1, 1995.

17. Франкен Герхард, Молявко Сергей «MS-DOS 6.2 для пользователя», Киев, BHV, 1994, Москва, БИНОМ, 1994.

# Глоссарий (список используемых терминов)

***API (Applicationprogramming interface)*** — интерфейс прикладного программирования — предопределенный набор функций, которые операционная система предоставляет в распоряжение приложений. Определяет состав, параметры и смысл функций, предоставляемых ОС программисту. Например стандарт POSIX на API ОС UNIX включает функции: набор файловых операций, операции со строками, функции многозадачности, управления процессами, управление терминалом. Помимо POSIX есть API Win32 и т. п. API ОС может включать самые разнообразные услуги, вплоть до поддержки функций телефонного аппарата на базе модема (TAPI Win32).

***BIOS (Plug & Play BIOS)*** — базовая система ввода-вывода персонального компьютера. BIOS обеспечивает интерфейс самого низкого уровня с такими устройствами, как системные часы, жесткий диск и монитор. Plug & Play BIOS дополняет функции BIOS рядом процедур, поддерживающих некоторые действия подсистемы Plug & Play, например перечисление устройств.

***Cat (команда UNIX)*** — выводит содержимое файла на экран.

***Cd (команда UNIX)*** — устанавливает указанный каталог текущим рабочим каталогом.

***COM (модель составного объекта)*** — архитектура, послужившая источником для создания *OLE.* Microsoft намеревается сделать COM стандартом отрасли на объектно-ориентированное программирование.

***Cp (команда UNIX)*** — копирование содержимого файла в файл с другим именем либо в другой каталог с сохранением существующего имени файла, всех файлов одного каталога в другой каталог.

***DPMI (DOS-интерфейс защищенного режима)*** — старый способ, благодаря использованию которого могли работать 32-разрядные программы *защищенного режима.*

***Drag-and-Drop*** («захватить-и-перетащить», «перетащить-и-отпустить», «буксировать», «перетаскивать» и пр.) — элемент технологии интерфейсов WIMPD, состоящий из следующих действий — «захват» экранного объекта (ярлык, имя файла и пр.) с помощью указателя

мыши, «буксировка» к месту назначения на экране при нажатой клавише, «сбрасывание» объекта при отпускании клавиши.

*EISA (Extended Industry Standard Architecture)* — стандартная архитектура *шины,* которая позволяет использовать 32-разрядные адаптеры и допускает некоторое автоматическое распознавание и конфигурирование устройств.

*PAT* — таблица размещения файлов.

*Find (команда UNIX)*— поиск файлов. Параметр -пате — поиск файлов с указанным именем. Параметр -print — вывод имен найденных файлов на экран, этот параметр обязателен, если пользователь хочет увидеть результат поиска.

*ISA (Industry Standard Architecture)* — сокращение для стандарта архитектуры системной шины ПК.

*Logname (команда unix)* — вывод системного идентификатора пользователя.

*Ls (команда UNIX)*— при вводе без параметров выдает список файлов и подкаталогов текущего каталога. При вводе с параметром -i — вывод индексов файлов.

*Mkdir (команда UNIX)*— создание нового каталога (каталогов).

*MV (команда UNIX)*— переименование файла или перемещение одного или нескольких файлов в другой каталог.

*MV (команда UNIX)f2.../dd2*— эта команда перемещает указанный файл в указанный каталог.

*OLE (связывание и внедрение объектов)* — реализованная в системах Windows архитектура *Component Object Model (COM).*

*Page (команда UNIX)*— выводит на экран весь файл или его части.

*PCI bus* — разработанная Intel *шина,* которая предназначена для поддержки высокоскоростного 32-разрядного обмена данными между устройствами, памятью и процессором. Подсистема *PLUG & PLAY* полностью поддерживает PCI.

*Pwd (команда UNIX)*— вывод имени текущего каталога (выведет регистрационный каталог, если мы в нем находимся).

*Tty (команда UNIX)*— вывод информации о терминале, соединенном со стандартным вводом.

*WIMPD (Windows, Menu, Pointng Device — окна, меню, указывающее устройство)* — аббревиатура, обозначающая графические интерфейсы (как перечень основных «действующих лиц» в подобном интерфейсе).

***Who (команда UNIX)*** — вывод списка пользователей подключенных в данный момент к системе, даты и времени входа каждого пользователя в систему.

***Автоматическое распределение*** — выделение памяти под данные в стеке. Такие данные существуют на протяжении работы текущей подпрограммы (функции или процедуры), затем уничтожаются.

***Активизация системы*** — реализация процесса (программы).

***Атрибуты файлов*** — каждый файл (каталог) имеет атрибут, который указывает на то, что этот файл является именно файлом, или на то, что он является каталогом. Атрибут файла — только для чтения Read only, скрытый Hidden, системный 5y51eт, архивированный Archive.

***Библиотеки объектных модулей*** — пакет объектных модулей, собранных в один файл и подключаемых к программе на этапе разрешения внешних ссылок (все идентификаторы, которые должны быть доступны из библиотек, объявляются в модулях как PUBLIC). Компоновщик может просматривать библиотеку и самостоятельно находить нужные модули, избавляя от этого программиста. Библиотеки делаются с помощью *программы-библиотекаря*. Библиотекарь может добавлять и извлекать модули, а также получать список доступных идентификаторов. Любой компилятор языка высокого уровня имеет в комплекте несколько стандартных библиотек, например библиотеки ввода-вывода, работы с плавающей точкой, графическую и т. п.

***Винчестер*** — несъемный жесткий магнитный диск (пакет дисков).

***Виртуальная память (ВП)*** отличается от обычной ОП тем, что какие-то ее редко используемые фрагменты могут находиться на диске и подгружаться в реальную ОП по мере необходимости. Такая организация памяти позволяет снять ограничение, накладываемое объемом физической памяти, установленной на ЭВМ. Для реализации ВП используют, например, динамическую переадресацию. Виртуальная память состоит из сегментов. Вначале по номеру в таблице сегментов отыскивается сегмент. Таблица сегментов содержит начальный адрес таблицы страниц. Вторая часть адреса используется для обращения в эту таблицу, и по ней находится физический адрес данной страницы. Результаты поиска по таблицам запоминаются в быстродействующем ассоциативном ЗУ, называемом TLB. Наиболее часто употребляемые адреса откладываются в TLB и поэтому 98—99 % обращений к памяти идут без просмотра таблиц.

***Внешние команды MS-DOS*** — отдельные программы, которые для выполнения загружаются командным процессором MS-DOS в ОЗУ.

***Внешние устройства*** — устройства ввода и вывода информации. Поскольку, как правило, они работают значительно медленнее остальных, управляющее устройство должно приостанавливать программу для завершения операции ввода-вывода с соответствующим устройством.

***Внутренние команды MS-DOS*** — команды, которые выполняются непосредственно командным процессором MS-DOS.

***Время оборота (turnaround time)*** — критерий эффективности планирования, измеряемый интервалом от момента появления процесса во входной очереди до момента его завершения. Это время названо временем оборота и включает время ожидания во входной очереди, время ожидания в очереди готовых процессов, время ожидания в очередях к оборудованию, время выполнения в процессоре и время ввода-вывода.

***Время ожидания (waiting time)*** — критерий эффективности планирования, под которым понимается суммарное время нахождения процесса в очереди готовых процессов.

***Время отклика (response time)*** — критерий эффективности планирования для сугубо интерактивных программ — время, прошедшее от момента попадания процесса во входную очередь до момента первого обращения к терминалу.

***Готовность процесса*** — в распоряжении процесса (программы) имеются все ресурсы, кроме процессора,

***Дескриптор (ОС UNIX)*** — уникальное целое положительное число, которое ставится в соответствие системой файлу при открытии. В процессе работы дескриптор используется процессом или его потомком для указания конкретного объекта операции.

***Дескриптор процесса*** — динамика процесса определяется динамическими характеристиками дескриптора. В нем отображаются динамически изменяемые связи процесса с другими процессами. Все процессы, находящиеся в текущий момент в одном и том же состоянии, объединяют через дескрипторы в одну списковую структуру, но с учетом приоритетов процессов.

***Динамическое выделение*** — выделение памяти под данные самой программой, когда это необходимо. Время жизни таких данных зависит от программы.

***Дисциплина обслуживания ПРО*** (First In First От) — в порядке поступления: первый пришел — первый обслуживается. Все заявки на обслуживание поступают в конец очереди. Первыми обслуживаются заявки, находящиеся в начале очереди.

***Дисциплина обслуживания LIFO*** — обслуживание заявок в порядке, обратном порядку поступления: последний пришел — первый обслуживается (Last In First Out). Является основой для построения стековой памяти.

***Дисциплина распределения ресурса*** — определяет порядок использования многими процессами того или иного ресурса, который в каждый момент времени может обслуживать только один процесс.

***Долгосрочное планирование*** — на данном уровне объектом является не отдельный процесс, а некоторое объединение процессов по функциональному назначению, которое называется ***работой*** (приложением). Каждая работа рассматривается как независимая от других работ деятельность, связанная с использованием одной или многих программ и характеризующаяся конечностью и определенностью. По мере порождения новых работ создается собственная виртуальная машина для их выполнения. Планирование реализует программа ОС *долгосрочный* (планировщик). В OS/360 долговременный планировщик назывался *планировщиком заданий.*

***Дорожка*** — концентрическая окружность на магнитной поверхности диска, где располагается информация. Дорожки нумеруются с 0-й (дорожка с самым большим радиусом).

***Зависимые переключатели*** (радиокнопки, RadioButton) — группа переключателей для выбора одного из нескольких возможных взаимоисключающих режимов работы. Описания режимов находятся справа от кружков. В одной группе может быть включен только один из переключателей, остальные автоматически сбрасываются. Включенный (активный) режим индицируется точкой внутри кружка.

***Задача*** — одна или несколько программ, связанных общим назначением, ресурсами.

***Защита памяти*** — осуществляется путем блокировки доступа к памяти других процессов, а также блокировки доступа к памяти ядра. Один из способов — вся память делится на страницы, и у каждой есть замок — 4-битовый признак, который можно установить только привилегированной командой. В процессоре есть 4-битовый регистр — ключ, который также можно установить только привилегированной командой. При обращении происходит сравнение замка и ключа. С появлением многозадачности появилась проблема распределения памяти. При работе реальной программы обращения к ОП имеют тенденцию к локализации. Память можно разделить на используемую и неиспользуемую. Чтобы отследить использование области памяти, всю ОП можно разбить на страницы фиксированного размера (4K) и с каждой страницей связать бит, который устанавливают при обращении к данной странице.

***Защита программ*** и данных в многозадачных ОС осуществляется с целью, что сбой одной из выполняющихся программ не вызовет повреждения данных или кода других программ, и по возможности изолировать процессы друг от друга. Во всех ОС существуют хотя бы 2 режима процессора — *системный* и *пользовательский.* Переключение режимов работы осуществляется системными вызовами. Системный вызов — специальная команда, приводящая к прерыванию, и в ядре ОС существует несколько точек, куда перейдет управление по этому прерыванию.

***Защищенный режим (protected mode)*** — режим работы процессора Intel 386, при котором он выполняет множество проверок корректности обращений к памяти, вызовов функций, доступа к портам ввода-вывода и т. д. Такая защищенность позволяет операционной системе обрабатывать ошибочные операции. Для того чтобы иметь возможность использовать все адресное пространство и преимущества виртуальной памяти процессора 386, приложение должно работать в *защищенном режиме.*

***Идентификатор группы процессов (ОС UNIX).*** Каждый активный процесс является членом какой-либо группы процессов. В качестве идентификатора группы процессов используется идентификатор процесса, старшего в группе (общего родителя всех процессов группы). Объединение процессов в группу позволяет работать с группой, как с одним объектом, например передавать сигнал всем процессам группы. В некоторых версиях ОС UNIX это свойство используется в языке управления заданиями.

***Идентификатор группы терминала (ОС UNIX).*** Каждый активный процесс является членом группы, управляемой с конкретного терминала. Группа идентифицируется целым положительным числом, называемым «идентификатором группы терминала». Этот вид объединения процессов используется для управления доступом различных процессов к одному и тому же устройству.

***Идентификатор процесса PID (ОС UNIX).*** Каждый активный процесс в системе идентифицируется уникальным целым положительным числом, называемым «идентификатором процесса». Диапазон представления идентификатора процесса — от 0 до ProcMax (значение ProcMax устанавливается при генерации системы).

***Идентификатор процесса-предка PPID (ОС UNIX)*** — идентификатор процесса, породившего данный процесс (посредством вызова fork).

***Имя файла (ОС UNIX)*** — последовательность от одного до максимально допустимого числа символов, использующаяся для именования обычных файлов, каталогов или специальных файлов. В имени файла допустимы любые символы кода ASCII, за исключением

управляющего кода О (ПУС) и символа «/». Не рекомендуется использовать в именах файлов символы, имеющие специальное значение для языков управления заданиями (типа «*», «?»).

***Исполняемый модуль*** — модуль, содержащий готовую к выполнению программу; может быть *2* видов: *точный образ памяти* программы с привязкой к абсолютным адресам (в MS-DOS — формат файла *.COM) и *перемещаемый* исполняемый формат.

***Исходный код программы*** — код, написанный на языке программирования. Может включать модули на ЯВУ и модули с подпрограммами на языке ассемблера.

***Каталог (ОС UNIX)***— специальный тип файла, содержащий информацию о файлах, которые могут адресоваться из данного каталога без указания полного имени (т. е. по имени файла). Любой каталог содержит по крайней мере два имени «.» и «..». Они соответствуют данному каталогу («.») и каталогу, в который данный каталог входит («..»).

***Кластер*** ~ минимальная порция информации, которую MS-DOS считывает/записывает за одно обращение диску. Кластер включает только последовательно расположенные секторы (цель — увеличить скорость обмена с диском). Размер кластера = *N*. Размер сектора = = *N* • 512 байт, где *N*= 2,4,8 и т. д.

***Корневой и текущий (рабочий) каталоги (ОС UNIX)*** — каждому процессу поставлены в соответствие корневой и рабочий каталоги, использующиеся для поиска конкретных файлов по их именам. Корневой каталог процесса может не совпадать с корневым каталогом системы.

***Краткосрочное планирование*** — на данном уровне объектом управления являются ***процессы,*** которые выступают как потребители центрального процессора для внутренних процессов или внешнего процессора для внешних процессов. Планирование осуществляет *краткосрочный* (short term scheduler / CPU scheduler). В О8/360 краткосрочный назывался *супервизором задач.*

***Куча.*** Для реализации динамических структур данных используют т. н. кучу (Ьеар). Это объем памяти, в котором можно выделить участок для произвольного элемента данных. Для кучи есть 2 операции: выделения памяти ALLOCATE и освобождения FREE. Эти функции не делают никаких действий с собственно памятью. При выделении программист получает адрес, а при освобождении доступный объем кучи становится больше. Одного адреса для этих операций недостаточно, требуется еще и размер элемента данных. Если указатель **типизированный,** размер будет получен автматически. В случае **нетипизированного** указателя размер должен быть передан в функцию. Для

реализации кучи ЯВУ снабжаются диспетчерами памяти, которые выделяют и освобождают память, имеют сведения о ее фрагментации, знают наибольший фрагмент свободной памяти и ее общее количество и т. п. При ненадобности память должна своевременно освобождаться. При использовании динамической памяти возможна ситуация образования «мусора» — кусков памяти, на которые утеряны ссылки, но которые не были своевременно освобождены, поэтому менеджер считает их занятыми. Для оптимизации известна процедура *«сбора мусора»* — перестройки динамических структур с освобождением памяти из-под тех данных, на которые отсутствуют ссылки.

***Кэш-память*** — сверхоперативная память, обращение к которой намного быстрее, чем к оперативной, и в которой хранятся наиболее часто используемые участки последней. При обращении к памяти сначала нужные данные ищутся в кэш-памяти. При их отсутствии производится обращение к оперативной памяти, в результате общее время доступа к памяти сокращается.

***Модульное программирование*** — технология программирования на основе разбиения программы на подпрограммы по специфике обрабатываемых данных. Для этой цели в ЯВУ используются функции и Процедуры. При вызове подпрограммы в стеке сохраняется текущее значение счетчика команд (ближняя модель вызова) и значение сегмента кода (дальняя модель вызова). При использовании дальней модели вызова подпрограмма необязательно должна находиться в том же сегменте, что и вызывающая программа. По окончании выполнения кода подпрограммы эти данные восстанавливаются, управление передается на следующий оператор после оператора вызова процедуры. До вызова подпрограммы в стек помещаются параметры — аргументы подпрограммы; Если передан не сам аргумент, а его адрес, то подпрограмма может изменить аргумент, в противном случае нет, т. к. по завершению работы подпрограммы стек очищается от аргументов. Отличие функций от процедур в том, что функции могут возвращать значения в вызывающую программу и их можно присваивать, например какой-либо переменной. В действительности функция при возврате значений просто модифицирует регистры процессора, а ЯВУ по соглашению использует данные из этих регистров.

***Наиболее подходящего стратегия (best fit strategy)*** — выбор процесса из очереди при освобождении раздела памяти. Выбирается процесс, которому в освободившемся разделе наиболее тесно (выигрыш в памяти).

***Наименее подходящего стратегия (last fit strategy)*** — выбор процесса из очереди при освобождении раздела памяти. Выбирается процесс,

которому в освободившемся разделе наиболее свободно (в этом случае остающийся фрагмент часто достаточен для размещения еще одного процесса).

***Объект (object)*** — формально это совокупность данных и методов работы с ними, некоторые из которых могут использоваться другим приложением. Объектно-ориентированные технологии позволяют создателю объекта определить интерфейсы к возможностям объекта, скрыв при этом особенности его реализации. Это делает возможным использование объекта многими непосредственно не относящимися к нему приложениями. Несмотря на то, что этот термин широко используется в Windows 95, в большинстве случаев он применяется в значении «данные» или «нечто». Слово «объект» — это, пожалуй, самый заметный кандидат на звание наиболее перегруженного термина в области программного обеспечения.

***Объектный модуль*** — код программы после трансляции (компиляции), преобразованный в машинные коды. Помимо них содержит внешние ссылки и информацию для редактора связей и может также содержать отладочную информацию (debug info).

***Ожидание процесса*** — ожидание всех ресурсов для развития процесса (программы).

***Окончание процесса*** — высвобождение всех ресурсов процессом.

***Оперативная память*** — предназначена для хранения программ и данных, которыми они манипулируют. Физически выполнена в виде некоторого числа микросхем. Логически ОП можно представить как линейную совокупность ячеек, каждая из которых имеет свой номер, называемый адресом. Время записи и чтения из ОП в современных машинах занимает доли микросекунды, а для других устройств это время в 10—1000 раз больше. Число микросхем памяти, физически присутствующих в компьютере, определяет объем памяти, которую можно использовать для программ и данных. Это число может меняться от машины к машине. Объем памяти обычно можно наращивать с помощью плат расширения, вставляемых в специальные разъемы. Для процессора память — это не более чем несколько тысяч 8-разрядных ячеек, каждая из которых имеет уникальный адрес.

***Операционная оболочка*** — программа, один из модулей которой (резидентный) постоянно находится в памяти компьютера и для выполнения каких-либо заданных пользователем функций загружает с диска в свободные области памяти необходимые исполнительные модули. Операционные оболочки предназначены в основном для упрощения выполнения команд ОС и удобного представления требуемой информации.

***Операционная система*** — обеспечивает следующие функции — управление процессором путем передачи управления программам, обработку прерываний, синхронизацию доступа к ресурсам, управление памятью, управление устройствами ввода-вывода, управление инициализацией программ, межпрограммные связи, управление данными на долговременных носителях путем поддержки файловой системы.

*Организация ввода-вывода* — в современных ЭВМ осуществлена с использованием *прерываний.* Это связано с тем, что устройства ввода-вывода работают намного медленнее, чем процессор и оперативная память. Поэтому управляющее устройство должно приостанавливать выполнение программы и ждать завершения операции ввода-вывода с внешним устройством. При выводе все результаты выполненной программы должны быть выведены на ВУ, после чего процессор переходит к ожиданию сигналов от ВУ. При вводе, например, с клавиатуры получение значений нажатых клавиш осуществляется при поступлении прерывания от клавиатуры.

*Очередь **готовых** процессов **(ready queue)*** — готовые к выполнению процессы, расположенные в основной памяти и ожидающие освобождения ресурса «процессорное время».

*Очередь к оборудованию ввода-вывода **(devices queue)*** — одна из очередей, в которой находится процесс при ожидании завершения операций ввода или вывода.

*Очередь работ / заданий **(job queue)*** — входная очередь для новых процессов.

*Пейджинг* — механизм виртуальной памяти, при котором страницы вытесняются на диск и подкачиваются с диска.

*Первого подходящего **стратегия (first fit strategy)*** — выбор процесса из очереди при освобождении раздела памяти. Выбирается первый процесс, который может разместиться в освободившемся разделе.

*Планировщик* — программа, выполняющая алгоритм планирования процессов. Планирование очередности предоставления выполняющимся процессам времени центрального процессора (диспетчеризация). Процессы работают с центральным процессором в режиме разделения времени.

*Поле ввода* — область, в которую пользователь может вводить информацию с клавиатуры. В этой области указатель мыши принимает новую форму. Если в этот момент щелкнуть кнопкой мыши, то в поле появится курсор и можно вводить данные.

*Поле ввода с раскрывающимся списком* — это комбинация элементов *поля ввода* и *раскрывающегося списка.* Такой элемент позволяет как непосредственно (вручную) вводить данные в *поле ввода,* так и заполнить его значением из раскрывающегося списка.

***Поле ввода со счетчиком*** — обычно используется для ввода числовых значений. Его можно заполнить как обычное поле ввода или воспользоваться кнопочками, расположенными справа. В этом случае значение в поле будет изменяться (соответственно увеличиваться и уменьшаться) с наиболее оптимальным шагом и при этом не превысит предельных значений.

***Полное имя файла (ОС UNIX)*** — последовательность имен каталогов, разделенных символами «/», предшествующая имени файла. Полное имя файла содержит информацию о положении каталога с файлом в дереве файлов. Если полное имя начинается с символа «/» (абсолютная адресация), оно указывает положение каталога с файлом относительно корня дерева файлов. Имя, начинающееся с любого другого символа, указывает положение каталога с файлом относительно текущего каталога. Длина полного имени файла не может превышать (PathnameMax) символов.

***Порождение процесса*** — создание условий для реализации программы.

***Поток*** (минизадача, шаг, цепь, нить) — это последовательности команд процесса, которые выполняются независимо одна от другой и используют общие ресурсы одного процесса.

***Прерывания*** — специфические сигналы, посылаемые процессору устройством или программой, когда требуется его немедленное вмешательство. В этом случае он останавливает всякую другую деятельность и вызывает программу *обработчик прерывания.* По окончании ее работы он продолжает прерванную работу с того места, где она остановилась. Прерывания бывают 2 типов — *аппаратные* (генерируются схемами ПК в ответ на какое-либо действие, например при нажатии клавиши на клавиатуре генерируется прерывание), иногда аппаратные прерывания генерируются устройством в случае некорректной работы программы, например деление на 0; *программные* — генерируются программой для вызова различных подпрограмм из ОЗУ и ПЗУ.

***Привилегированный пользователь и привилегированный процесс (ОС UNIX).*** Процесс считается привилегированным, т. е. получает исключительные права доступа ко всем ресурсам, если его эффективный идентификатор пользователя равен нулю.

***Пропускная способность*** — пропускная способность процессора измеряется количеством заданий, которые выполняются в единицу времени.

***Процесс*** — минимальный программный объект, обладающий собственными системными ресурсами (запущенная программа). Процесс, как любая деятельность по исполнению программы на процессоре, нуждается в управлении, которое заключается в переводе его из од-

ного состояния в другое: порождение — готовность — активизация системы — ожидание — окончание.

*Процесс выполняемый* ***(running)*** — команды программы выполняются процессором.

*Процесс готовый* ***(ready)*** — процесс ожидает освобождения процессора ЭВМ.

*Процесс завершенный* ***(terminated)*** — процесс завершил свою работу.

*Процесс новый* ***(new)*** — процесс только что создан.

*Процесс ожидающий* ***(waiting)***— процесс ожидает завершения некоторого события, чаще всего операции ввода-вывода.

*Процессор (ЦП,* ***CPU)*** — устройство, выполняющее вычислительные операции и управляющее работой ЭВМ. Содержит *устройство управления,* выбирающее машинные команды из памяти и выполняющее их, и *арифметико-логическое устройство,* выполняющее арифметические и логические операции. Работа всех электронных устройств машины координируется сигналами, вырабатываемыми ЦП. В современных ПК процессор представлен одной СБИС, содержащей свыше миллиона транзисторов.

*Работа* — объединение процессов (задач), рассматриваемых как единое целое в операционной системе (проект).

*Раскрывающийся список* (List) — при нажатии на пиктограмму со стрелкой открывается список всех возможных значений, которые можно выбрать для установки в этом элементе. Если список длинный, то появится линейка прокрутки, с помощью которой можно просмотреть все элементы списка.

*Реальные идентификаторы* ***пользователя*** *и группы* ***UID, GID*** *(ОС* ***UNIX).*** Каждый пользователь системы идентифицируется целым положительным числом, называемым «идентификатором пользователя». В то же время пользователь может являться членом одной или нескольких групп. Группы отличаются друг от друга специфическими правами доступа. Положительное целое число, поставленное в соответствие группе, называется «идентификатором группы». Процесс имеет реальные идентификаторы пользователя и группы, значения которых равны соответствующим идентификаторам пользователя, инициировавшего данный процесс. Реальные идентификаторы наследуются всеми потомками процесса.

*Реальный режим (real* ***mode)*** — режим работы процессора Intel 386, совместимый с процессором Intel 8086. В реальном режиме невозможны доступ к огромному *виртуальному адресному пространству 386* процессора или такие возможности, как, например, *замещение страниц по требованию.*

*Регулятор* — используется для установки параметров от минимального до максимального с помощью движка.

*Редактирование связей* — разрешение внешних ссылок и создание исполняемого модуля из совокупности объектных. Основные функции редактора связей — распределение памяти, разрешение внешних ссылок.

*Режим доступа файла (ОС UNIX)* — определяет права процесса на выполнение операций над файлом (например, открытие файла на запись), является принадлежностью файла и устанавливается при его создании, может переопределяться с помощью вызова **chmod.** Режим доступа содержит информацию о возможности чтения, записи и выполнения файла для трех групп пользователей: владельца файла, членов его группы и всех остальных пользователей. Для каталогов признак «выполнения» интерпретируется как право просмотра его содержимого.

*Резидентная программа* — программа, которая после загрузки в ОЗУ и передаче ей управления инициализируется таким образом, что постоянно находится в ОЗУ и выполняется параллельно другим программам.

*Ресурс* ***(resource)*** — сетевой *объект,* такой, как принтер или набор связанных в каталог файлов, доступный для совместного использования.

*Свопинг* — алгоритм реализации виртуальной памяти. Его можно разбить на три части: управление пространством на устройстве выгрузки, выгрузка процессов из основной памяти и подкачка процессов в основную память. В качестве устройства выгрузки используют раздел на устройстве типа жесткого диска или дисковый файл (swap-file) на таком устройстве.

*Сегментная виртуальная память* — использует сегментно-страничное виртуальное адресное пространство. При этом требуется явное выделение частей программы. Реальная память также подвергается сегментной детализации по модульному признаку.

***Сегментно-страничная*** *виртуальная память* — каждый модуль программы подвергается дополнительному страничному структурированию. При этом размеры получаемых сегментов могут быть увеличены. В этом случае оперативная память подвергается также страничной структуризации. Такая схема реализует компромисс между операционной системой и пользователем, а именно: операционная система увеличивает пропускную способность, используя преимущества страничной по запросу схемы функционирования оперативной памяти, а пользователь избавлен от ограничений на размеры разрабатываемой программы.

*Сектор* — каждая дорожка, размещенная на диске, делится на секторы (блоки). Каждый сектор имеет размер 512 байт (для MS-DOS).

*Сервер (server)* — сетевой компьютер, на котором находятся доступные клиентам *ресурсы.* Ресурсами сервера могут быть файлы, принтеры или приложения сервера (такие, как многопользовательские базы данных).

*Сети типа «клиент-сервер» (client-server networking)* — сетевая архитектура, в которой предназначенные для совместного использования *ресурсы (resources)* сосредоточены на мощных компьютерах *серверах (server machines),* а подключенные к ним настольные машины играют роль *клиентов (clients),* посылая по сети запросы на ту или иную информацию.

*Системное программное обеспечение* — обеспечивает интерфейс между программистом или пользователем и аппаратной частью ЭВМ (операционная система, программы-оболочки), выполняет вспомогательные функции (программы-утилиты).

*Системные вызовы (ОС UNIX)* — представляют собой интерфейс между программами пользователя и ядром операционной системы UNIX. Реализация системных вызовов — одна из функций ядра ОС UNIX.

*Системные процессы (ОС UNIX)* — процессы с идентификаторами 0, 1 считаются системными. Это: планировщик (процесс 0), инициализирующий процесс, который одновременно является родителем всех остальных процессов (процесс 1).

*Смежное размещение (contiguous allocation)* — размещение программ в памяти, предполагает, что в памяти, начиная с некоторого начального адреса, выделяется один непрерывный участок адресного пространства. При *несмежном размещении* программа разбивается на множество частей, которые располагаются в различных, необязательно смежных участках адресного пространства.

*Список* — элемент, содержащий все возможные в каждом конкретном случае значения, которые пользователь может установить. Добавить или изменить эти значения непосредственно в списке нельзя.

*Статическое выделение памяти* — выделение памяти под информацию внутри сегмента данных программы. Такие данные существуют на протяжении всей жизни программы до ее завершения.

*Стек* — среда для размещения данных для возврата из подпрограмм, а также их аргументы и автоматические данные. Все это может потребовать достаточно большого размера стека. Как правило, программист может определять размер стека в программе.

*Страничная организация памяти* — организация, при которой адресное пространство памяти разбивается на малые участки — страницы.

***Регулятор*** — используется для установки параметров от минимального до максимального с помощью движка.

***Редактирование связей*** — разрешение внешних ссылок и создание исполняемого модуля из совокупности объектных. Основные функции редактора связей — распределение памяти, разрешение внешних ссылок.

***Режим доступа файла (ОС UNIX)***— определяет права процесса на выполнение операций над файлом (например, открытие файла на запись), является принадлежностью файла и устанавливается при его создании, может переопределяться с помощью вызова **chmod.** Режим доступа содержит информацию о возможности чтения, записи и выполнения файла для трех групп пользователей: владельца файла, членов его группы и всех остальных пользователей. Для каталогов признак «выполнения» интерпретируется как право просмотра его содержимого.

***Резидентная программа*** — программа, которая после загрузки в ОЗУ и передаче ей управления инициализируется таким образом, что постоянно находится в ОЗУ и выполняется параллельно другим программам.

***Ресурс (resource)*** — сетевой *объект,* такой, как принтер или набор связанных в каталог файлов, доступный для совместного использования.

***Свопинг*** — алгоритм реализации виртуальной памяти. Его можно разбить на три части: управление пространством на устройстве выгрузки, выгрузка процессов из основной памяти и подкачка процессов в основную память. В качестве устройства выгрузки используют раздел на устройстве типа жесткого диска или дисковый файл (swap-file) на таком устройстве.

***Сегментная виртуальная память*** — использует сегментно-страничное виртуальное адресное пространство. При этом требуется явное выделение частей программы. Реальная память также подвергается сегментной детализации по модульному признаку.

***Сегментно-страничная виртуальная память*** — каждый модуль программы подвергается дополнительному страничному структурированию. При этом размеры получаемых сегментов могут быть увеличены. В этом случае оперативная память подвергается также страничной структуризации. Такая схема реализует компромисс между операционной системой и пользователем, а именно: операционная система увеличивает пропускную способность, используя преимущества страничной по запросу схемы функционирования оперативной памяти, а пользователь избавлен от ограничений на размеры разрабатываемой программы.

*Сектор* — каждая дорожка, размещенная на диске, делится на секторы (блоки). Каждый сектор имеет размер 512 байт (для MS-DOS).

*Сервер* ***(server)*** — сетевой компьютер, на котором находятся доступные клиентам *ресурсы.* Ресурсами сервера могут быть файлы, принтеры или приложения сервера (такие, как многопользовательские базы данных).

*Сети типа «клиент-сервер»* ***(client-server networking)*** — сетевая архитектура, в которой предназначенные для совместного использования *ресурсы (resources)* сосредоточены на мощных компьютерах *серверах (server machines),* а подключенные к ним настольные машины играют роль *клиентов (clients),* посылая по сети запросы на ту или иную информацию.

*Системное программное обеспечение* — обеспечивает интерфейс между программистом или пользователем и аппаратной частью ЭВМ (операционная система, программы-оболочки), выполняет вспомогательные функции (программы-утилиты).

*Системные вызовы (ОС* ***UNIX)*** — представляют собой интерфейс между программами пользователя и ядром операционной системы UNIX. Реализация системных вызовов — одна из функций ядра ОС UNIX.

*Системные процессы (ОС* ***UNIX)*** — процессы с идентификаторами О, 1 считаются системными. Это: планировщик (процесс 0), инициализирующий процесс, который одновременно является родителем всех остальных процессов (процесс 1).

*Смежное размещение* ***(contiguous allocation)*** — размещение программ в памяти, предполагает, что в памяти, начиная с некоторого начального адреса, выделяется один непрерывный участок адресного пространства. При *несмежном размещении* программа разбивается на множество частей, которые располагаются в различных, необязательно смежных участках адресного пространства.

*Список* — элемент, содержащий все возможные в каждом конкретном случае значения, которые пользователь может установить. Добавить или изменить эти значения непосредственно в списке нельзя.

*Статическое выделение памяти* — выделение памяти под информацию внутри сегмента данных программы. Такие данные существуют на протяжении всей жизни программы до ее завершения.

*Стек* — среда для размещения данных для возврата из подпрограмм, а также их аргументы и автоматические данные. Все это может потребовать достаточно большого размера стека. Как правило, программист может определять размер стека в программе.

*Страничная организация памяти* — организация, при которой адресное пространство памяти разбивается на малые участки — страницы.

***Регулятор*** — используется для установки параметров от минимального до максимального с помощью движка.

***Редактирование связей*** — разрешение внешних ссылок и создание исполняемого модуля из совокупности объектных. Основные функции редактора связей — распределение памяти, разрешение внешних ссылок.

***Режим доступа файла (ОС UNIX)*** — определяет права процесса на выполнение операций над файлом (например, открытие файла на запись), является принадлежностью файла и устанавливается при его создании, может переопределяться с помощью вызова **chmod.** Режим доступа содержит информацию о возможности чтения, записи и выполнения файла для трех групп пользователей: владельца файла, членов его группы и всех остальных пользователей. Для каталогов признак «выполнения» интерпретируется как право просмотра его содержимого.

***Резидентная программа*** — программа, которая после загрузки в ОЗУ и передаче ей управления инициализируется таким образом, что постоянно находится в ОЗУ и выполняется параллельно другим программам.

***Ресурс (resource)*** — сетевой *объект,* такой, как принтер или набор связанных в каталог файлов, доступный для совместного использования.

***Свопинг*** — алгоритм реализации виртуальной памяти. Его можно разбить на три части: управление пространством на устройстве выгрузки, выгрузка процессов из основной памяти и подкачка процессов в основную память. В качестве устройства выгрузки используют раздел на устройстве типа жесткого диска или дисковый файл (swap-file) на таком устройстве.

***Сегментная виртуальная память*** — использует сегментно-страничное виртуальное адресное пространство. При этом требуется явное выделение частей программы. Реальная память также подвергается сегментной детализации по модульному признаку.

***Сегментно-страничная виртуальная память*** — каждый модуль программы подвергается дополнительному страничному структурированию. При этом размеры получаемых сегментов могут быть увеличены. В этом случае оперативная память подвергается также страничной структуризации. Такая схема реализует компромисс между операционной системой и пользователем, а именно: операционная система увеличивает пропускную способность, используя преимущества страничной по запросу схемы функционирования оперативной памяти, а пользователь избавлен от ограничений на размеры разрабатываемой программы.

***Сектор*** — каждая дорожка, размещенная на диске, делится на секторы (блоки). Каждый сектор имеет размер 512 байт (для MS-DOS).

*Сервер* ***(server)*** — сетевой компьютер, на котором находятся доступные клиентам *ресурсы.* Ресурсами сервера могут быть файлы, принтеры или приложения сервера (такие, как многопользовательские базы данных).

***Сети типа*** *«клиент-сервер»* ***(client-server networking)*** — сетевая архитектура, в которой предназначенные для совместного использования *ресурсы (resources)* сосредоточены на мощных компьютерах *серверах (servermachines),* а подключенные к ним настольные машины играют роль *клиентов (clients),* посылая по сети запросы на ту или иную информацию.

***Системное программное обеспечение*** — обеспечивает интерфейс между программистом или пользователем и аппаратной частью ЭВМ (операционная система, программы-оболочки), выполняет вспомогательные функции (программы-утилиты).

***Системные вызовы (ОС UNIX)*** — представляют собой интерфейс между программами пользователя и ядром операционной системы UNIX. Реализация системных вызовов — одна из функций ядра ОС UNIX.

***Системные процессы (ОС UNIX)*** — процессы с идентификаторами О, 1 считаются системными. Это: планировщик (процесс 0), инициализирующий процесс, который одновременно является родителем всех остальных процессов (процесс 1).

***Смежное размещение (contiguous allocation)*** — размещение программ в памяти, предполагает, что в памяти, начиная с некоторого начального адреса, выделяется один непрерывный участок адресного пространства. При *несмежном размещении* программа разбивается на множество частей, которые располагаются в различных, необязательно смежных участках адресного пространства.

***Список*** — элемент, содержащий все возможные в каждом конкретном случае значения, которые пользователь может установить. Добавить или изменить эти значения непосредственно в списке нельзя.

***Статическое выделение памяти*** — выделение памяти под информацию внутри сегмента данных программы. Такие данные существуют на протяжении всей жизни программы до ее завершения.

***Стек*** — среда для размещения данных для возврата из подпрограмм, а также их аргументы и автоматические данные. Все это может потребовать достаточно большого размера стека. Как правило, программист может определять размер стека в программе.

***Страничная организация памяти*** — организация, при которой адресное пространство памяти разбивается на малые участки — страницы.

***Регулятор*** — используется для установки параметров от минимального до максимального с помощью движка.

***Редактирование связей*** — разрешение внешних ссылок и создание исполняемого модуля из совокупности объектных. Основные функции редактора связей — распределение памяти, разрешение внешних ссылок.

***Режим доступа файла (ОС UNIX)*** — определяет права процесса на выполнение операций над файлом (например, открытие файла на запись), является принадлежностью файла и устанавливается при его создании, может переопределяться с помощью вызова **chmod.** Режим доступа содержит информацию о возможности чтения, записи и выполнения файла для трех групп пользователей: владельца файла, членов его группы и всех остальных пользователей. Для каталогов признак «выполнения» интерпретируется как право просмотра его содержимого.

***Резидентная программа*** — программа, которая после загрузки в ОЗУ и передаче ей управления инициализируется таким образом, что постоянно находится в ОЗУ и выполняется параллельно другим программам.

***Ресурс (resource)*** — сетевой *объект,* такой, как принтер или набор связанных в каталог файлов, доступный для совместного использования.

***Свопинг*** — алгоритм реализации виртуальной памяти. Его можно разбить на три части: управление пространством на устройстве выгрузки, выгрузка процессов из основной памяти и подкачка процессов в основную память. В качестве устройства выгрузки используют раздел на устройстве типа жесткого диска или дисковый файл (swap-file) на таком устройстве.

***Сегментная виртуальная память*** — использует сегментно-страничное виртуальное адресное пространство. При этом требуется явное выделение частей программы. Реальная память также подвергается сегментной детализации по модульному признаку.

***Сегментно-страничная виртуальная память*** — каждый модуль программы подвергается дополнительному страничному структурированию. При этом размеры получаемых сегментов могут быть увеличены. В этом случае оперативная память подвергается также страничной структуризации. Такая схема реализует компромисс между операционной системой и пользователем, а именно: операционная система увеличивает пропускную способность, Используя преимущества страничной по запросу схемы функционирования оперативной памяти, а пользователь избавлен от ограничений на размеры разрабатываемой программы.

*Сектор* — каждая дорожка, размещенная на диске, делится на секторы (блоки). Каждый сектор имеет размер 512 байт (для MS-DOS).

*Сервер* ***(server)*** — сетевой компьютер, на котором находятся доступные клиентам *ресурсы.* Ресурсами сервера могут быть файлы, принтеры или приложения сервера (такие, как многопользовательские базы данных).

*Сети типа «клиент-сервер»* ***(client-server networking)*** — сетевая архитектура, в которой предназначенные для совместного использования *ресурсы (resources)* сосредоточены на мощных компьютерах *серверах (server machines),* а подключенные к ним настольные машины играют роль *клиентов (clients),* посылая по сети запросы на ту или иную информацию.

*Системное программное обеспечение* — обеспечивает интерфейс между программистом или пользователем и аппаратной частью ЭВМ (операционная система, программы-оболочки), выполняет вспомогательные функции (программы-утилиты).

*Системные вызовы (ОС* ***UNIX)*** — представляют собой интерфейс между программами пользователя и ядром операционной системы UNIX. Реализация системных вызовов — одна из функций ядра ОС UNIX.

*Системные процессы (ОС* ***UNIX)*** — процессы с идентификаторами 0, 1 считаются системными. Это: планировщик (процесс 0), инициализирующий процесс, который одновременно является родителем всех остальных процессов (процесс 1).

*Смежное размещение* ***(contiguous allocation)*** — размещение программ в памяти, предполагает, что в памяти, начиная с некоторого начального адреса, выделяется один непрерывный участок адресного пространства. При *несмежном размещении* программа разбивается на множество частей, которые располагаются в различных, необязательно смежных участках адресного пространства.

*Список* — элемент, содержащий все возможные в каждом конкретном случае значения, которые пользователь может установить. Добавить или изменить эти значения непосредственно в списке нельзя.

*Статическое выделение памяти* — выделение памяти под информацию внутри сегмента данных программы. Такие данные существуют на протяжении всей жизни программы до ее завершения.

*Стек* — среда для размещения данных для возврата из подпрограмм, а также их аргументы и автоматические данные. Все это может потребовать достаточно большого размера стека. Как правило, программист может определять размер стека в программе.

*Страничная организация памяти* — организация, при которой адресное пространство памяти разбивается на малые участки — страницы.

***Регулятор*** — используется для установки параметров от минимального до максимального с помощью движка.

***Редактирование связей*** — разрешение внешних ссылок и создание исполняемого модуля из совокупности объектных. Основные функции редактора связей — распределение памяти, разрешение внешних ссылок.

***Режим доступа файла (ОС UNIX)***— определяет права процесса на выполнение операций над файлом (например, открытие файла на запись), является принадлежностью файла и устанавливается при его создании, может переопределяться с помощью вызова **chmod.** Режим доступа содержит информацию о возможности чтения, записи и выполнения файла для трех групп пользователей: владельца файла, членов его группы и всех остальных пользователей. Для каталогов признак «выполнения» интерпретируется как право просмотра его содержимого.

***Резидентная программа*** — программа, которая после загрузки в ОЗУ и передаче ей управления инициализируется таким образом, что постоянно находится в ОЗУ и выполняется параллельно другим программам.

***Ресурс (resource)*** — сетевой *объект,* такой, как принтер или набор связанных в каталог файлов, доступный для совместного использования.

***Свопинг*** — алгоритм реализации виртуальной памяти. Его можно разбить на три части: управление пространством на устройстве выгрузки, выгрузка процессов из основной памяти и подкачка процессов в основную память. В качестве устройства выгрузки используют раздел на устройстве типа жесткого диска или дисковый файл (swap-file) на таком устройстве.

***Сегментная виртуальная память*** — использует сегментно-страничное виртуальное адресное пространство. При этом требуется явное выделение частей программы. Реальная память также подвергается сегментной детализации по модульному признаку.

***Сегментно-страничная виртуальная память*** — каждый модуль программы подвергается дополнительному страничному структурированию. При этом размеры получаемых сегментов могут быть увеличены. В этом случае оперативная память подвергается также страничной структуризации. Такая схема реализует компромисс между операционной системой и пользователем, а именно: операционная система увеличивает пропускную способность, используя преимущества страничной по запросу схемы функционирования оперативной памяти, а пользователь избавлен от ограничений на размеры разрабатываемой программы.

*Сектор* — каждая дорожка, размещенная на диске, делится на секторы (блоки). Каждый сектор имеет размер 512 байт (для MS-DOS).

*Сервер (server)* — сетевой компьютер, на котором находятся доступные клиентам *ресурсы.* Ресурсами сервера могут быть файлы, принтеры или приложения сервера (такие, как многопользовательские базы данных).

*Сети типа «клиент-сервер» (client-server networking)* — "сетевая архитектура, в которой предназначенные для совместного использования *ресурсы (resources)* сосредоточены на мощных компьютерах *серверах (server machines),* а подключенные к ним настольные машины играют роль *клиентов (clients),* посылая по сети запросы на ту или иную информацию.

*Системное программное обеспечение* — обеспечивает интерфейс между программистом или пользователем и аппаратной частью ЭВМ (операционная система, программы-оболочки), выполняет вспомогательные функции (программы-утилиты).

*Системные вызовы (ОС UNIX)* — представляют собой интерфейс между программами пользователя и ядром операционной системы UNIX. Реализация системных вызовов — одна из функций ядра ОС UNIX.

*Системные процессы (ОС UNIX)* — процессы с идентификаторами 0, 1 считаются системными. Это: планировщик (процесс 0), инициализирующий процесс, который одновременно является родителем всех остальных процессов (процесс 1).

*Смежное размещение (contiguous allocation)* — размещение программ в памяти, предполагает, что в памяти, начиная с некоторого начального адреса, выделяется один непрерывный участок адресного пространства. При *несмежном размещении* программа разбивается на множество частей, которые располагаются в различных, необязательно смежных участках адресного пространства.

*Список* — элемент, содержащий все возможные в каждом конкретном случае значения, которые пользователь может установить. Добавить или изменить эти значения непосредственно в списке нельзя.

*Статическое выделение памяти* — выделение памяти под информацию внутри сегмента данных программы. Такие данные существуют на протяжении всей жизни программы до ее завершения.

*Стек* — среда для размещения данных для возврата из подпрограмм, а также их аргументы и автоматические данные. Все это может потребовать достаточно большого размера стека. Как правило, программист может определять размер стека в программе.

*Страничная организация памяти* — организация, при которой адресное пространство памяти разбивается на малые участки — страницы.

***Регулятор*** — используется для установки параметров от минимального до максимального с помощью движка.

***Редактирование связей*** — разрешение внешних ссылок и создание исполняемого модуля из совокупности объектных. Основные функции редактора связей — распределение памяти, разрешение внешних ссылок.

***Режим доступа файла (ОС UNIX)***— определяет права процесса на выполнение операций над файлом (например, открытие файла на запись), является принадлежностью файла и устанавливается при его создании, может переопределяться с помощью вызова **chmod.** Режим доступа содержит информацию о возможности чтения, записи и выполнения файла для трех групп пользователей: владельца файла, членов его группы и всех остальных пользователей. Для каталогов признак «выполнения» интерпретируется как право просмотра его содержимого.

***Резидентная программа*** — программа, которая после загрузки в ОЗУ и передаче ей управления инициализируется таким образом, что постоянно находится в ОЗУ и выполняется параллельно другим программам.

***Ресурс (resource)*** — сетевой *объект,* такой, как принтер или набор связанных в каталог файлов, доступный для совместного использования.

***Свопинг*** — алгоритм реализации виртуальной памяти. Его можно разбить на три части: управление пространством на устройстве выгрузки, выгрузка процессов из основной памяти и подкачка процессов в основную память. В качестве устройства выгрузки используют раздел на устройстве типа жесткого диска или дисковый файл (swap-file) на таком устройстве.

***Сегментная виртуальная память*** — использует сегментно-страничное виртуальное адресное пространство. При этом требуется явное выделение частей программы. Реальная память также подвергается сегментной детализации по модульному признаку.

***Сегментно-страничная виртуальная память*** — каждый модуль программы подвергается дополнительному страничному структурированию. При этом размеры получаемых сегментов могут быть увеличены. В этом случае оперативная память подвергается также страничной структуризации. Такая схема реализует компромисс между операционной системой и пользователем, а именно: операционная система увеличивает пропускную способность, Используя преимущества страничной по запросу схемы функционирования оперативной памяти, а пользователь избавлен от ограничений на размеры разрабатываемой программы.

*Сектор* — каждая дорожка, размещенная на диске, делится на секторы (блоки). Каждый сектор имеет размер 512 байт (для MS-DOS).

*Сервер* ***(server)*** — сетевой компьютер, на котором находятся доступные клиентам *ресурсы.* Ресурсами сервера могут быть файлы, принтеры или приложения сервера (такие, как многопользовательские базы данных).

*Сети типа «клиент-сервер»* ***(client-server networking)*** — сетевая архитектура, в которой предназначенные для совместного использования *ресурсы (resources)* сосредоточены на мощных компьютерах *серверах (server machines),* а подключенные к ним настольные машины играют роль *клиентов (clients),* посылая по сети запросы на ту или иную информацию.

*Системное программное обеспечение* — обеспечивает интерфейс между программистом или пользователем и аппаратной частью ЭВМ (операционная система, программы-оболочки), выполняет вспомогательные функции (программы-утилиты).

*Системные вызовы (ОС* ***UNIX)*** — представляют собой интерфейс между программами пользователя и ядром операционной системы UNIX. Реализация системных вызовов — одна из функций ядра ОС UNIX.

*Системные процессы (ОС* ***UNIX)*** — процессы с идентификаторами 0, 1 считаются системными. Это: планировщик (процесс 0), инициализирующий процесс, который одновременно является родителем всех остальных процессов (процесс 1).

*Смежное размещение* ***(contiguous allocation)*** — размещение программ в памяти, предполагает, что в памяти, начиная с некоторого начального адреса, выделяется один непрерывный участок адресного пространства. При *несмежном размещении* программа разбивается на множество частей, которые располагаются в различных, необязательно смежных участках адресного пространства.

*Список* — элемент, содержащий все возможные в каждом конкретном случае значения, которые пользователь может установить. Добавить или изменить эти значения непосредственно в списке нельзя.

*Статическое выделение памяти* — выделение памяти под информацию внутри сегмента данных программы. Такие данные существуют на протяжении всей жизни программы до ее завершения.

*Стек* — среда для размещения данных для возврата из подпрограмм, а также их аргументы и автоматические данные. Все это может потребовать достаточно большого размера стека. Как правило, программист может определять размер стека в программе.

*Страничная организация памяти* — организация, при которой адресное пространство памяти разбивается на малые участки — страницы.

***Регулятор*** — используется для установки параметров от минимального до максимального с помощью движка.

***Редактирование связей*** — разрешение внешних ссылок и создание исполняемого модуля из совокупности объектных. Основные функции редактора связей — распределение памяти, разрешение внешних ссылок.

***Режим доступа файла (ОС UNIX)*** — определяет права процесса на выполнение операций над файлом (например, открытие файла на запись), является принадлежностью файла и устанавливается при его создании, может переопределяться с помощью вызова **chmod.** Режим доступа содержит информацию о возможности чтения, записи и выполнения файла для трех групп пользователей: владельца файла, членов его группы и всех остальных пользователей. Для каталогов признак «выполнения» интерпретируется как право просмотра его содержимого.

***Резидентная программа*** — программа, которая после загрузки в ОЗУ и передаче ей управления инициализируется таким образом, что постоянно находится в ОЗУ и выполняется параллельно другим программам.

***Ресурс (resource)*** — сетевой *объект,* такой, как принтер или набор связанных в каталог файлов, доступный для совместного использования.

***Свопинг*** — алгоритм реализации виртуальной памяти. Его можно разбить на три части: управление пространством на устройстве выгрузки, выгрузка процессов из основной памяти и подкачка процессов в основную память. В качестве устройства выгрузки используют раздел на устройстве типа жесткого диска или дисковый файл (swap-file) на таком устройстве.

***Сегментная виртуальная память*** — использует сегментно-страничное виртуальное адресное пространство. При этом требуется явное выделение частей программы. Реальная память также подвергается сегментной детализации по модульному признаку.

***Сегментно-страничная виртуальная память*** — каждый модуль программы подвергается дополнительному страничному структурированию. При этом размеры получаемых сегментов могут быть увеличены. В этом случае оперативная память подвергается также страничной структуризации. Такая схема реализует компромисс между операционной системой и пользователем, а именно: операционная система увеличивает пропускную способность, используя преимущества страничной по запросу схемы функционирования оперативной памяти, а пользователь избавлен от ограничений на размеры разрабатываемой программы.

*Сектор* — каждая дорожка, размещенная на диске, делится на секторы (блоки). Каждый сектор имеет размер 512 байт (для MS-DOS).

*Сервер* ***(server)*** — сетевой компьютер, на котором находятся доступные клиентам *ресурсы.* Ресурсами сервера могут быть файлы, принтеры или приложения сервера (такие, как многопользовательские базы данных).

*Сети типа «клиент-сервер»* ***(client-server networking)*** — сетевая архитектура, в которой предназначенные для совместного использования *ресурсы (resources)* сосредоточены на мощных компьютерах *серверах (server machines),* а подключенные к ним настольные машины играют роль *клиентов (clients),* посылая по сети запросы на ту или иную информацию.

*Системное программное обеспечение* — обеспечивает интерфейс между программистом или пользователем и аппаратной частью ЭВМ (операционная система, программы-оболочки), выполняет вспомогательные функции (программы-утилиты).

*Системные вызовы (ОС* ***UNIX)*** — представляют собой интерфейс между программами пользователя и ядром операционной системы UNIX. Реализация системных вызовов — одна из функций ядра ОС UNIX.

*Системные процессы (ОС* ***UNIX)*** — процессы с идентификаторами О, 1 считаются системными. Это: планировщик (процесс 0), инициализирующий процесс, который одновременно является родителем всех остальных процессов (процесс 1).

*Смежное размещение* ***(contiguous allocation)*** — размещение программ в памяти, предполагает, что в памяти, начиная с некоторого начального адреса, выделяется один непрерывный участок адресного пространства. При *несмежном размещении* программа разбивается на множество частей, которые располагаются в различных, необязательно смежных участках адресного пространства.

*Список* — элемент, содержащий все возможные в каждом конкретном случае значения, которые пользователь может установить. Добавить или изменить эти значения непосредственно в списке нельзя.

*Статическое выделение памяти* — выделение памяти под информацию внутри сегмента данных программы. Такие данные существуют на протяжении всей жизни программы до ее завершения.

*Стек* — среда для размещения данных для возврата из подпрограмм, а также их аргументы и автоматические данные. Все это может потребовать достаточно большого размера стека. Как правило, программист может определять размер стека в программе.

*Страничная организация памяти* — организация, при которой адресное пространство памяти разбивается на малые участки — страницы.

Используется для управления памятью в системах, работающих в защищенном режиме. Как правило, такая организация памяти подразумевает *пейджинг.*

***Страничная по запросу виртуальная память*** — снимает требование полного расположения программы в оперативной памяти, то есть в оперативной памяти может размещаться блок программы и снимается ограничение на размер виртуальной памяти. Отображение происходит динамически и по частям.

***Супервизор*** — программа многозадачной ОС, обеспечивающая наилучшее использование ресурсов ЭВМ при одновременном выполнении нескольких задач.

***Таблица управления процессом*** (PCB — proce$$ control block). В PCB процесс описывается набором значений, параметров, характеризующих его текущее состояние и используемых операционной системой для управления прохождением процесса через компьютер.

***Трансляция*** — получение объектного кода из исходного.

***Управление процессами*** — обеспечивает повышение производительности операционной системы за счет организации параллельной работы процессора с внешними устройствами различного быстродействия. Решение этой задачи связано с управлением памятью, так как процесс может развиваться только в оперативной памяти. Реализация управления процессами требует дополнительных ресурсов времени и памяти. Управление процессами в любой операционной системе реализуется с помощью специальной структуры — дескриптора процесса, которая содержит основную информацию о процессе.

***Управляющие кнопки*** (Button) — предназначены для выполнения действий. Какое именно действие выполняет кнопка, написано непосредственно на ней. Кнопка приводится в действие нажатием мыши на ней. Если в конце названия кнопки присутствуют три точки, то такая кнопка вызовет новое диалоговое окно.

***Утилизация (использование) CPU (utilization)*** — критерий эффективности планирования. Утилизация CPU теоретически может находиться в пределах от 0 до 100 %. В реальных системах утилизация CPU колеблется в пределах 40—90 %.

***Флажок*** (или Независимый переключатель, CheckBox) — переключатель для режима работы, описание которого находится справа от квадрата. Он может быть включен (установлен) — внутри квадрата изображен значок, или выключен (сброшен) — внутри пусто. Для установки или сброса флажка необходимо щелкнуть мышью в квадрате или на его описании. Такой элемент вполне самостоятельно определяет свой параметр и поэтому называется независимым, в отличие от следующего элемента.

***флоппи-диск*** (дискета) — съемный гибкий магнитный диск.

***Цилиндр*** — объединение дорожек с одним и тем же номером, расположенных на разных поверхностях диска (для флоппи-диска под цилиндром подразумеваются 2 дорожки). Цилиндр — пространство, доступное для записи-считывания при фиксированном положении блока головок дисковода.

***Чисто страничная виртуальная память*** — схема виртуальной памяти, основанная на том, что выделяется вся необходимая память для реализации программы или блока программ, то есть ставится условие, что программа целиком располагается в оперативной памяти.

***Шина (bus)*** — устройство для организации интерфейса с другими устройствами. К шине подключаются платы адаптеров.

***Эффективные идентификаторы пользователя и группы, список групп доступа. (ОС UNIX).*** Доступ к системным ресурсам определяется тремя значениями: «эффективным идентификатором пользователя», «эффективным идентификатором группы» и «списком групп доступа». Эффективные идентификаторы пользователя и группы при старте процесса обычно совпадают с реальными. Исключение составляет случай, когда статус выполняемого файла содержит **признаки set-UID и set-GID** (см. описание вызова **execve).** Список групп доступа вместе с идентификатором группы используется для определения прав доступа к системным ресурсам.

***ро ОС UNIX*** — программа, которая обеспечивает разделение времени центрального процессора между выполняющимися процессами; осуществляет управление памятью и устройствами ввода-вывода, реализует файловую систему.

# Приложение 1

## Сравнительные характеристики ОС и оболочек семейства WINDOWS

| | Windows 3.11 | Wiondows95 | Wiondows NT 3.51 | Wiondows NT 4.0 |
|---|---|---|---|---|
| Рекомендуемый объем ОЗУ, Мбайт | 2 | 8 | 16 | 16 |
| Требуемый процессор | 386SX | 486DX | 486DX | Pentium-60 |
| Минимально необходимое пространство на диске, Мбайт | 7 | 30 | 90 | 120 |
| Файловые системы | PAT | PAT, PAT32 | PAT, NTFS, HPP3 | PAT, NTFS, HPP3 |
| Поддержка Plug&Play | нет | да | | |
| Поддержка APM | нет | да | нет | нет |
| Пароль при запуске ПК | нет | да | Да | да |
| Программы DOS | да | да | да | да |
| Программы Windows 3.1 | да | да | да | да |
| Вместе в одной виртуальной DOS-машине | да | да | нет | нет |
| Каждая в своей виртуальной DOS-машине | нет | нет | да | да |
| 32-разрядные программы Windows95 | нет | да | да | да |
| Стандартные клиенты сети и протоколы | Microsoft Windows Network, Microsoft Mail, Shedule+ | Microsoft Exchange, Microsoft Network, Novell Netware 3.x, 4.x, IPX/SPX, Microsoft DLC, NetBEUI, TCP/IP | Client Service тог NetWare, PTP Server, PTP telnet, SLIP, PPP, TCP/IP, Remote Access, AppleTalk, DLC, NetBEUI, IPX/SPX, TCP/IP | Client Service (or NetWare, PTP Server, PTP telnet, SLIP, PPP, TCP/IP, Remote Access, AppleTalk, DLC, NetBEUI, IPX/SPX, TCP/IP |

# Приложение 2

## Сравнительные аппаратные требования ОС для ПК

| ОС | Требования - минимальные/рекомендуемые | | |
|---|---|---|---|
| | Оперативная память | Процессор | Места на НЖМД |
| MS-DOS | 640Кб/4Мб | 80286/80386 или совместимый | 10 Мб/40 Мб |
| Windows 3.1 | 1 Мб/4 Мб | 80286/80386 или совместимый | 20 Мб/80 Мб |
| Windows 3.11 | 2 Мб/8 Мб | 80386/80486 или совместимый | 40 Мб/100 Мб |
| Windows95 | 8 Мб/16 Мб | 80486/IP*)-133 МГц или совместимый | 150 Мб/250 Мб |
| Windows98 | 16 Мб/32 Мб | IP-200 МГц /IP-233 МГц или совместимый | 250 Мб/500 Мб |
| Windows NT 3.51 | 16 Мб/24 Мб | IP-60 МГц/IP-133 МГц или совместимый | 200 Мб/400 Мб |
| Windows NT 4.0 | 32 Мб/64 Мб | IP-166 МГц/IP-200 МГц или совместимый | 300 Мб/500 Мб |
| Windows NT 2000 | 32 Мб/64 Мб | IP-200 МГц /IP-233 МГц или совместимый | 300 Мб/500 Мб |
| 05/2 | 4 Мб/8 Мб | 80386/80486 или совместимый | 80 Мб/150 Мб |
| Linux | 4 Мб/32 Мб | 80386/IP-166 МГц или совместимый | 100 Мб/600 Мб |

*) Intel Pentium Commander.

# Приложение 3

## Использование функциональных клавиш NORTON COMMANDER

| | |
|---|---|
| <F1> | Вызов помощи |
| <F2> | Меню пользователя |
| <P3> | Просмотр файла |
| <P4> | Редактирование файла |
| <P5> | Копирование файла (группы файлов) |
| <P6> | Переименование или перенос файла (группы файлов) |
| <P7> | Создание каталога |
| <P8> | Удаление файла или каталога (группы файлов) |
| <P9> | Управляющее меню Norton Commander |
| <P10> | Выход из Norton Commander |
| <Ctrl+F1> | Убрать/восстановить левую панель экрана |
| <Ctrl+F2> | Убрать/восстановить правую панель экрана |
| <Ctrl+F3> | Сортировка файлов по имени (по алфавиту) |
| <Ctrl+F4> | Сортировка файлов по расширению |
| <Ctrl+F5> | Сортировка файлов по времени создания |
| <Ctrl+F6> | Сортировка файлов по размеру |
| <Ctrl+F7> | Сортировка файлов не устанавливается |
| <Ctrl+F9> | Печать файла |
| <Shift+<F4>> | Создание текстового файла. Имя файла задается |
| <Shift+F9> | Сохранение настроек Norton Commander |
| <Alt+F1> (<Alt+F2>) | Переключение на другой дисковод на левой (правой) панели |
| <Alt+F4> | Редактирование файла с помощью альтернативного редактора |

| | |
|---|---|
| <Alt+F5> | Создание архива |
| <Alt+F6> | Извлечение файлов из архива |
| <Alt+F7> | Поиск файла на диске |
| <Alt+F8> | История ранее выполненных команд |
| <Alt+F9> | Переключение режима монитора с 25 на 43 линии |
| <Alt+F10> | Вывод дерева каталогов, поиск каталога на диске. При этом в корневом каталоге создается файл treeinfo.ncd, в котором содержится информация о всем дереве каталогов |
| <Ins> | Выделение одного файла |
| <*> (на левой части клавиатуры) | Выделение всех файлов в каталоге или инверсия выделения файлов |
| <++Enter> | Выделение всех файлов в каталоге |
| <-+Eп(ег> | Отмена выделения файлов |
| <Ctrl+O> | Убрать/восстановить панели экрана |
| <Ctrl+U> | Поменять панели местами |
| <Ctrl+P> | Убрать/восстановить не текущую панель экрана |
| <Ctrl+L> | Вывод информационной панели диска (дискеты) |
| <Ctrl+Q> | Быстрый просмотр файла, информация о содержимом каталога |
| <Ctrl+E> | Вывод ранее выполненной команды из буфера History |
| <Ctrl+X> | Вывод следующей выполненной команды из буфера History |
| <Ctrl+J> (или <Ctrl+Enter>) | Перенос выделенного курсором файла в командную строку |
| <Ctrl+R> | Быстрое обновление информации на панели диска (дискеты) |
| <Ctrl+Z> | Паспорт каталога, информация о содержимом каталога и число дискет формата 1,2 и 1,44 Мб, на которых эта информация может быть размещена |
| <Ctrl+\> | Быстрый переход из подкаталога в корневой каталог |
| <TaЬ> | Перемещение курсора с левой панели на правую и обратно |
| <Esc> | Отмена команд в Norton Commander |
| Поиск подстроки при просмотре файла - <Р3>, F7, ввести искомое слово и <Enter> | |

# Приложение 4

## Команды и функции FAR MANAGER

*Табл. П4.1.* **Редактирование командной строки**

| Действие | Команда (клавиши) |
|---|---|
| Символ влево | <←>, <Ctrl+S> |
| Символ вправо | <→>, <Ctrl+D> |
| Слово влево | <Ctrl+←> |
| Слово вправо | <Ctrl+→> |
| В начало строки | <Ctrl+Home> |
| В конец строки | <Ctrl+End> |
| Удалить символ | <Del> |
| Удалить символ слева | <BS> |
| Удалить до конца строки | <Ctrl+K> |
| Удалить слово слева | <Ctrl+BS> |
| Удалить слово справа | <Ctrl+Del> |
| Рассматривать следующую комбинацию клавиш как код | <Ctrl+Q> |
| Копировать в Буфер Обмена | <Ctrl+Ins> |
| Вставить из Буфера Обмена | <Shift+Ins> |
| Предыдущая команда | <Ctrl+E> |
| Следующая команда | <Ctrl+X> |
| Очистить командную строку | <Ctrl+Y> |
| Вставить имя файла из активной панели | <Ctrl+Enter> |
| Вставить полное имя файла из активной панели | <Ctrl+F> |
| Вставить путь из левой панели | <Ctrl+[> |
| Вставить путь из правой панели | <Ctrl+]> |
| Вставить путь из активной панели | <Ctrl+Shift+[> |
| Вставить путь из пассивной панели | <Ctrl+Shift+]> |

***Примечания:***

1. Если командная строка пуста, <Ctrl+Ins> будет копировать имена выбранных в панели файлов в Буфер Обмена так же, как и <Ctrl+Shift+Ins>.

2. <Ctrl+End>, нажатая в конце командной строки, заменяет ее текущее содержимое командой из **Истории Команд,** начинающейся с уже введенных букв, если такая команда существует. Чтобы перейти к следующей такой команде, вы можете нажать <Ctrl+End> повторно.

3. Большинство из описанных выше команд действительно для всех строк редактирования, включая строки в диалогах и встроенный редактор.

*Табл. П4.2.* **Общие команды управления панелями**

| Действие | Команда (клавиши) |
|---|---|
| Изменить активную панель | <Tab> |
| Поменять панели местами | <Ctrl+U> |
| Перечитать содержимое панели | <Ctrl+R> |
| Убрать/показать информационную панель | <Ctrl+L> |
| Убрать/показать панель быстрого просмотра | <Ctrl+Q> |
| Убрать/показать дерево папок | <Ctrl+T> |
| Убрать/показать обе панели | <Ctrl+O> |
| Убрать/показать неактивную панель | <Ctrl+P> |
| Убрать/показать левую панель | <Ctrl+F1> |
| Убрать/показать правую панель | <Ctrl+F2> |
| Изменить высоту панелей | <Ctrl+↑>, <Ctrl+↓> |
| Изменить ширину (при пустой командной строке) | <Ctrl+←>,<Ctrl+→> |
| Восстановить ширину панелей по умолчанию | <Ctrl+Num5> |

*Табл. П4.3.* **Команды файловой панели**

| | |
|---|---|
| Пометить/снять пометку файла | <Ins>, <Shift+Клавиши курсора> |
| Пометить группу | <Num+> |
| Снять пометку с группы | <Num-> |
| Инвертировать пометку | <Num*> |
| Пометить файлы с расширением как у текущего файла | <Ctrl+Num+> |
| Снять пометку с файлов с расширением как у текущего | <Ctrl+Num-> |
| Инвертировать пометку, включая папки | <Ctrl+Num*> |

*Продолжение табл. П4.3*

| | |
|---|---|
| Пометить файлы с именем как у текущего файла | <Alt+Num+> |
| Снять пометку с файлов с именем как у текущего файла | <Alt+Num-> |
| Пометить все файлы | <Shift+Num+> |
| Снять пометку со всех файлов | <Shift+Num-> |
| Поместить помеченные имена в Буфер Обмена | <Ctrl+Shift+Ins> |
| Восстановить предыдущую пометку | <Ctrl+M> |
| Прокрутка длинных имен и описаний | <Alt+←>, <Alt+→> |
| Установить краткий режим просмотра | <ЛевыйCtrl+1> |
| Установить средний режим просмотра | <ЛевыйCtrl+2> |
| Установить полный режим просмотра | <ЛевыйCtrl+3> |
| Установить широкий режим просмотра | <ЛевыйCtrl+4> |
| Установить детальный режим просмотра | <ЛевыйCtrl+5> |
| Установить режим просмотра описаний | <ЛевыйCtrl+6> |
| Установить режим просмотра длинных описаний | <ЛевыйCtrl+7> |
| Установить режим просмотра владельцев файлов | <ЛевыйCtrl+8> |
| Установить режим просмотра связей файлов | <ЛевыйCtrl+9> |
| Установить альтернативный полный режим просмотра | <ЛевыйCtrl+0> |
| Убрать/показать файлы с атрибутом Скрытый и Системный | <Ctrl+H> |
| Переключить вывод длинных/коротких имен файлов | <Ctrl+N> |
| Спрятать/показать левую панель | <Ctrl+F1> |
| Спрятать/показать правую панель | <Ctrl+F2> |
| Сортировать файлы активной панели по имени | <Ctrl+F3> |
| Сортировать файлы активной панели по расширению | <Ctrl+F4> |
| Сортировать файлы активной панели по времени модификации | <Ctrl+F5> |
| Сортировать файлы активной панели по размеру | <Ctrl+F6> |
| Не сортировать файлы активной панели | <Ctrl+F7> |
| Сортировать файлы активной панели по времени создания | <Ctrl+F8> |
| Сортировать файлы активной панели по времени доступа | <Ctrl+F9> |
| Сортировать файлы активной панели по описаниям | <Ctrl+F10> |
| Сортировать файлы активной панели по владельцу | <Ctrl+F11> |
| Вывести меню режимов сортировки | <Ctrl+F12> |
| Использовать сортировку по группам | <Shift+F11> |
| Показывать помеченные файлы первыми | <Shift+F12> |

***Примечания:***

1. Если командная строка пуста, <Ctrl+Ins> будет копировать меню выбранных в панели файлов в Буфер Обмена так же, как и <Ctrl+Shift+Ins>.

2. Если включена опция «Разрешить обратную сортировку» в диалоге Настроек панели, то повторное нажатие одной и той же клавиши сортировки файлов приводит к смене направления сортировки с возрастающей на убывающую и наоборот.

3. Комбинации <Alt+→> и <Alt+←>, используемые для прокрутки длинных имен и описаний, работают только с клавишами <→> и <←>, расположенными на основной (левой) части клавиатуры. Так происходит потому, что при нажатой клавиши <Alt> управления курсором на Num используются для ввода символов через их десятичные коды.

*Табл. П4.4.* **Использование функциональной клавиатуры для управления файлами и выполнения сервисных комаид**

| | | |
|---|---|---|
| Помощь | <F1> | |
| Вызвать Пользовательское меню | <F2> | |
| Просмотр | <Ctrl+Shift+F3>, <Num5>, <F3> | Если Num5 или F3 нажаты при курсоре |
| Редактирование, распаковка | <Ctrl+Shift+F4>, <F4> | <F4> вызывает встроенный редактор, внешний или ассоциированный с файлом в зависимости от типа файла и настроек редактора. <Ctrl+Shift+F4> всегда вызывает встроенный редактор вне зависимости от файловых ассоциаций |
| Копирование | <F5> | Копирует файлы и папки. Если требуется создать папку назначения перед копированием, следует добавить к ее имени обратную черту (\) |
| Переименование или перенос | <F6> | Переименование или перенос файлов и папок. Если вы хотите создать папку назначения перед переносом, добавьте к ее имени обратную черту (\) |
| Создание новой папки | <F7> | |
| Удаление | <Shift+Del>, <Shift+F8>, <F8> | Удаление файлов и папок. <F8> и <Shift+Del> удаляют все выбранные файлы, <Shift+F8> – только файл под курсором. <Shift+Del> всегда удаляет файлы, не используя Корзину (Recycle Bin). Использование Корзины командами <F8> и <Shift+F8> зависит от конфигурации |
| Уничтожение файлов и папок | <Alt+Del> | Уничтожает файлы и папки. Перед удалением файл перезаписывается нулями, усекается до нулевой длины и переименовывается во временное имя |
| Показать Горизонтальное Меню | <F9> | |
| Завершить работу с FAR | <F10> | |
| Показать команды Подключаемых Модулей (Plugins) | <F11> | |

*Табл. П4.5.* **Команды с использованием ФК и клавиш <Alt>, <Shift>**

| | | |
|---|---|---|
| Изменить текущий диск в левой панели | <Alt+F1> | |
| Изменить текущий диск в правой панели | <Alt+F2> | |
| Встроенная/внешняя программа просмотра | <Alt+F3> | Вызывает внешнюю программу просмотра, если по умолчанию используется внутренняя, и внутреннюю, если по умолчанию используется внешняя |
| Встроенный/внешний редактор | <Alt+F4> | Вызывает внешний редактор, если по умолчанию используется внутренний, и внутренний, если по умолчанию используется внешний |
| Печать файлов | <Alt+F5> | |
| Создать жесткую связь (только на NTFS) | <Alt+F6> | Используя жесткие связи файлов, можно иметь несколько различных имен файлов, ссылающихся на одни и те же данные |
| Поиск файла | <Alt+F7> | |
| Показать Историю Команд | <Alt+F8> | |
| Переключение между 25 и 50 строками на экране | <Alt+F9> | |
| Поиск папки | <Alt+F10> | |
| Показать историю просмотра и редактирования | <Alt+F11> | |
| Показать историю папок | <Alt+F12> | |
| Добавить файлы к архиву | <Shift+F1> | |
| Извлечь файлы из архива | <Shift+F2> | |
| Выполнить команды управления архивом | <Shift+F3> | |
| Редактировать новый файл | <Shift+F4> | |
| Копирование файла под курсором | <Shift+F5> | |
| Переименование или перенос файла под курсором | <Shift+F6> | |
| Удаление файла под курсором | <Shift+F8> | |
| Сохранить конфигурацию | <Shift+F9> | |
| Выбрать последний выполненный пункт меню | <Shift+F10> | |

*Табл. П4.6.* **Прочие команды**

| | |
|---|---|
| Запуск, смена папки, вход в архив | <Enter> |
| Запуск в отдельном окне | <Shift+Enter> |
| Сменить папку на корневую | <Ctrl+\> |
| Смена папки, вход в архив (также в ZPX архив) | <Ctrl+PgDn> |
| Перейти в. папку уровнем выше | <Ctrl+PgUp> |
| Создать ссылку на текущую папку | <Ctrl+Shift+N> |
| Использовать ссылку на папку | <ПравыйCtrl+N> |
| Установить атрибуты файлов | <Ctrl+A> |
| Применить Команду к помеченным файлам | <Ctrl+G> |
| Добавить Описания к помеченным файлам | <Ctrl+Z> |
| Записать клавиатурную макрокоманду | <Ctrl+<.> |
| Очистка истории в строках редактирования диалогов | <Del> |

*Табл. П4.7.* **Дополнительные возможности**

| | | |
|---|---|---|
| Копирование текста с экрана | <Alt+Ins> | Эта команда позволяет выбрать и поместить в Буфер Обмена любую область экрана. Для перемещения курсора следует использовать клавиши управления курсором или левую кнопку мыши. Для выбора текста необходимо использовать клавиши управления курсором при нажатой <Shift> или нажатую левую кнопку мыши. <Enter>, <Ctrl+Ins>, правая кнопка мыши или двойное нажатие левой кнопки мыши копируют выбранный текст в Буфер Обмена, <Ctrl+Num +> добавляет его к текущему содержанию Буфера Обмена, <Esc> отменяет пометку и завершает операцию |
| История в строках редактирования диалогов | <Ctrl+↑>, <Ctrl+↓> | В истории строк редактирования диалогов можно использовать <Enter> для копирования текущего элемента в строку редактирования или <Ins> для отметки элемента. Отмеченные элементы не вытесняются из истории новыми элементами, так что можно отметить часто используемые строки, чтобы все время иметь их в истории |
| Вставить имя файла под курсором в диалог | <Shift+Enter> | Эта комбинация клавиш может быть использована во всех строках редактирования, кроме командной строки, включая диалоги и Встроенный Редактор |

# Приложение 5

## Управляющие и горячие клавиши оболочки MS-DOS SHELL

| Команды общего назначения | |
|---|---|
| <En(er> | Выполнение команды или операции |
| <Esc> | Отмена текущей команды или операции |
| <P1> | Вывод справочной информации о выбранном поле, команде или опциях диалоговой рамки |
| <P3> или <Alt+F4> | Выход из оболочки MS-DOS Shell и переход к командной строке DOS |
| <Shift+F5> | Вывод изображения на экран заново. Эта команда эквивалентна команде Repaint Screen. Информационный кадр не обновляется |
| <Shift+F9> | Переход к командной строке DOS с сохранением оболочки MS-DOS ЗпеН в памяти |
| Команды перемещения | |
| <F10> или <Alt> | Выбор (активизация) строки основного меню в верхней части информационного кадра |
| <Tab> | Переход в следующее окно оболочки MS-DOS ЗпеН или диалоговой рамки |
| <Shift+Tab> | Переход к предыдущему окну оболочки MS-DOS Shell или диалоговой рамки |
| <Home> | Перемещение к началу строки или списка |
| <End> | Перемещение к концу строки или списка |
| <Ctrl+Home> | Перемещение к началу списка |
| <Ctrl+End> | Перемещение к концу списка |
| <PgUp> | Прокрутка к предыдущей странице выводимой на экран информации |
| <PgDn> | Прокрутка к следующей странице выводимой на экран информации |
| Команды работы со справочной информацией | |
| <Enter> | Выполнение команды или операции или вывод на экран выбранной темы |
| <Esc> | Закрытие текущего окна справочника |
| <P1> | Вывод справочного материала о том, как работать со справочником |
| <Tab> | Переход к следующей экранной клавише темы справочника или к следующему заголовку темы |
| <Shift+Tab> | Переход к предыдущей экранной клавише темы справочника или к предыдущему заголовку темы |
| <PgUp> | Прокрутка к предыдущей странице выводимой на экран информации |
| <PgDn> | Прокрутка к следующей странице выводимой на экран информации |

| | |
|---|---|
| <T> или <↓> | Прокрутка изображения вверх или вниз строка за строкой |
| Команды работы с активными задачами | |
| Для использования нижеперечисленных команд следует активизировать механизм свопинга задач | |
| <Shift+Enter> | Запуск программы и добавление ее в список активных задач без выхода из оболочки MS-DOS Shell |
| <Ctrl+Esc> | Выход из программы в оболочку MS-DOS Shell |
| При нажатой клавише <Alt> повторными нажатиями клавиши <Tab> вы придете к требуемой программе | |
| Переход к другой программе путем циклической прокрутки по списку программ | |
| <Alt+Tab> | Переход между двумя программами |
| <Alt+Esc> | Переход к следующей программе |
| <Shift+Alt+Esc> | Переход к предыдущей программе |
| <Shift+Alt+Tab> | Циклический обход программ назад |
| <Ctrl+буква>, <Alt+буква>, <Shift+буква>, <Alt+Ctrl+буква>, <Shift+Ctrl+буква>, <Shift+Alt+буква> | Горячие клавиши, определяемые пользователем с помощью диалоговой рамки File-New-Program Item-Add Program. Служат для переключения на программу |
| Команды работы со списком программных групп и элементов | |
| <F2> | Копирование программ из списка с помощью команды Copy |
| <Del> | Удаление из списка выбранной программы или группы |
| Нижеперечисленные команды могут использоваться, только если активизирован механизм свопинга задач | |
| <Shift+Enter> | Добавление выбранного программного элемента в список активных задач без переключения на программу |
| <Shift+Ctri+Enter> | Добавление выбранного программного элемента в список активных задач без переключения на программу. При переключении на эту программу она автоматически открывает файл, указанный в диалоговой рамке File-Properties |
| Команды работы со списком файлов | |
| <P5> | Обновление дерева каталогов и списка файлов. Эта команда эквивалентна команде View-Refresh основного меню |
| <Ctrl+F5> | Обновление списка файлов текущего каталога |
| <P7> | Перемещение выделенного файла или файлов из одного каталога в другой. Эта команда эквивалентна команде File-Move основного меню |
| <P8> | Копирование выделенного файла или файлов из одного каталога в другой. Эта команда эквивалентна команде File-Copy основного меню |
| <P9> | Вывод на экран содержимого файла. Эта команда эквивалентна команде File-View File Contents основного меню |

| | |
|---|---|
| <Del> | ) Удаление выделенного файла или файлов |
| Следующая команда используется в режиме просмотра содержимого файла: | |
| <F9> - | Переключение между текстовым и 16-ричным режимами вывода. |
| Следующая команда используется при активизированном механизме свопинга: | |
| <Shift>+<Enter> | Добавление выбранного программного элемента в список активных задач без переключения на программу |
| Команды выделения файлов | |
| <Shift+F8> | Включение и выключение режима добавления. Этот режим позволяет выделять файлы, находящиеся в произвольных позиция списка (не рядом). Если режим добавления включен, в левом нижнем углу окна оболочки MS-DOS Shell выводится слово Add |
| <Shift+↑> | К числу выделенных добавляется предыдущий по списку файл |
| <Shift+↓> | К числу выделенных следующий по списку файл |
| <Shift+PgUp> | К числу выделенных добавляются файлы предыдущей страницы списка файлов |
| <Shift+PgDn> | К числу выделенных добавляются файлы следующей страницы списка файлов |
| <Ctrl+/> | Выделяются все файлы списка |
| <Ctrl+\> | Отменяется выделение файлов списка. Следующие команды действуют в режиме добавления |
| <Shift+Space> | Выделяются файлы, расположенные в списке между предыдущим выделенным файлом и курсором |
| <5pace> | К числу выделенных добавляется файл в позиции курсора |
| Команды работы с деревом каталогов | |
| <T> или <↑> | Перемещение курсора вверх или вниз к следующему каталогу |
| <Ctrl+*> | Вывод в кадр всех каталогов дерева |
| <-> | В кадре не показываются подкаталоги выбранного каталога |
| <+> | В кадр выводится один нижележащий уровень подкаталогов выбранного каталога |
| <*> | Вывод в кадр всех подкаталогов выбранного каталога |
| Команды выбора диска | |
| <3pace> | Вывод в кадр дерева каталогов выбранного диска |
| <Ctrl+6уква> | Чтение диска и вывод в кадр его каталогов. Команда эквивалентна выбору буквы требуемого диска и нажатию пробельной клавиши |
| <Enter> | Чтение выбранного диска и вывод в кадр его содержимого |
| <F5> | Обновление изображения дерева каталогов и списка файлов. Эта команда эквивалентна команде View-Refresh основного меню |
| <Ctrl+F5> | Обновление списка файлов для текущего каталога |

# Содержание