МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
ЮРГИНСКИЙ ТЕХНОЛОГИЧЕСКИЙ ИНСТИТУТ

А.А. Захарова, Е.В. Молнина, Т.Ю. Чернышева

ИНФОРМАТИКА И ПРОГРАММИРОВАНИЕ: ПРОГРАММНЫЕ СРЕДСТВА РЕАЛИЗАЦИИ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ

Рекомендовано Учебно-методическим объединением по образованию в области прикладной информатики в качестве учебника для студентов высших учебных заведений, обучающихся по специальности 080801 «Прикладная информатика (по областям)» и другим экономическим специальностям

Издательство
Томского политехнического университета
2013

ББК 681.3
УДК 32.973
З 38

Захарова А.А.

З 38 Информатика и программирование: программные средства реализации информационных процессов: учебник / А.А. Захарова, Е.В. Молнина, Т.Ю. Чернышева; Юргинский технологический институт. 3е изд. – Томск: Изд-во Томского политехнического университета, 2013. – 326 с.

В учебнике изложены основные понятия разделов «Программное обеспечение ПК», «Методы проектирования и программирования», «Методы оценки экономической эффективности вложений в информационные технологии» дисциплины «Информатика и программирование». Подробно рассмотрены функции операционных систем, состав прикладного программного обеспечения с точки зрения пользователя. Приведён теоретический материал, содержащий все дидактические единицы, предъявляемые Государственным образовательным стандартом. Предназначено для студентов высших учебных заведений, обучающихся по специальности 080801 «Прикладная информатика (по областям)» (для бакалавров и магистров направления 230700 «Прикладная информатика») и другим экономическим специальностям, а также для тех, кто интересуется вопросами информатики и информационных технологий.

УДК 32.973
ББК 681.3

Рецензенты

Доктор технических наук, профессор,
зав. кафедрой автоматизированных систем управления ТУСУРа
А.М. Кориков

Доктор технических наук, профессор Кем ГУ
зав. кафедрой автоматизации исследований
и технической кибернетики
В.Я.Карташов

Оглавление

ПРЕДИСЛОВИЕ

Современное общество не может продуктивно существовать без информационных технологий. В постиндустриальном обществе информационные ресурсы страны, региона или отдельной организации рассматриваются как стратегические ресурсы аналогично другим классическим видам производственных ресурсов – материальных, природных, трудовых, финансовых и др.

Как атрибут основной деятельности человечества информационная деятельность появилась довольно давно, задолго до появления персональных компьютеров. Информационная система (ИС) – это одно из средств информационного обмена, взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели. Информационная технология является процессом, состоящим из четко регламентированных правил выполнения операций, действий, этапов разной степени сложности над данными, хранящимися в компьютерах.

В ИС машинная обработка информации предполагает последовательно-параллельное во времени решение вычислительных и функциональных задач.

Обработка экономической информации на вычислительных системах (ВС) производится, как правило, централизованно, на мини- и микро вычислительных машинах – в местах возникновения первичной информации, где организуются автоматизированные рабочие места специалистов той или иной управленческой службы (отдела материально-технического снабжения и сбыта, отдела главного технолога, конструкторского отдела, бухгалтерии, планового отдела и т.п.). Автоматизированное рабочее место (АРМ) специалиста включает персональную электронную вычислительную машину (ПЭВМ), работающую автономно или в вычислительной сети, набор программных средств и информационных массивов для решения функциональных задач.

ИС имеется в любом экономическом объекте, являясь для него естественной составной частью. Взаимосвязь информационных потоков, средств обработки, передачи и хранения данных, а также сотрудников управленческого аппарата, выполняющих операции по переработке данных, и составляет ИС экономического объекта.

Современному поколению информационных систем присущи:

- использование разных платформ в одной ИС;

- повышение интеллектуальности банка данных в следующих направлениях:

1) новые модели знаний, учитывающие не только структуру информации, но и активный характер знаний;

2) средства оперативного анализа информации (OLAP) и средства поддержки принятия решений (DSS);

3) новые формы представления информации, более естественные для человека (мультимедиа, полнотекстовые БД, гипертекстовые БД, средства восприятия и синтеза речи);

- программное обеспечение. Новым является появление и развитие открытой компонентной архитектуры ИС. Программное обеспечение ИС собирается из готовых компонентов. Компонент – это программа, выполняющая какой-либо осмысленный с точки зрения конечного пользователя набор функций и имеющая открытый интерфейс. Компонент может функционировать на разных типах ВС, и связь между компонентами устанавливается не на этапе компиляции, а в реальном масштабе времени. Такой принцип построения позволяет использовать накопленный опыт программистов, ускорять разработку ИС, создавать распределенные ИС;
- архитектура ИС разнообразна в связи с многоплатформенностью.

В последние годы меняется и совершенствуется методика обучения информатике и программированию. Если раньше основное внимание уделялось изучению систем команд электронных вычислительных машин (ЭВМ), то сейчас главным становится освоение наиболее прогрессивных методов разработки алгоритмов решения задач и их проверки на ЭВМ.

В данном учебнике уделено внимание такому функциональному компоненту ИС, как ***программное обеспечение***, представляющему собой комплексы программ для решения типовых задач обработки информации, а также программы, разработанные для конкретной ИС.

В программный инструментарий входят программные продукты, использование которых позволяет достичь поставленную пользователем цель.

К ним относятся программные продукты *общего назначения*:

- текстовые редакторы и издательские системы;
- электронные таблицы;
- системы управления базами данных;
- графические редакторы;
- интегрированные пакеты программ;

- электронные записные книжки;
- телекоммуникационные и сетевые программы и т.п.

К программным средствам *специального назначения* относятся: гиперсреды, информационно-поисковые системы; мультисреды (музыкальные редакторы, среды подготовки презентаций, игры и тренажеры и др.); авторские системы; экспертные системы; системы поддержки принятия решений; системы искусственного интеллекта, базы знаний и пр.

Специалист в области прикладной информатики должен обладать знаниями и навыками применения программных средств *профессионального уровня* в своей профессиональной деятельности. Это проблемно-ориентированные программные продукты таких ведущих фирм, как «1С», «Интеллект-Сервис», «Инфософт», «КОМТЕХ+», «Галактика» и др.

В пособии изложены основные понятия разделов «Программное обеспечение ПК», «Методы проектирования и программирования программного обеспечения», «Методы оценки экономической эффективности вложений в информационные технологии» дисциплины «Информатика и программирование». Подробно рассмотрены функции операционных систем, состав прикладного программного обеспечения с точки зрения пользователя.

При написании учебника использованы не только современные знания, содержащиеся в подобных учебниках, но и обширные сведения глобальной сети Internet. *Курсивом* выделены определения и понятия, на которые читателю необходимо обратить внимание.

Методика изложения материала прошла более чем десятилетнюю апробацию на студентах, обучающихся по специальности 080801 «Прикладная информатика (в экономике)» очной и заочной форм обучения, а также бакалавров направления 230700 «Прикладная информатика».

Контролирующие материалы соответствуют современным требованиям промежуточного и итогового контроля знаний.

В тексте приведён теоретический материал, содержащий все дидактические единицы, предъявляемые Государственным образовательным стандартом по дисциплине «Информатика и программирование».

Предназначено для студентов высших учебных заведений, обучающихся по специальности 080801 «Прикладная информатика (по областям)» (для бакалавров и магистров направления 230700 «Прикладная информатика»), а также для тех, кто интересуется вопросами информатики и информационных технологий.

1. Характеристика информационного процесса и программных средств решения задач на вычислительных системах

1.1. Общая характеристика процессов сбора, передачи, обработки и хранения информации

Информационный процесс – процесс восприятия, сбора, обработки и передачи, накопления и хранения информации. Информационный процесс может состояться только при наличии информационной системы. Информационные системы и обеспечивают все эти процессы при решении задач из любой области.

Любая система, как социально-экономическая, так и система живой и неживой природы, действует в постоянной взаимосвязи с внешней средой – системами более высокого или более низкого уровней. Взаимосвязь осуществляется посредством информации, которая по потокам прямой связи передает цель функционирования, различные команды управления от системы более высокого уровня к системам низкого звена, а по потокам обратной связи – все сведения, необходимые для регулирования функционального процесса.

Восприятие информации – процесс преобразования сведений, поступающих в техническую систему из внешнего мира, в форму, пригодную для дальнейшего использования. Благодаря восприятию информации обеспечивается связь системы с внешней средой. Система восприятия информации может представлять собой довольно сложный комплекс программных и технических средств. Для развития систем восприятия можно выделить несколько этапов переработки поступающей информации: предварительная обработка для приведения входных данных к стандартному для данной системы виду, выделение в поступающей информации семантически и прагматически значимых информационных единиц, распознавание объектов и ситуаций, коррекция внутренней модели мира. С точки зрения информационной системы в целом система восприятия осуществляет первичную обработку собираемой извне информации. В свою очередь, для системы восприятия первичную обработку информации производит система сбора информации.

Сбор информации. Из изложенного выше легко сделать вывод, что система сбора информации может представлять собой сложный программно-аппаратный комплекс. Как правило, современные системы

сбора информации не только обеспечивают кодирование информации и ее ввод в вычислительную машину (ВМ), но и выполняют предварительную (первичную) обработку этой информации. Сбор информации – это процесс получения информации из внешнего мира и приведение ее к виду, стандартному для данной информационной системы. Обмен информацией между воспринимающей информацию системой и окружающей средой осуществляется посредством сигналов. Сигнал можно определить как средство перенесения информации в пространстве и времени. В качестве носителя сигнала могут выступать звук, свет, электрический ток, магнитное поле и т.д.

Сбор информации, как правило, сопровождается ее регистрацией, т.е. фиксацией информации на материальном носителе (документе или машинном носителе).

Передача информации может осуществляться различными способами: с помощью курьера, по почте, с помощью транспортных средств или по каналам связи.

Дистанционная передача по каналам связи сокращает время передачи данных. Для ее осуществления необходимы специальные технические средства. Некоторые технические средства сбора и регистрации, собирая автоматически информацию с датчиков, установленных на рабочих местах, передают ее в ВМ.

В современных условиях большое распространение получила распределенная обработка информации, при этом сети передачи данных превращаются в информационно-вычислительные сети. Информационно-вычислительные сети представляют наиболее динамичную и эффективную отрасль автоматизированной технологии процессов ввода, передачи, обработки и выдачи информации.

Обработка информации. В современных развитых информационных системах машинная обработка информации предполагает последовательно-параллельное во времени решение вычислительных задач. Это возможно при наличии определенной организации вычислительного процесса.

Для управления любым экономическим объектом необходимо располагать и манипулировать определенными сведениями о его фактическом и желаемом состояниях. Совокупность информации конкретного экономического объекта образует информационную систему, в которой различают входящую, исходящую, внутреннюю и внешнюю информацию.

Информацию, поступающую в информационную систему, называют входящей. Информационная система, обрабатывая входящие данные, порождает новую – результатную информацию (сводную).

Передаваемая за пределы данной информационной системы информация называется исходящей. Если сведения поступают в информационную систему от объектов управления, то такая информация будет входящей внутренней, если из внешнего мира (например, для предприятий из министерства, от других организаций), информация называется входящей внешней. Аналогичным образом подразделяются и выходящие сведения. Входящую внутреннюю информацию в условиях предприятия называют первичной. Она возникает в процессе первичного учета хозяйственных операций – измерение и регистрация данных – в ходе производственной деятельности объекта управления.

По критерию соответствия отражаемым явлениям экономическая информация может быть отнесена к достоверной и недостоверной. К этому классификационному признаку примыкает оценка своевременности и несвоевременности информационного отображения производственных и хозяйственных операций, получения исходной и результатной информации в установленные сроки.

По отношению к процессам обработки и хранения различают следующие виды экономической информации: исходную, производственную, хранимую без обработки, результатную, промежуточную.

С точки зрения отражаемых функций управления экономическая информация подразделяется на плановую, прогнозную, нормативную, конструкторско-технологическую, учетную, финансовую.

Обработка экономической информации на ВМ производится, как правило, централизованно, на мини- и микро ВМ – в местах возникновения первичной информации, где организуются автоматизированные рабочие места специалистов той или иной управленческой службы (отдела материально-технического снабжения и сбыта, отдела главного технолога, конструкторского отдела, бухгалтерии, планового отдела и т.п.). Автоматизированное рабочее место (АРМ) специалиста включает персональную электронную ВМ (ПЭВМ), работающую автономно или в вычислительной сети, набор программных средств и информационных массивов для решения функциональных задач.

Рассмотренные технологические процессы и режимы работы пользователей в системе «человек–машина» особенно четко проявляются при интегрированной обработке информации, которая характерна для современного автоматизированного решения задач в многоуровневых информационных системах.

Хранение и накопление информации вызвано многократным ее использованием, применением постоянной информации, необходимостью комплектации первичных данных до их обработки.

Хранение информации осуществляется на машинных носителях в виде информационных массивов, где данные располагаются по установленному в процессе проектирования группировочному признаку.

Поиск данных – это выборка нужных данных из хранимой информации, включая поиск информации, подлежащей корректировке или замене запроса на нужную информацию.

1.2. Информационные системы

В работе информационной системы на равных участвуют как технические и программные средства, так и человек.

Информационная система (ИС) – это взаимосвязанная совокупность средств, методов и персонала, участвующих в обработке данных.

Как упоминалось выше, ИС должна обеспечивать прием поступающей из источника информации, ее преобразование (обработку), хранение и передачу результатов преобразования.

Информационная система должна содержать в себе:

- *информационные ресурсы* – вся та информация, которая циркулирует в системе (массивы накопленной информации, всевозможные архивы на разных носителях и т.д., а также методики, инструкции и программы, регламентирующие процессы прохождения информации в системе, ее обработки, хранения, представления);
- *материальные ресурсы,* в том числе носители информации, технические средства сбора, передачи, обработки информации;
- *каналы циркулирования информации;*
- *определенный контингент работников.*

Внедрение ИС способствует:

- получению более рациональных вариантов решения управленческих задач за счет внедрения математических методов, интеллектуальных систем и т.д.;
- освобождению работников от рутинной работы за счет ее автоматизации;
- обеспечению достоверности информации;
- замене бумажных носителей данных на магнитные (или оптические) диски или магнитные ленты, что приводит к более

рациональной организации переработки информации на компьютере и снижению объемов документов на бумаге;

- совершенствованию структуры потоков информации и системы документооборота в фирме;
- уменьшению затрат на производство продуктов и услуг;
- отысканию новых рыночных ниш.

Информационная система экономического объекта – это совокупность средств и методов, обеспечивающих реализацию всего комплекса операций по обеспечению процесса управления необходимой информацией.

Информационная система имеется в любом экономическом объекте, являясь для него естественной составной частью. Взаимосвязь информационных потоков, средств обработки, передачи и хранения данных, а также сотрудников управленческого аппарата, выполняющих операции по переработке данных, и составляет ИС экономического объекта.

При разработке и организации ИС необходим анализ основных её компонентов, которыми являются:

1. Функциональные компоненты. Под функциональными компонентами понимается система функций управления – полный набор взаимоувязанных во времени и пространстве работ по управлению, необходимых для достижения поставленных перед предприятием целей. Декомпозиция ИС по функциональному признаку включает в себя выделение ее отдельных частей, называемых функциональными подсистемами. Функциональный признак определяет назначение подсистемы, т.е. то, для какой области деятельности она предназначена и какие основные цели, задачи и функции она выполняет. Функциональные подсистемы в значительной степени зависят от предметной области ИС. Ряд функциональных подсистем имеют одно и то же наименование (например, бухгалтерский учет, отчетность), но внутреннее содержание для различных объектов значительно отличается друг от друга.

2. Компоненты системы обработки данных. Основная функция систем обработки данных – реализация типовых операций обработки данных:

- сбор, регистрация и перенос информации на машинные носители;
- передача информации в места ее хранения и обработки;
- обработка информации и т.д.

Принято выделять:

а) *информационное обеспечение* – совокупность методов и средств по размещению и организации информации: системы классификации и кодирования информации; унифицированные системы документации, главная цель которых – обеспечение сопоставимости показателей различных сфер общественного производства (чтобы уменьшить дублирующие показатели, чтобы не было неиспользуемых показателей и т.д.); схемы информационных потоков; методологии построения баз данных;

б) *техническое обеспечение* (аппаратное обеспечение) – комплекс технических средств, предназначенных для работы ИС (компьютеры, устройства сбора, обработки, передачи и вывода информации, устройства передачи данных и т.д.);

в) *математическое обеспечение* – средства моделирования процессов управления, типовые задачи управления, методы математического программирования, математической статистики, теории массового обслуживания и др., т.е. совокупность математических методов и моделей для реализации целей и задач ИС;

г) *программное обеспечение* – комплексы программ для решения типовых задач обработки информации, а также программы, разработанные для конкретной ИС. К программному инструментарию относятся программные продукты, использование которых позволяет достичь поставленную пользователем цель. Это, например, программные продукты общего назначения: текстовые редакторы, электронные таблицы, системы управления базами данных, электронные записные книжки и т.п.;

д) *организационное обеспечение* – анализ существующей системы управления организацией, где будет использоваться ИС, и выявление задач, подлежащих автоматизации; подготовка задач к решению на компьютере, включая техническое задание на проектирование ИС и технико-экономическое обоснование ее эффективности;

е) *правовое обеспечение* – совокупность правовых норм, определяющих создание и функционирование ИС: статус ИС; права, обязанности и ответственность персонала; порядок создания и использования информации.

3. Организационные компоненты – совокупность методов и средств, позволяющих усовершенствовать организационную структуру объектов и управленческие функции, выполняемые структурными подразделениями; определить штатное расписание и численный состав каждого структурного подразделения; разработать должностные инструкции персоналу управления в условиях функционирования системы обработки данных.

Одним из базовых элементов компьютерной ИС является информационная технология. Информационные технологии определяют способы, методы и средства сбора, регистрации, передачи, хранения, обработки и выдачи информации в информационной системе.

Информационная технология (ИТ) – процесс, использующий совокупность средств и методов обработки и передачи первичной информации для получения информации нового качества о состоянии объекта, процесса или явления.

1.3. Решение функциональных и вычислительных задач на вычислительных системах

Решение любой задачи должно начинаться с точной её постановки, т.е. четкого выделения того, что требуется, и того, что дано (рис. 1.1).

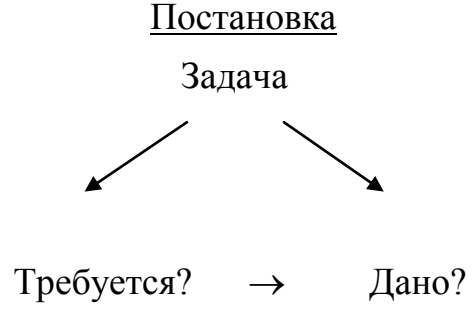


Рис.1.1. Постановка задачи

Следующий этап – определение способа решения задачи. Способ решения – это набор действий, позволяющих получить требуемое из исходных данных (рис. 1.2).

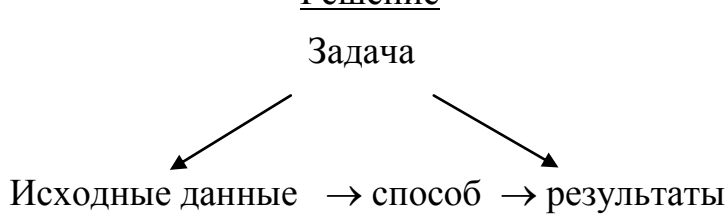


Рис.1.2. Последовательность решения задачи

Результат правильный, если он отвечает требованиям. Результат неправильный, если он не соответствует требованиям. Однако при отсутствии четких требований невозможно однозначно судить о правильности или неправильности результатов.

Главным и наиболее значимым современным техническим средством (или совокупностью, комплексом технических средств) решения функциональных и вычислительных задач, безусловно, является так называемая вычислительная машина (или вычислительная система).

В словосочетании «вычислительная машина» термин *«машина»* сохраняет свое традиционное толкование как некоторое техническое устройство, облегчающее трудовую деятельность человека (в данном случае – умственную деятельность) по обеспечению его жизненных потребностей. Прилагательное *«вычислительная»* уже давно переросло свое дословное толкование и к настоящему времени под обработкой информации понимают не только «вычисления» (в узком смысле этого слова), а гораздо более широкий спектр операций с (или над) информацией, таких как сбор, ввод, формализация, фильтрация, сортировка, хранение, накопление, защита, преобразование, выдача информации (и этот список типовых операций может быть продолжен).

Поэтому современная трактовка указанного прилагательного предполагает именно такое его широкое толкование. Определение понятия «вычислительная машина, представленное в специальном стандарте терминов и определений в области систем обработки информации (ГОСТ 15971-90): «*вычислительная машина* – это комплекс технических средств, создающих возможность автоматизации обработки информации по заданному *алгоритму* и получения результата в необходимой форме».

Но в современной литературе приоритет отдаётся термину «*вычислительная система*», который появился в результате появления новых технических решений: разделение процессов обработки информации и ее ввода-вывода, множественный доступ и коллективное использование вычислительных ресурсов в пространстве и во времени; появление сложных режимов работы ВМ (многопользовательская и многопрограммная обработка); возможности построения параллельных ветвей в вычислениях, что не предусматривалось классической структурой ВМ; комплексирование элементов или модулей в систему и др.

Под *вычислительной системой* (ВС) будем понимать совокупность взаимосвязанных и взаимодействующих процессоров или ВМ, периферийного оборудования и программного обеспечения, предназначенную для сбора, хранения, обработки и распределения информации. Отличительной особенностью ВС по отношению к ВМ является наличие в них нескольких вычислителей, реализующих параллельную обработку. При решении задач на вычислительных

системах (ВС) постановка задач предполагает представление требуемого и исходного в виде данных. Способы решения задач на ВС в такой постановке должны быть представлены соответствующими алгоритмами и программами обработки данных.

При отсутствии готовых программ для решения задач возникает проблема создания соответствующих алгоритмов и программ. В любом случае необходимо подобрать и определить способы, методы и средства для решения поставленных задач (рис. 1.3).

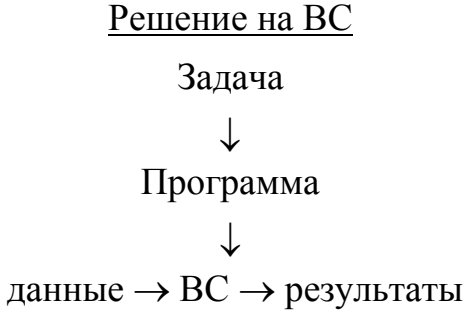


Рис.1.3. Последовательность решения задачи на ВС

Систематический подход к составлению программ предполагает в качестве первого этапа составление *спецификаций* – описаний форм ввода и хранения данных в ВС, а также получения и вывода результатов. Эти спецификации в дальнейшем будут использоваться для оценки правильности созданных программ. Для диалоговых программ в роли таких спецификаций выступают сценарии диалога – полные описания результатов и правил работы с ВС при решении поставленных задач. Только после создания таких спецификаций должны составляться соответствующие им алгоритмы и программы (рис. 1.4).

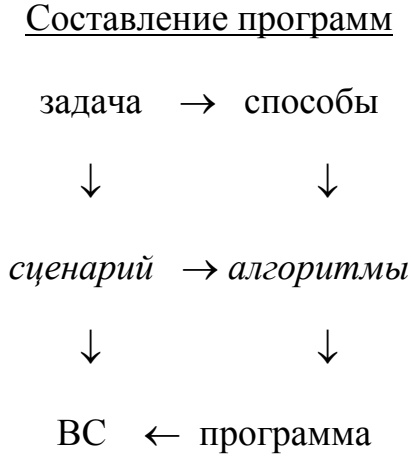


Рис.1.4. Последовательность составления алгоритмов и программ

Приведенная схема представляет основной принцип систематических методов составления алгоритмов и программ для решения различных прикладных задач – экономических, математических, физических, инженерных и т. д.

Особенностью систематических методов является возможность полного устранения ошибок из алгоритмов и программ. При этом подходе программы сверяются с описаниями алгоритмов, а алгоритмы – с описаниями сценариев и методов решения. Такой систематический подход к составлению алгоритмов и программ может применяться к решению на ВС любых прикладных задач с использованием самых различных языков программирования – Бейсик, Паскаль, Delphi, Си ++.

Для решения поставленных задач на ВМ предназначены *программные среды*. Как говорилось выше, при отсутствии готовых программ возникает проблема создания соответствующих алгоритмов и программ. Средства создания программ называются *средами программирования, инструментарием программирования* или *оболочками.*

1.4. Аппаратные и программные средства

В процессе проектирования логической структуры системы приходится иметь дело с целым рядом факторов технического характера для того, чтобы обеспечить правильную разработку безошибочного проекта, удовлетворяющего всем системным требованиям. Эти факты могут быть как качественными, так и количественными. *Качественные аспекты проектирования* включают модульность, гибкость и надежность системы. *Количественные факторы* связаны с описанием принципов функционирования системы с точки зрения обеспечиваемого времени реакции и производительности (факторы, влияющие на надежность системы, рассматриваются как качественные). Исторически сложилось так, что две составные части любой системы – аппаратные и программные средства – имеют совершенно различные характеристики и разрабатываются совсем непохожими методами. Технологический взрыв в корне изменяет это положение дел. Внутренне присущая аппаратным средствам модульность навела специалистов по системам на мысль о новом подходе к созданию программных средств, которые, в свою очередь, позволяют проектировать и описывать и программы, и аппаратные средства с использованием программных методов. В этом смысле аппаратные и программные средства все более сближаются. В определенных сферах применения современные методы разработки программного обеспечения обещают «подорвать авторитет»

исторически сложившейся концепции жизненного цикла. Более того, всепроникающее появление персональных вычислительных машин делает процесс разработки средств программного обеспечения доступным практически каждому.

1.5. Программно-аппаратный принцип построения алгоритма

Любой модуль, подсоединенный к другому с помощью кабеля и разъема, можно без ущерба для системы в целом отсоединить и заменить совместимыми по разъему модулем с совершенно отличной внутренней структурой. Поскольку совместимость на уровне разъемов обеспечивает рост объемов производства компонентов систем, становится возможным принятие альтернативных решений о покупке отдельных модулей или изготовлении их собственными силами. Наконец, при исчезновении питания или блокировании функции, выполняемой конкретным модулем, в случае аппаратных средств оказывается возможной их реконфигурация в процессе эксплуатации.

Другая отличительная *особенность аппаратных средств* – это возможность параллельного функционирования двух и более модулей системы, причем любой модуль может сам состоять из параллельно работающих частей. Концепции совместимости на уровне разъемов и параллелизма настолько естественны в мире технических средств, что уже совершенно не допускают какого-либо иного образа мышления в этом плане. Однако в области программного обеспечения исторически ситуация сложилась по-другому. Получившие широкое распространение традиционные языки программирования не содержат ни средств обеспечения совместимости модулей, ни средств реализации параллелизма. До недавнего времени немногие языки, которые действительно обладали такими возможностями, либо не имели широкого распространения, либо не были доступны широкому кругу пользователей. Различные организации, осознавшие необходимость подобных функций, пытались нестандартными способами так расширить стандартные языки, чтобы они удовлетворяли указанным требованиям. В итоге широкое распространение получили нестандартные узко-специализированные методы разработки программного обеспечения.

Провести однозначную грань между программной, программно-аппаратной и аппаратной реализацией достаточно затруднительно. Критерием выбора будем считать:

а) программную реализацию алгоритма, когда алгоритм реализуется таким образом, при котором его выполнение базируется на

виртуальном программном уровне, представленном, например, операционной системой, и является переносимой на аналогичный уровень абстракции среды выполнения без изменений;

б) программно-аппаратную реализацию алгоритма, когда учитываются архитектурные и другие особенности аппаратуры, например система команд микропроцессоров, их взаимосвязи и их количество, подчиненность и т.д., благодаря чему получается выигрыш в производительности или в других характеристиках конкретной реализации алгоритма;

в) аппаратную реализацию алгоритма, когда алгоритм выполняется одним или несколькими аппаратными компонентами, вся алгоритмическая часть заключена внутри этих компонентов, и для изменения алгоритма требуется замена или модификация этого компонента и/или его взаимосвязей с другими компонентами.

Программно-аппаратная реализация предполагает, что средства разработки ПС имеют средства взаимодействия непосредственно с аппаратными компонентами системы: памятью, программными и аппаратными прерываниями, портами ввода–вывода и другими компонентами аппаратуры.

Классическим способом аппаратно-программной реализации является подход разделения среды разработки и моделирования от среды целевого исполнения и окончательной отладки. Таким образом, имеются:

а) аппаратно-программные средства разработки ПС и моделирования их работы для определенной архитектуры аппаратуры;

б) аппаратно-программные средства загрузки разработанных ПС в целевую аппаратную среду и отладки ПС на целевой аппаратуре;

в) целевая аппаратура, на которой выполняются разработанные ПС.

Для программно-аппаратной реализации чаще всего используются следующие языки программирования:

а) язык Ассемблер, для каждого микропроцессора свой;

б) язык С, как правило, свой диалект языка для каждого микропроцессора;

в) язык Модула-2, разрядность скалярных типов зависит от разрядности процессора;

г) язык Ада, обладает максимально возможной переносимостью;

д) другие языки, например RTL-2, среднее между С и Модула-2.

Средства разработки на ассемблере и языке С, как правило, распространяются производителем микропроцессоров и могут значительно отличаться друг от друга.

1.6. Средства создания программ

Инструментами программиста являются тоже программы. Именно с помощью программ создают новые программы. В самом общем случае для создания программы на выбранном языке программирования нужно иметь несколько компонентов:

1. *Текстовый редактор.* Так как текст программы записывается с помощью ключевых слов, обычно происходящих от слов английского языка, и набора стандартных символов для записи всевозможных операций, то формировать этот текст можно в любом редакторе, получая в итоге текстовый файл с *исходным текстом* программы. Лучше использовать специализированные редакторы, которые ориентированы на конкретный язык программирования и позволяют в процессе ввода текста выделять ключевые слова и идентификаторы разными цветами и шрифтами. Подобные редакторы созданы для всех популярных языков и дополнительно могут автоматически проверять правильность синтаксиса программы непосредственно во время ее ввода.

2. Исходный текст с помощью *программы-компилятора* переводится в машинный код. Если обнаружены синтаксические ошибки, то результирующий код создан не будет. На этом этапе уже возможно получение готовой программы, но чаще всего в ней не хватает некоторых компонентов, поэтому компилятор обычно выдает промежуточный *объектный код* (двоичный файл, стандартное расширение OBJ). Объектный код (или объектный модуль) – это программная единица, которая является результатом ассемблирования или компиляции и пригодна для обработки её редактором связей.

3. Исходный текст большой программы состоит, как правило, из нескольких *модулей* (файлов с исходными текстами), потому что хранить все тексты в одном файле неудобно – в них сложно ориентироваться. Каждый модуль компилируется в отдельные файлы с объектным кодом, которые затем надо объединить в одно целое. Кроме того, к ним надо добавить машинный код подпрограмм, реализующих различные стандартные функции (например, вычисляющих математические функции sin или ln). Такие функции содержатся в *библиотеках* (файлах со стандартным расширением LIB), которые поставляются вместе с компилятором.

Девяносто пять процентов большой программы состоит из небольших стандартных подпрограмм (*иначе процедур)*. Из многих тысяч написанных во всем мире программ редкая программа обходится без стандартных процедур, управляющих вводом данных с клавиатуры

или выводе информации на экран. Зачем много раз писать одно и то же, когда существуют библиотечные файлы, из которых извлекают стандартные блоки и используют без какой-либо переделки.

Год от года такие библиотеки наращиваются, становятся всё крупнее и крупнее. Поэтому с каждым годом растет производительность труда программистов. Пятнадцать лет назад средний размер компьютерной программы составлял 40–50 кбайт. Десять лет назад он равнялся сотням кбайт. Сегодня исполняемые файлы занимают мегабайты. Без использования библиотек такие программы готовились бы около двадцати лет.

Сгенерированный код модулей и подключенные к нему стандартные функции надо не просто объединить в одно целое, а выполнить такое объединение с учетом требований операционной системы, то есть получить на выходе программу, отвечающую определенному формату. Объектный код обрабатывается специальной программой – *редактором связей* или *сборщиком,* который выполняет связывание объектных модулей и машинного кода стандартных функций, находя их в библиотеках, и формирует на выходе работоспособное приложение – *исполнимый код* для конкретной платформы. Если по каким-то причинам один из объектных модулей или нужная библиотека не обнаружены (например, неправильно указан каталог с библиотекой), то сборщик сообщает об ошибке и готовой программы не получается.

Исполнимый код – это завершенная программа, которую можно запустить на любом компьютере, где установлена операционная система, для которой эта программа создавалась. Как правило, итоговый файл имеет расширение ~.EXE или ~.COM.

Итак, для создания программы нужны:

- текстовый редактор;
- компилятор;
- редактор связей;
- библиотеки функций;
- отладчик.

Как правило, в стандартную поставку входят как минимум четыре последних компонента, но хорошая *интегрированная система* включает в себя и специализированный текстовый редактор, причем почти все этапы создания программы в ней автоматизированы: после того как исходный текст введен, его компиляция и сборка выполняются одним нажатием клавиши. Это очень удобно, так как не требует ручной настройки множества параметров запуска компилятора и редактора

связей, указывания им нужных файлов вручную и т. д. Процесс компиляции обычно демонстрируется на экране: показывается, сколько строк исходного текста откомпилировано, или выдаются сообщения о найденных ошибках.

Отладчик позволяет анализировать работу программы во время ее выполнения. С его помощью можно последовательно выполнять отдельные операторы исходного текста по *шагам,* наблюдая при этом, как меняются значения различных переменных. Без отладчика разработать крупное приложение очень сложно.

В качестве примера можно привести VBA как систему объектно–ориентированного программирования. Для приложений MS Office фирма Microsoft использует единый и мощный язык программирования Visual Basic, приспосабливая его к специфике приложений. Эта версия языка носит название *Visual Basic for Applications (Visual Basic для приложений)* или *VBA.*

Решение прикладной задачи с помощью VB выполняется с использованием *проекта.* Проекты содержат *модули,* а модули включают *процедуры* и *функции.* В Visual Basic программный код оформляется в виде процедур и функций. Разработанные отдельные функции или процедуры можно накапливать в библиотеках и в дальнейшем использовать по мере необходимости. Главное окно редактора обычно занимает весь экран (рис. 1.5).

В окне имеются строка заголовка, меню и панель инструментов. В строке заголовка выводится имя текущей рабочей книги. В главном окне размещаются все другие окна редактора. В окне проекта отображается список проектов всех открытых рабочих книг и иерархическая структура каждого проекта.

Проект – эта та часть программы, которая видима на экране при ее создании. На рисунке 1.5 приведен пример проекта *VBA* для приложения Excel. Как видно из рисунка, проект VBA имеет иерархическую структуру и включает объекты Excel, формы, стандартные модули и модули класса. Объектами Excel, входящими в проект, являются рабочие книги (WorkBooks), рабочие листы (Worksheets) диаграммы (Charts). С каждым из объектов связан специальный модуль, в который может быть помещен программный код, выполняющий определенные действия. Весь проект представляет собой один файл – рабочую книгу, и сохраняется вместе с ней.

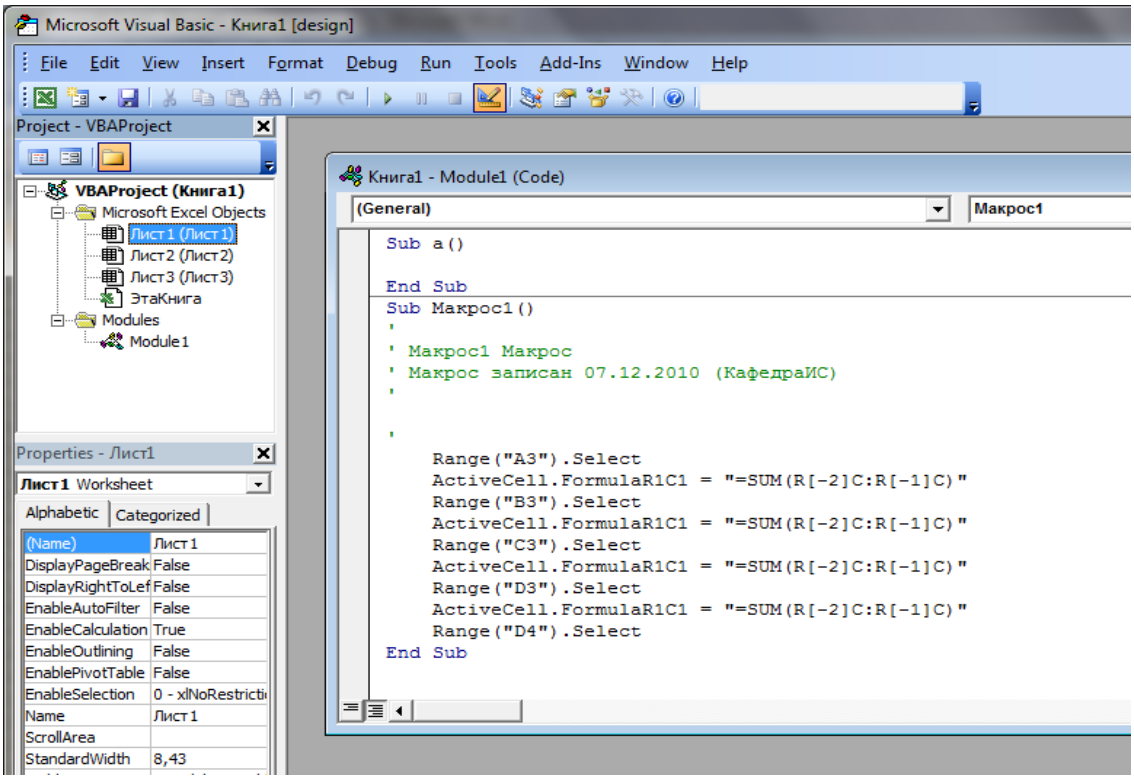


Рис. 1.5. Главное окно редактора VBA

1.7. Методы проектирования программ

Вычислительная машина исполняет программу в машинных кодах. Составляют программу люди на удобном для себя языке.

Программа – это последовательность команд компьютера, приводящая к решению задачи.

Приложение – это программная реализация на компьютере решения задачи.

Программное обеспечение (ПО) – это программные продукты и техническая документация к ним.

Программный продукт (ПП) – это комплекс взаимосвязанных программ, предназначенный для реализации определенной задачи массового спроса.

Различают языки:

- низкого уровня (машинно-ориентированные, например, Ассемблер),
- высокого уровня (не зависят от типа ВМ).

Разные типы процессоров имеют разные наборы команд. Если язык программирования ориентирован на конкретный тип процессора и учитывает его особенности, то он называется *языком программирования низкого уровня.* В данном случае «низкий уровень» не значит «плохой».

Имеется в виду, что операторы языка близки к машинному коду и ориентированы на конкретные команды процессора.

Языком самого низкого уровня является *язык Ассемблер,* который просто представляет каждую команду машинного кода, но не в виде чисел, а с помощью символьных условных обозначений, называемых *мнемониками.* Однозначное преобразование одной машинной инструкции в одну команду ассемблера называется *транслитерацией.* Так как наборы инструкций для каждого модели процессора отличаются, конкретной компьютерной архитектуре соответствует свой язык Ассемблера, и написанная на нем программа может быть использована только в этой среде.

С помощью языков низкого уровня создаются очень эффективные и компактные программы, так как разработчик получает доступ ко всем возможностям процессора. С другой стороны, при этом требуется очень хорошо понимать устройство компьютера, затрудняется отладка больших приложений, а результирующая программа не может быть перенесена на компьютер с другим типом процессора.

Подобные языки обычно применяют для написания небольших системных приложений, драйверов устройств, модулей стыковки с нестандартным оборудованием, когда важнейшими требованиями становятся компактность, быстродействие и возможность прямого доступа к аппаратным ресурсам. В некоторых областях, например в машинной графике, на языке ассемблера пишутся библиотеки, эффективно реализующие требующие интенсивных вычислений алгоритмы обработки изображений.

Языки программирования высокого уровня значительно ближе и понятнее человеку, нежели компьютеру. Особенности конкретных компьютерных архитектур в них не учитываются, поэтому создаваемые программы на уровне исходных текстов легко переносимы на другие платформы, для которых создан транслятор этого языка. Разрабатывать программы на языках высокого уровня с помощью понятных и мощных команд значительно проще, а ошибок при создании программ допускается гораздо меньше.

Языки высокого уровня по подходу к реализации задачи бывают:

- процедурно-ориентированные (Паскаль),
- проблемно-ориентированные (MathCAD),
- объектно-ориентированные (C++).

Основные методы проектирования программных продуктов представлены на рисунке 1.6. Проектирование алгоритмов и программ – наиболее ответственный этап жизненного цикла ПП.

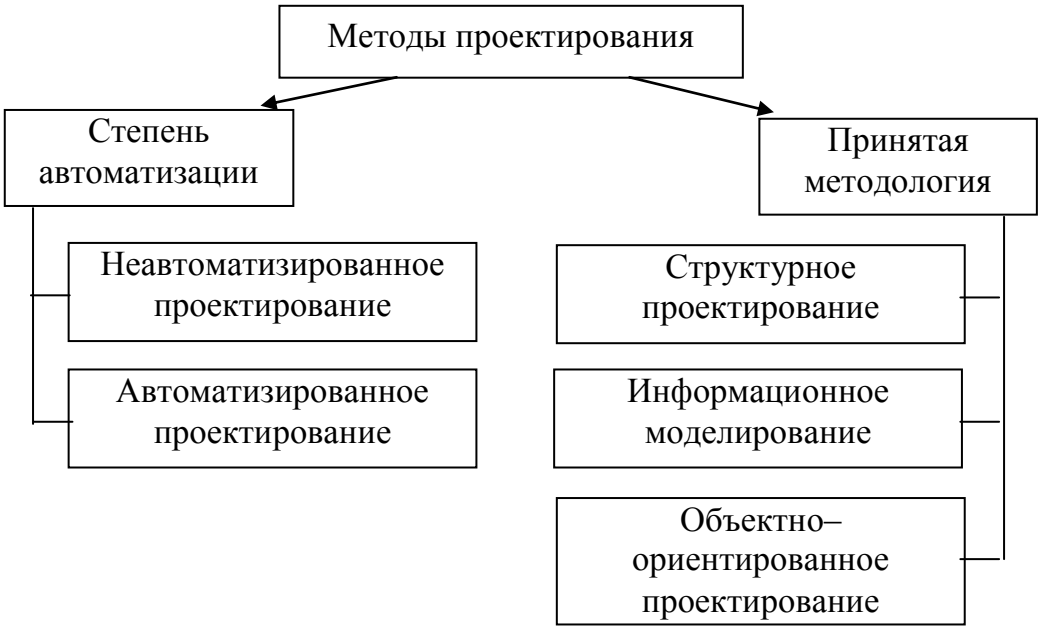


Рис. 1.6. Классификация методов проектирования программных продуктов

По степени автоматизации методы проектирования алгоритмов и программ делятся на неавтоматизированное и автоматизированное проектирование. Традиционное неавтоматизированное проектирование алгоритмов и программ используется при разработке небольших по трудоемкости и структурной сложности ПП, не требующих большого числа разработчиков. ПП имеют прикладной характер. Автоматизированное проектирование алгоритмов и программ возникло с необходимостью уменьшить затраты на проектные работы, сократить сроки их выполнения, создать типовые «заготовки», многократно тиражируемые для различных разработок, координации работ большого коллектива разработчиков.

Автоматизированное проектирование используется в крупных фирмах при разработке определенного класса ПП большого коллектива разработчиков.

Методология – это подходы к проектированию систем.

По используемой методологии принято следующее деление на методы:

- структурное проектирование;
- информационное моделирование;
- объектно-ориентированное проектирование.

Структурное проектирование – это последовательная декомпозиция, целенаправленное разбиение на отдельные составляющие.

Структурное проектирование включает в себя:

- нисходящее проектирование («сверху вниз»);
- модульное программирование;
- структурное программирование (кодирование).

Метод *нисходящего проектирования* предполагает последовательное разложение общей функции обработки данных на простые функциональные элементы («сверху вниз»).

Метод предполагает последовательное разложение функции обработки данных на простые функциональные элементы. В результате строится функциональная структура алгоритма (ФСА) приложения, в которой отражаются:

- цели предметной области (*цель-подцель*);
- состав приложений (задач обработки), обеспечивающих реализацию поставленных целей;
- характер взаимосвязи приложений с их основными характеристиками;
- функции обработки данных;

В результате строится иерархическая схема, которая отражает состав и взаимоподчиненность отдельных функций. Она носит название *функциональная структура алгоритма (ФСА)* приложения.

Подобная структура отражает состав и взаимосвязь функций обработки информации для реализации приложений, не раскрывая логику выполнения каждой отдельной функции.

Разложение должно носить строго функциональный характер, т.е. отдельный элемент ФСА описывает законченную содержательную функцию обработки информации, которая предполагает определенный способ реализации на программном уровне.

Функции ввода/вывода информации отделяют от функций вычислительной или логической обработки данных.

Структурное программирование основано на модульной структуре программного продукта и базовых алгоритмических структурах.

Модуль представляет собой совокупность логически связанных элементов, предназначенных для использования другими модулями и программами.

Модули предназначены для хранения готовых программ. Модуль сам по себе не является выполняемой программой – его объекты

используются другими программными единицами (процедурами, функциями).

Модуль имеет:

- один вход и один выход – на входе программный модуль получает определенный набор исходных данных, выполняет обработку данных и возвращает один набор результатных данных, т.е. реализует принцип IPO (Input–Process–Output) – вход–процесс–выход;
- функциональную завершенность – модуль выполняет перечень операций для реализации каждой отдельной функции в полном составе, достаточных для завершения начатой обработки;
- логическую независимость – результат работы модуля зависит только от исходных данных, и не зависит от работы других модулей;
- слабые информационные связи с другими программными модулями – обмен информации между модулями должен быть по возможности минимизирован;
- обозримый по размеру и сложности программный элемент.

Каждый модуль состоит из 2 частей: *спецификации* – правила использования модуля, и *тела* – способа реализации процесса обработки.

В основе модульного программирования ПП лежат следующие принципы:

- определение состава и подчиненность функций,
- определение набора программных модулей, реализующих эти функции.

При составлении алгоритма необходимо учитывать:

- каждый модуль вызывается на выполнение вышестоящим модулем и, закончив работу, возвращает управление вызвавшему его модулю;
- принятие основных решений в алгоритме выносится на максимально «высокий» по иерархии уровень;
- для использования одной и той же функции в разных местах алгоритма создается один модуль, который вызывается на выполнение по мере необходимости

В результате детализации алгоритма создается *функционально-модульная схема* алгоритма приложения, которая является основой для программирования (рис.1.7).

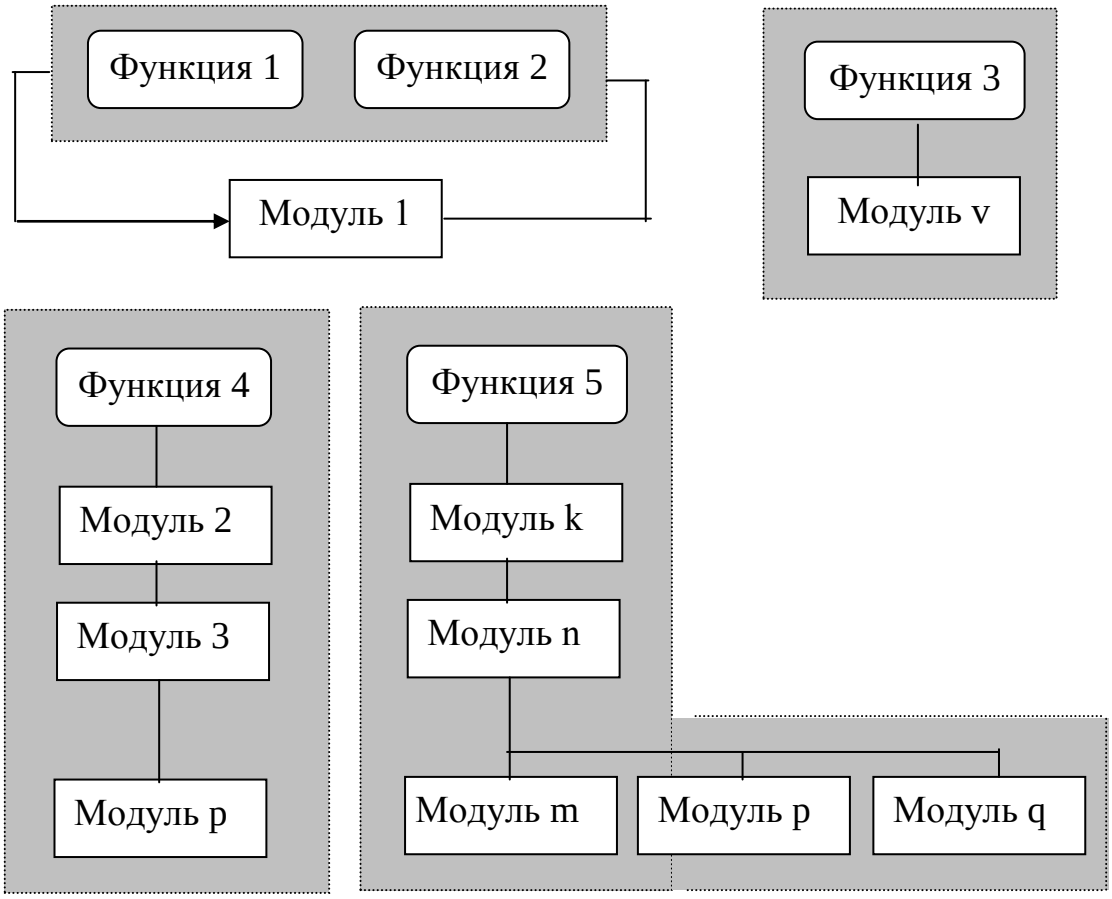


Рис.1.7. Функционально–модульная схема алгоритма приложения

Состав и вид программных модулей, их назначение и характер использования в программе в значительной степени определяются инструментальными средствами. Например, применительно к средствам систем управления базами данных (СУБД) отдельными модулями могут быть:

- экранные формы ввода и редактирования информации базы данных;
- отчеты генератора отчетов;
- макросы;
- стандартные процедуры обработки информации;
- меню, обеспечивающее выбор функции обработки и др.

Информационное моделирование – построение комплекса взаимосвязанных моделей данных. В основе информационного моделирования лежит положение об определяющей роли данных при проектировании алгоритмов и программ.

Для *информационного моделирования предметной области* большую значимость имеют информационные модели и структуры

данных, в основе которого положение об определяющей роли данных при проектировании алгоритмов и программ. Подход появился в условиях развития программных средств организации хранения и обработки данных – СУБД. Первоначально строятся *информационные модели* различных уровней представления:

- информационно-логическая модель, не зависящая от средств программной реализации хранения и обработки данных, отражающая интегрированные структуры данных предметной области;
- даталогические модели, ориентированные на среду хранения и обработки данных.

Даталогические модели имеют логический и физический уровни представления. *Физический уровень* соответствует организации хранения данных в памяти компьютера. *Логический уровень* данных применительно к СУБД реализован в виде:

- концептуальной модели базы данных – интегрированные структуры данных под управлением СУБД;
- внешних моделей данных – подмножество структур данных для реализации приложений.

Выбор средств реализации базы данных определяет вид даталогических моделей и, следовательно, алгоритмы преобразования данных. В большинстве случаев используется реляционное представление данных базы данных и соответствующие реляционные языки для программирования (манипулирования) обработки данных СУБД и реализации алгоритмов обработки. Данный подход использован во многих CASE–технологиях.

Объектно-ориентированный подход (ООП) основан на следующих принципах:

- выделении классов объектов;
- установлении свойств объектов и методов их обработки;
- создании иерархии классов, наследовании свойств объектов и методов их обработки.

Каждый объект объединяет данные и программу обработки этих данных и относится к определенному классу.

Основная цель ООП – преодолеть следующие недостатки проектирования «сверху вниз»:

- недостаточное внимание к структурам данных,
- слабая связь структур данных с процессами их обработки.

Кроме того, ООП позволяет резко сократить объем и трудоемкость подготовки программ, имеющих дело с множеством связанных друг с другом объектов.

Объектно-ориентированный анализ – это процесс выявления объектов, определение свойств и методов обработки объектов, установление их взаимосвязей.

ООП – процесс объектной декомпозиции и представления с использованием моделей данных проектируемой системы на логическом и физическом уровнях.

1.8. Стили программирования

В настоящее время к основным стилям программирования относят:

- структурное;
- модульное;
- объектно-ориентированное;
- событийное;
- визуальное;
- автоматное.

Самым распространенным и легким для понимания является структурное программирование. *Структурное программирование* – это не язык, а стиль, его основные принципы состоят в том, чтобы пользоваться при записи алгоритмов ограниченным набором конструкций. Эти конструкции называются базовыми управляющими конструкциями структурного программирования.

Структурное программирование – это стиль программирования, позволяющий разрабатывать хорошо структурированные программы. Оно представляет собой некоторые принципы написания программ в соответствии со строгой дисциплиной и имеет целью облегчить процесс тестирования, повысить производительность труда программистов, улучшить ясность и читабельность программы, а также повысить ее эффективность.

Основные положения структурного программирования:

1. Программа разбивается на блоки, каждый из которых имеет один «вход» и один «выход».
2. Любая программа может быть составлена из трех структур или блоков (линейная структура, ветвление, циклическая).
3. Алгоритмы при структурном программировании представляются в виде структограмм.

Принципы *модульного программирования* программных продуктов во многом сходны с принципами нисходящего проектирования. Сначала определяются состав и подчиненность функций, а затем – набор программных модулей, реализующих эти функции. Однотипные функции реализуются одними и теми же модулями. Функция верхнего

уровня обеспечивается *главным* модулем; он управляет выполнением нижестоящих функций, которым соответствуют *подчиненные* модули.

Объектно-ориентированное программирование – это методология разработки программ, основанная на представлении программы в виде совокупности объектов, каждый из которых является реализацией определенного класса. В объектно-ориентированном программировании (ООП) объект представляет собой элемент программного приложения, например, лист, ячейку, диаграмму, форму или отчет. Программный код и данные структурируются так, чтобы имитировалось поведение фактически существующих объектов. В объектно-ориентированном программировании важную роль играют три понятия (парадигма): *инкапсуляция; наследование; полиморфизм.*

При абстрагировании реальные процессы ограничиваются их функциями, существенными для программирования. Внутреннее содержимое объекта защищается от внешнего мира посредством *инкапсуляции.* Благодаря *наследованию* уже запрограммированные функциональные возможности можно использовать и для других объектов. *Полиморфизм* позволяет использовать различные объекты и по-разному реализуемые функции под одним именем.

Объекты являются программным представлением физических и/или логических сущностей реального мира. Они необходимы для моделирования поведения физических или логических сущностей, которые они представляют.

Объектно-ориентированный подход использует следующие базовые понятия:

- объект;
- свойство объекта;
- метод обработки;
- событие;
- класс объектов.

Объект – совокупность свойств (параметров) определенных сущностей и методов их обработки (программных средств).

Объект содержит *инструкции* (программный код), определяющие действия, которые может выполнять объект, и обрабатываемые *данные.*

Свойство – характеристика объекта, его параметр. Все объекты наделены определенными свойствами, которые в совокупности выделяют объект из множества других объектов.

Объект обладает *качественной* определенностью, что позволяет выделить его из множества других объектов и обусловливает независимость создания и обработки от других объектов.

Например, объект можно представить перечислением присущих ему свойств:

ОБЪЕКТ_А (свойство-1, свойство-2,...., свойство-k).

Метод – программа действий над объектом или его свойствами.

Метод рассматривается как программный код, связанный с определенным объектом; осуществляет преобразование свойств, изменяет поведение объекта.

Класс – совокупность объектов, характеризующихся общностью применяемых методов обработки или свойств.

Один объект может выступать объединением вложенных в него по иерархии других объектов.

В объектно-ориентированном программировании используется следующий формат записи работы и работы с объектами (так называемая *точечная нотация*):

ОБЪЕКТ–МЕТОД;

ОБЪЕКТ–СВОЙСТВО–МЕТОД.

Программный продукт, созданный с помощью инструментальных средств объектно-ориентированного программирования, содержит объекты с их характерными свойствами, для которых разработан графический интерфейс пользователя. Как правило, работа с программным продуктом осуществляется с помощью экранной формы, с объектами управления, которые содержат методы обработки, вызываемые при наступлении определенных событий. Экранные формы также используются для выполнения заданий и перехода от одного компонента программного продукта к другому.

Каждый объект управления обладает определенными свойствами, значения которых могут изменяться. Для объектов управления уточняется перечень событий и создаются пользовательские методы обработки – программный код на языке программирования в виде *событийных процедур*.

Событийное программирование является развитием идей нисходящего проектирования, когда постепенно определяются и детализируются реакции программы на различные события.

Событие – изменение состояния объекта. *Внешние события* генерируются пользователем (например, клавиатурный ввод или нажатие кнопки мыши, выбор пункта меню, изменение размеров объекта, изменение видимости объекта, запуск макроса); *внутренние события* генерируются системой.

В визуальном программировании разработка программы или приложения производится с помощью выбора визуальных компонент и

размещения их на главное окно приложения, которое представляет собой интерфейс приложения. Визуальное программирование поддерживает идеологию автоматического написания программного кода. Программирование сводится к тому, что программист пишет только обработчик событий, которые связаны с соответственными визуальными компонентами.

В последнее время активно развивается такой стиль программирования, как автоматный.

Автоматное программирование – подход к разработке программных систем со сложным поведением, основанный на модели автоматизированного объекта управления (расширении конечного автомата). Этот подход позволяет создавать качественное программное обеспечение для ответственных систем, охватывая все этапы его жизненного цикла и поддерживая его спецификацию, проектирование, реализацию, тестирование, верификацию и документирование.

1.9. Краткий обзор современных языков программирования

Хронология создания некоторых языков программирования приведена на рисунке 1.8 (до 2000 года) и рисунке 1.9 (до 2005 года).

Рис. 1.8. Хронология создания языков программирования до 2000 года

Ниже приведены краткие характеристики языков программирования, включенных в обзор.

C (Си). Главный инструмент системного программиста на сегодняшний день. Си создавался программистами Bell laboratories в 70-е годы как инструмент для разработки ОС UNIX и получил первоначальное распространение как базовый язык этой операционной системы.

Си имеет очень приятный набор операторов (исключение составляет неструктурный switch) и неудачный конструктор типов. Серьезный недостаток (для языка системного программирования) – почти полное отсутствие механизмов раздельной компиляции. Си во многом похож на Паскаль и имеет дополнительные средства для прямой работы с памятью *(указатели)*.

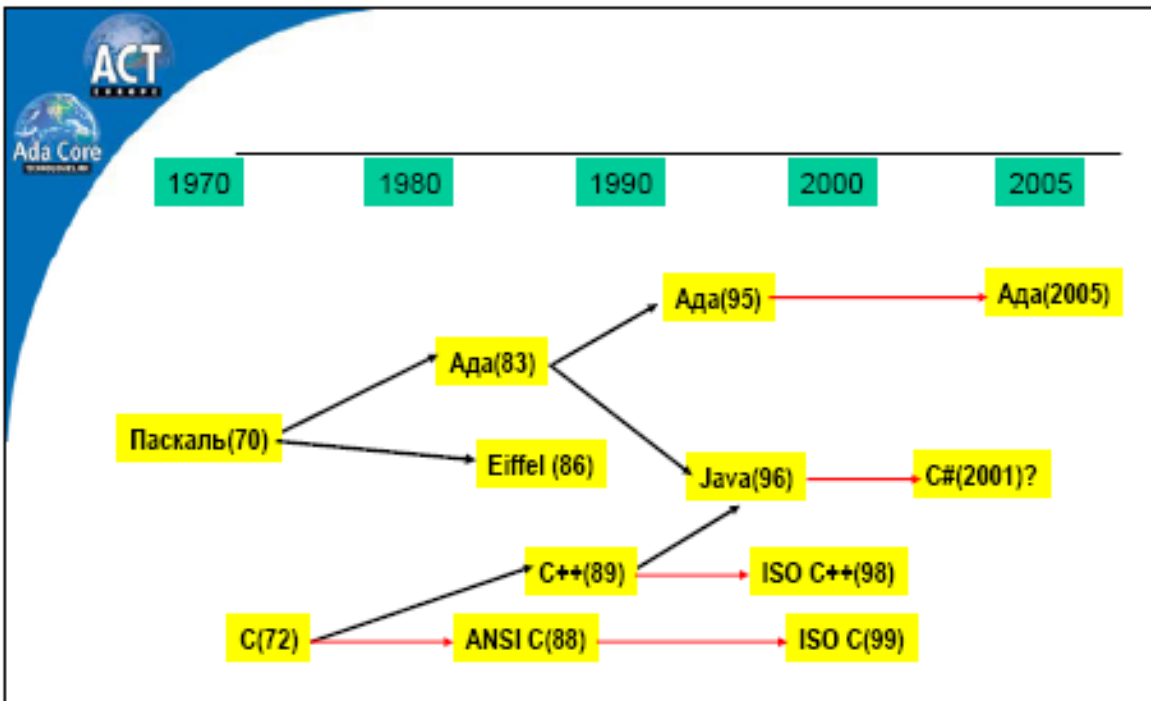


Рис.1.9. Хронология создания языков программирования до 2005 года

Поскольку Си доминирует на американском рынке средств разработки ПО, ему гарантирована долгая жизнь, несмотря на многочисленные попытки модернизировать его (например, такие, как C++ или Java).

Наиболее популярный клон языка С, в котором реализован наиболее полный (на сегодняшний день) механизм объектно-ориентированного программирования.

При создании языка делались также попытки модернизировать C, введя в его состав такие современные конструкции, как скалярный тип (enum), передача параметра по ссылке (&) или логический тип (bool).

C++ *(Си плюс-плюс)*. К сожалению, последовательными эти попытки назвать нельзя. Получился очень объемный (по этому показателю с ним может соперничать только Ада) и очень несистемный, «рыхлый» язык программирования, где наряду с суперсовременными понятиями класса соседствует адресная арифметика, доставшаяся в наследство от C.

Java (Ява). Самый «молодой» из включенных в обзор языков программирования и основной инструмент программирования для Internet. Создатели Java безжалостно удалили из C все несовременные конструкции, и в то же время сумели удержаться от излишнего «раздувания» языка включением в него новых теоретических разработок. В результате получился не очень объемный, но стройный, «крепко сбитый» язык программирования с ярко выраженной идеологией. К сожалению, ориентация на Internet не дает возможности использовать Java как язык системного программирования, однако это хороший пример реформы C. Клоном C он является только внешне. Идеологически это хороший пример европейской языковой школы (к которой можно отнести клоны PASCAL и ADA).

PASCAL (Паскаль). Ветеран европейской языковой школы, первый и самый популярный из языков, созданных К. Виртом. В языке реализован классический набор операторов и идеальный конструктор типов. Языку очень вредит отсутствие стандартных механизмов раздельной компиляции и ООП. В отличие от своего сверстника и конкурента C, PASCAL создавался одномоментно, что и предопределило его малые размеры и идеологическую стройность. С появлением MODULA 7, Ада-95 и OBERON PASCAL можно считать устаревшим языком, однако консервативность программного обеспечения не позволяет считать такой вывод окончательным.

MODULA-2. Наиболее известный клон PASCAL. Классический набор операторов и конструктор типов. Хорошо разработанные механизмы раздельной компиляции (конкуренцию MODULA в этом классе может составить только ADA). Маленький и удобный язык с точки зрения разработчика компилятора (как и все языки Н. Вирта, видимо, сказывается то обстоятельство, что Н. Вирт сам пишет компиляторы для своих языков). Недостатком языка можно считать полное отсутствие механизмов ООП.

OBERON–2. Последний из языков Вирта и клонов PASCAL. OBERON позиционировался как MODULA + ООП, однако при

создании языка Вирт выбросил из MODULA много приятных возможностей (часть из которых была добавлена при создании OBERON-2, считающегося современным вариантом языка). К сожалению, необратимо пострадал механизм раздельной компиляции (крайне важный для языка системного программирования).

С точки зрения идеологии OBERON скорее наследник MODULA-1, чем MODULA-2. Реализации ООП сделана очень красиво и с минимальными издержками для разработчика компилятора. Самый мощный из используемых сегодня языков программирования, Ада вызывает противоречивые чувства. С одной стороны, это самый яркий представитель европейской школы языков программирования, в котором реализовано множество теоретических наработок.

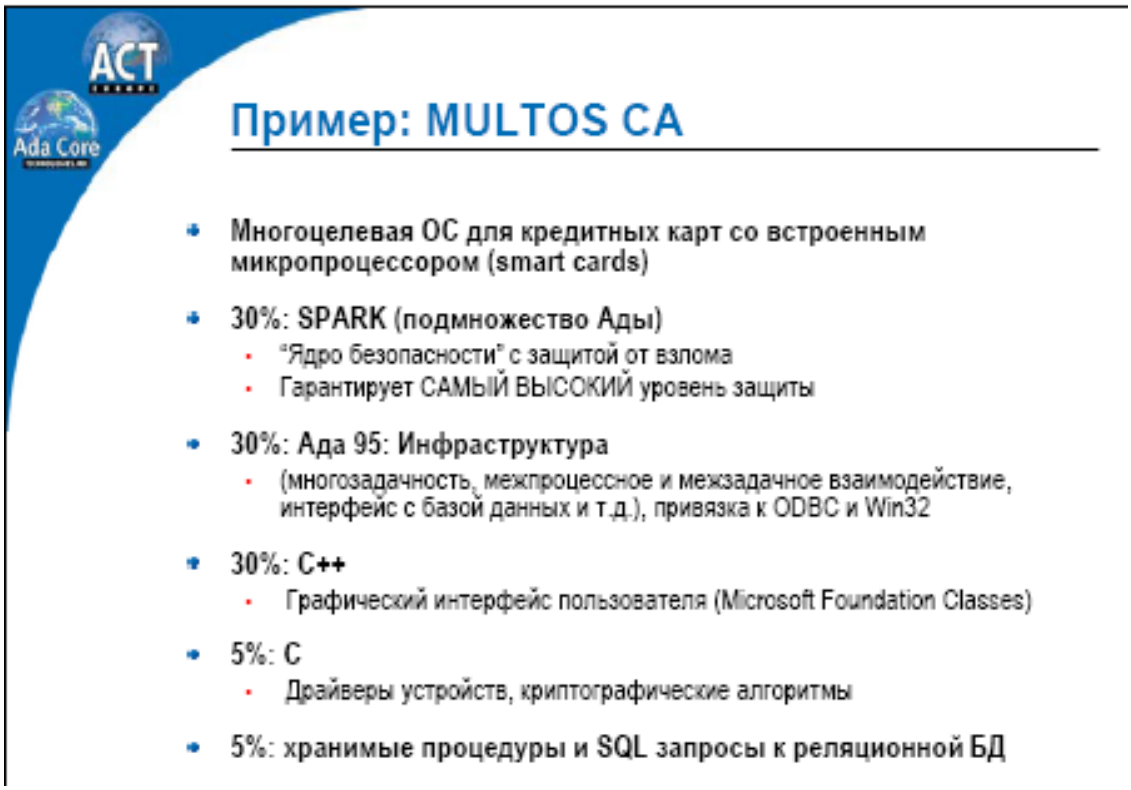


Рис. 1.10. Пример использования различных языков программирования

Несомненным плюсом языка Ада можно считать наиболее полную и подробную стандартизацию, вследствие чего переносимость программ на Ада наиболее лучшая по сравнению с другими языками программирования. Примером использования различных языков программирования в зависимости от решаемых задач может служить система MULTOS CA (рисунок 1.10).

1.8. Критерии выбора языка при программном принципе построения алгоритма

При программном принципе построения алгоритмы оформляются, как правило, в виде библиотеки функций и процедур. В дополнение к библиотеке создается интерфейс прикладного уровня для одного или нескольких языков программирования. Библиотека может поставляться в виде исходного текста на выбранном языке программирования или в бинарном виде.

В случае поставки в виде исходных текстов достигается переносимость библиотеки на другие аппаратные платформы и в какой-то степени на различные операционные системы. Переносимость в этом случае напрямую зависит от совместимости средств разработки для выбранного языка программирования на различных аппаратно-програм-мных платформах. В целом поставка библиотеки в виде исходных текстов на алгоритмическом языке высокого уровня обеспечивает максимально возможную переносимость и повторное использование, но требует этапа компиляции при использовании на целевой системе. Это усложняет использование библиотеки и отладку целевой системы, кроме того исходные тексты библиотеки не защищены от случайной или преднамеренной модификации. Возникают трудности в защите авторских прав и т.д. Поэтому исходные тексты библиотеки поставляются как дополнительная опция, а сама библиотека поставляется в бинарном виде.

Бинарный вид библиотеки может быть в нескольких форматах:

а) статическая объектная библиотека;

б) динамическая библиотека;

в) COM (DCOM) модуль.

Форматы хранения и переносимость ограничиваются одной аппаратно-программной платформой. Как правило, для каждой платформы генерируется своя библиотека. В любом случае бинарный вид библиотеки предполагает использование процессора с совместимой архитектуры и набором инструкций.

Таким образом, очень важно правильно выбрать средства разработки обеспечивающие:

а) переносимость библиотеки;

б) интерфейс прикладного уровня для нескольких алгоритмических языков высокого уровня, используемых в прикладной области целевых систем;

в) сопровождаемость разработанной библиотеки;

г) правильный генерируемый бинарный код;

д) другие требования, в зависимости от прикладной области целевых систем.

С другой стороны, средства разработки программных сред (ПС) должны предоставлять системным аналитикам средства достаточно формального и однозначного определения проектных решений целевых прикладных систем, подлежащих реализации в виде совокупности конкретных пакетов прикладных программ, образующих целостную систему.

Надежность при проектировании применительно к выбору языка программирования является мерой степени автоматического обнаружения ошибок, которое может быть выполнено либо компилятором, либо системой, поддерживающей выполнение скомпилированной программы.

В общем случае выбранный язык должен быть таким, чтобы как можно больше ошибок обнаруживалось во время компилирования, а не во время работы скомпилированной. Это необходимо по двум причинам:

а) чем раньше при разработке процесса обнаружена ошибка, тем меньше общая стоимость (Компиляция может быть выполнена на любой воспринимающий язык машине, тогда как тестирование при прогоне программы должно выполняться на фактическом целевом машинном оборудовании. Следовательно, тестирование прогоном часто будет очень дорогим, особенно когда оно приводит к прерыванию работ отдельных средств оборудования);

б) для обнаружения ошибок во время прогона часто требуется внесение компилятором в выходной объектный код добавочных контрольных команд, что приводит к дополнительным расходам как в терминах скорости выполнения, так и в терминах использования памяти, в обоих случаях не способствует эффективному функционированию в реальном времени.

Принципиальным средством достижения высокой надежности компиляционного уровня является система строгой типизации. Существенное преимущество этого подхода состоит в том, что при этом значительно увеличивается ясность программ, так как область значений, принимаемых каждой переменной, устанавливается явно.

Следует подчеркнуть, что, безусловно, имеется предел числа ошибок, которые могут быть обнаружены любой языковой системой. Ошибки в логическом построении программы автоматически не могут быть обнаружены. Впрочем, ошибки такого рода будут случаться реже, если сам язык поощряет программиста писать ясные, хорошо структурированные алгоритмы, предоставляя ему возможность выбора

подходящих языковых конструкций. Отсюда следует, что надежный язык должен также быть хорошо структурированным и удобочитаемым языком.

Удобочитаемость языка зависит от широкого спектра факторов, включающих, с одной стороны, выбор ключевых слов, а с другой – возможности модуляризации программ. Главная задача состоит в выборе такой четкой нотации языка, которая позволяла бы при чтении текста программы легко выделять основные понятия каждой конкретной части программы, не обращаясь к сопровождающим программу блок-схемам и словесным описаниям.

Высокая степень удобочитаемости оказывается полезной с различных точек зрения:

а) уменьшается стоимость документирования, если центральным элементом документации является сама программа (это весьма справедливо, в частности, для проектов программного обеспечения с большим временем существования, когда поддержание обновляемой сопроводительной документации в условиях неизбежного множества последовательных модификаций может оказаться весьма трудным делом);

б) позволяет легче понимать программу и, следовательно, быстрее находить ошибки (в результате увеличивается надежность);

в) позволяет легче сопровождать программу (ясно, что существенные изменения могут быть сделаны лишь тогда, когда работа программы понимается совершенно правильно).

Очевидно, что реализация требований хорошей удобочитаемости зависит от самого программиста, который должен постараться по возможности четче структурировать свою программу и так расположить ее текст, чтобы подчеркнуть эту структуру. Тем не менее, важную роль играет и структура используемого программистом языка. На нижних уровнях программных конструкций язык должен обеспечить возможности четкой спецификации того, какие объекты данных подвергаются обработке и как они используются. Эта информация определяется выбором идентификаторов и спецификаций типов данных. На более высоких уровнях программных конструкций язык должен обеспечивать возможности модуляризации и управления областями действия имен. Общее поведение программы гораздо легче понять, когда она составлена из ряда автономных операционных единиц, каждая из которых должна быть понята вне связи с остальными частями программы. Это имеет особое значение тогда, когда программе предстоит пройти этап модификации. Если текст программы прямо указывает на ее логическую структуру, то влияние

любого изменения на всю программу гораздо легче проследить, так как нужно исследовать подробно только ту логическую единицу, в которую вносится изменение.

Следует принять как должное увеличение длины программы, являющееся неизбежной платой за выполнение требования хорошей удобочитаемости. Некая многословность – это малая плата за все остальные выгоды, особенно если вспомнить, что программа пишется лишь один раз, а читается, в общем-то, многократно.

Гибкость. Язык должен предоставить программисту достаточно различных возможностей для выражения всех операций, которые требуются в программе, не заставляя его прибегать к вставкам машинного кода или различным ухищрениям. Критерий гибкости особенно существен в режиме реального времени, где, возможно, потребуется работать с широким спектром нестандартного периферийного оборудования. Для достижения высокой гибкости приходится идти на компромисс с требованиями надежности. Как правило, при разработке языка обеспечивается достаточная гибкость, соответствующая выбранной области применения и не больше.

Простота уменьшает затраты на обучение программиста и уменьшает вероятность совершения программистских ошибок, возникающих в результате неправильной интерпретации спецификаций языка. Обычно простота языка уменьшает и размер компилятора, облегчает получение более эффективного объектного кода и увеличивает мобильность компилятора.

Для достижения простоты не обязательно избегать сложных языковых механизмов, скорее при введении некоторой языковой особенности нужно стараться не накладывать случайные ограничения на ее использование. В общем случае основные правила языка изучить легко, трудно запомнить связанные с ними списки условий и ограничений на их применение

Мобильность. Как и в случае критерия простоты, необходимость мобильности также очевидна. Проектирование языка, независимое от базового машинного оборудования, дает возможность переносить программное обеспечение с машины на машину с относительной легкостью. Это позволяет высокую стоимость программного обеспечения распределить на целый ряд машинных конфигураций.

На практике мобильности достичь довольно трудно, особенно в системах реального времени, где одна из противоречивых задач состоит в извлечении максимальных выгод из базового машинного оборудования. В этом отношении особенно трудно поддаются

решению проблемы, связанные с различающимися длинами слов памяти.

Очень важным моментом является принятие стандарта языка программирования и соответствие средств разработки этому стандарту.

Эффективность. Системы реального времени часто должны обеспечивать высокую вычислительную пропускную способность, чтобы не нарушить ограничений, налагаемых управляемых ими внешним оборудованием. Поэтому язык должен быть построен так, чтобы его можно было реализовать эффективно. Более того, поскольку необходимо гарантировать определенное время ответа, следует избегать языковых конструкций, ведущих к непредсказуемым издержкам прогона программ (например, сборки мусора в схеме динамического распределения памяти).

В связи снижающейся стоимостью машинного оборудования и все возрастающей стоимостью ПС, основной целью при выборе языка программирования является обеспечение надежных и недорогих программ. Хотя необходимость эффективной реализации сохраняется, она постепенно отходит на второй план.

Требования к языку для систем реального времени. Шесть основных критериев языка реального времени перечислены в приблизительном порядке их важности. Отсюда следует, что требования высокой надежности и хорошей удобочитаемости считаются совершенно необходимыми для успешного применения языка в ПС реального времени.

Необходимость высокой надежности оказывает большое влияние на все аспекты проектирования языка, основой надежного языка является система его типов данных. Тип данных специфицирует область значений, которые может принимать объект этого типа, и множество операций, применимых к объектам этого типа. Поэтому тип данных определяет, что представляет собой объект, и накладывает ограничения на то, как он может использоваться. Язык реального времени должен иметь логичную систему типов данных, так чтобы появляющиеся во время манипуляции данными ошибки можно было обнаружить, а также, чтобы можно было однозначно специфицировать предлагаемый вид использования данных.

Необходимо также точно специфицировать действия, которые должна выполнять программа. Выгоды «структурного программирования» признаются всеми. Чтобы язык помогал написанию хорошо структурированных программ, он должен содержать конструкции для представления различных условных и итерационных структур управления и для объединения логически

соотносящихся действий в процедуры и функции. Чтобы удовлетворить требованиям «крупномасштабного программирования», язык реального времени должен также предоставлять механизм для модуляризации.

Реальные системы, естественно, описываются как множество одновременно выполняемых задач, язык реального времени должен предоставлять некоторые возможности для мультипрограммирования. Это может быть сделано либо путем спецификации стандартного интерфейса с мультипрограммной операционной системой, либо (что более предпочтительно) путем предоставления в самом языке возможностей описания мультипроцессорных систем.

Одной из уникальных характеристик ПС реального времени является необходимость работы с нестандартными устройствами ввода–вывода. Отсюда следует, что язык должен обеспечивать некоторые возможности прямого программирования работы устройств низкого уровня.

Для достижения как можно более высокой надежности ПС реального времени должно обладать способностью справляться с ситуациями, когда проявляются ошибки. Язык должен иметь стандартизованные возможности для обработки ошибок.

Таким образом, несомненное сходство языков программирования состоит в основных компонентах, таких, как набор операторов, конструктор типов, механизм процедур и ООП. Различия проявляются только в деталях, семантически современные языки программирования практически идентичны.

Необходимо понимать, что нет единого языка программирования на все случаи жизни. При создании сложных систем используется смесь различных языков программирования.

Таким образом, при выборе языка программирования необходимо учитывать другие свойства языка, важные для области его применения, значительное влияние при этом оказывает также и экономический аспект, наличие подготовленных сотрудников, принятые стандарты на предприятии и другие факторы временного или местного значения.

1.10. Инструментарий решения функциональных задач. Понятие информационной технологии

Рядовые пользователи редко обращаются непосредственно к программированию для решения функциональных и вычислительных задач. Для этого они используют уже готовые программные продукты, среды и оболочки.

Например, решение функциональных и вычислительных задач можно осуществить средствами пакета прикладных программ *MathCAD2000*. MathCAD – это мощная и в то же время простая универсальная среда для решения задач в различных отраслях науки и техники, финансов и экономики, физики и астрономии, строительства и архитектуры, математики и статистики, организации производства и управления.

Она располагает широким набором инструментальных, информационных и графических средств (рис. 1.11).

Сегодня MathCAD – одна из самых популярных математических систем. Она пользуется большим спросом у студентов, инженеров, экономистов, менеджеров, научных работников и всех тех, чья деятельность связана с количественными методами расчета.

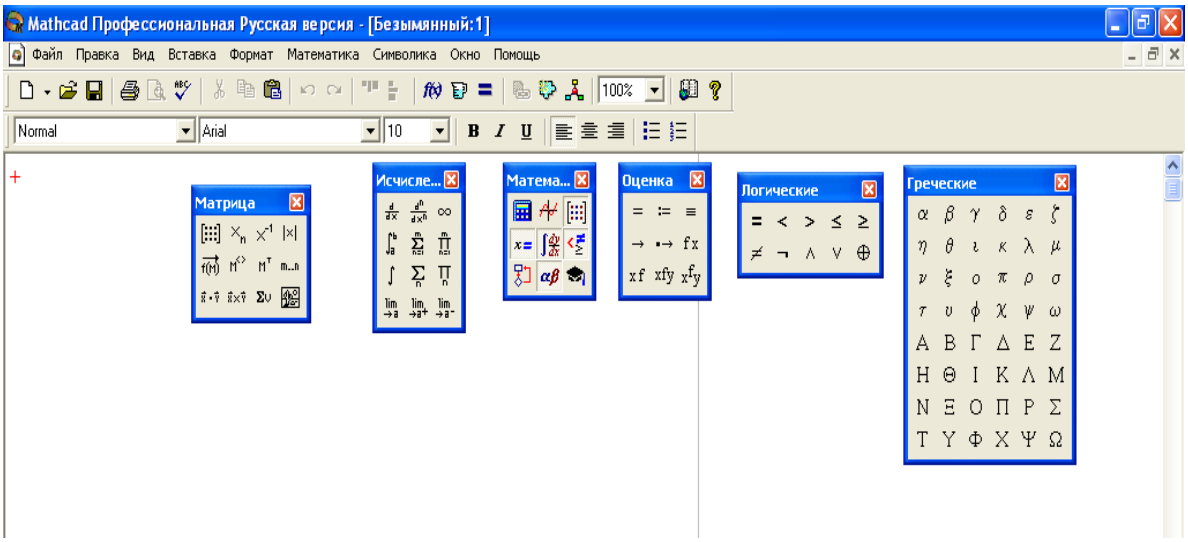


Рис. 1.11. Вид главного окна MathCAD 2000

В настоящее время разработано и функционирует множество различных математических систем: Maple, Mathlab, Mathematica, Reduce, Derive, Theorist, Macsyma и др. Каждая из них имеет свои преимущества и недостатки, а также свои области применения.

В чем же отличие системы MathCAD от аналогичных?

В математических системах Reduce, Macsyma, Derive, Maple, Mathematica, Theorist в основном используются целочисленное представление и символьная обработка данных, а в Matlab преимущественно ориентированна на работу с массивами. Запись задач в MathCAD наиболее приближена к записи их без использования компьютера, что существенно упрощает применение системы.

Система MathCAD более доступна для массового пользователя: она в несколько раз дешевле своих аналогов (речь идет о

лицензионных продуктах). Система MathCAD – это, скорее, универсальная, чем специализированная математическая система. MathCAD имеет встроенную систему автоматического пересчета и контроля единиц измерений в процессе вычислений.

MathCAD имеет достаточно мощную, но простую систему наглядного представления результатов расчета в виде различного рода графиков. MathCAD может взаимодействовать с другими приложениями.

С помощью MathCAD 2000 Professional можно вводить исходные данные (как в обычном текстовом редакторе), традиционно описывать решение задачи и получать результаты вычислений в аналитическом и численном виде с возможностью использования средств графического представления результатов. Запись математических выражений производится с применением общепринятых знаков (квадратный корень, знак деления в виде горизонтальной черты, знаки интеграла, дифференциала, сумму и т.д.).

В MathCAD 2000 Pro встроены хорошо организованные текстовый, формульный и графический редакторы. Они оснащены удобным пользовательским интерфейсом и разнообразными математическими возможностями.

В последних версиях MathCAD допускается импортировать любые графические изображения (от простых графиков функций до специализированных чертежей системы AutoCAD) и использовать средства анимации, звуковые и стереофонические эффекты.

Программа MathCAD 2000 Professional оснащена приложениями SmartSketch, Axum LE, Autodesk’s Volo View, MathCAD add-in for Excel.

SmartSketch позволяет аннотировать рабочие документы с рисунками, диаграммами, изображениями. Данное приложение обеспечивает параметрическое управление, как рисунками, так и вычислениями. Axum LE дает более полный контроль над двумерными графиками. Autodesk’s Volo View обеспечивает просмотр изображений AutoCAD в MathCAD. С помощью MathCAD add-in for Excel можно работать в Excel.

Другой пример инструмента решения функциональных и вычислительных задач – *Microsoft Office для Windows*. Microsoft Office для Windows представляет собой интегрированный программный комплекс. Это означает, что входящие в него компоненты могут использоваться как отдельно каждый, так и вместе для решения повседневных деловых задач. Истинная многозадачность Windows позволяет легко переключаться между приложениями Office (рис.1.12).

Рис. 1.12. Состав интегрированного программного комплекса MS Office

В качестве следующего примера программной среды для решения функциональных задач пользователя предлагаем коммуникационную среду Moodle.

Moodle поддерживает весь традиционный функционал образовательных коммуникационных сред, множество дополнительных функций и может быть гибко надстроен и адаптирован к нуждам конкретного образовательного учреждения.

На сайте Юргинского Технологического Института Томского Политехнического Университета Кафедра информационных систем разместила образовательную среду обучения Moodle (http://moodle.uti.tpu.ru:8080). Для студентов специальности 080801 Прикладная информатика (в экономике) и смежных с ней разработан электронный образовательный ресурс по дисциплине Информатика и программирование.

В среде Moodle студенты имеют возможность изучать дополнительный материал к лекциям, практическим занятиям и лабораторным работам, пользоваться глоссарием, общаться через чат, вносить предложения по курсу, узнавать о новинках информационных технологий, просмотреть презентации, задания, сдать контрольные точки, пройти тестирование и др.

Преподаватель имеет в своём распоряжении удобный инструмент для контроля знаний студентов, общения с ними, проверки заданий, контроля активности посещения ресурса и многое др.

Сущность информационной деятельности как атрибута основной человеческой деятельности заключается в информационном обеспечении потребностей человека или общества, возникающих в материальной или духовной сферах жизни, и выражается в подготовке информации для непосредственного использования в каждом конкретном случае.

Для того чтобы правильно выбрать «инструмент», действительно отвечающий всем требованиям и условиям поставленной задачи, необходимо хорошо ориентироваться в структуре и составе современного программного обеспечения, а также владеть средствами человеко-машинного интерфейса, т.е. иметь навыки применения *информационных технологий.*

Выше давалось определение ИТ как процесса, использующего совокупность средств и методов сбора, обработки и передачи данных (первичной информации) для получения информации нового качества о состоянии объекта, процесса или явления (информационного продукта).

Цель любой *информационной технологии* – производство информации для ее анализа человеком и принятия на его основе решения по выполнению какого-либо действия. Информационная технология представляет собой процесс, состоящий из четко регламентированных правил выполнения различных операций с данными, хранящимися в компьютере. ИТ базируется на следующих основных принципах:

- интерактивный (диалоговый) режим работы с компьютером;
- интегрированность с другими программными продуктами;
- гибкость процесса изменения данных и постановки задач

В качестве инструментария информационной технологии используются распространенные виды программных продуктов: текстовые процессоры, издательские системы, электронные таблицы, системы управления базами данных, электронные календари, информационные системы функционального назначения.

К *основным видам* информационных технологий относятся следующие:

- информационная технология обработки данных предназначена для решения хорошо структурированных задач, алгоритмы решения которых хорошо известны, и имеются все необходимые входные данные для успешного решения задачи. Эта технология применяется на уровне исполнительской деятельности персонала невысокой квалификации в целях автоматизации некоторых рутинных, постоянно повторяющихся операций управленческого труда;
- информационная технология управления предназначена для информационного обслуживания всех работников предприятий, связанных с принятием управленческих решений. Здесь информация обычно представляется в виде регулярных или специальных

управленческих отчетов и содержит сведения о прошлом, настоящем и возможном будущем предприятия;

- информационная технология автоматизированного офиса призвана дополнить существующую систему связи персонала предприятия. Автоматизация офиса предполагает организацию и поддержку коммуникационных процессов как внутри фирмы, так и с внешней средой на базе компьютерных сетей и других современных средств передачи и работы с информацией;
- информационная технология поддержки принятия решений предназначена для выработки управленческого решения, происходящего в результате итерационного процесса, в котором участвуют система поддержки принятия решений (вычислительное звено и объект управления) и человек (управляющее звено, задающее входные данные и оценивающее полученный результат);
- информационная технология экспертных систем основана на использовании искусственного интеллекта. Экспертные системы дают возможность менеджерам получать консультации экспертов по любым проблемам, о которых в этих системах накоплены знания.

Сами информационные технологии требуют сложной подготовки, больших первоначальных затрат и наукоемкой техники. Их введение должно начинаться с создания математического обеспечения, формирования информационных потоков в системах подготовки специалистов.

Главным направлением перестройки менеджмента и его радикального усовершенствования, приспособления к современным условиям стало массовое использование новейшей компьютерной и телекоммуникационной техники, формирование на ее основе высокоэффективных информационно-управленческих технологий.

Средства и методы прикладной информатики используются в менеджменте и маркетинге. Новые технологии, основанные на компьютерной технике, требуют радикальных изменений организационных структур менеджмента, его регламента, кадрового потенциала, системы документации, фиксирования и передачи информации. Особое значение имеет внедрение информационного менеджмента, значительно расширяющее возможности использования компаниями информационных ресурсов. Развитие информационного менеджмента связано с организацией системы обработки данных и знаний, последовательного их развития до уровня интегрированных автоматизированных систем управления, охватывающих по вертикали и горизонтали все уровни и звенья производства и сбыта.

Контрольные вопросы и задания

1. Какие этапы включает в себя информационный процесс?
2. Понятие ИС, ИТ, ЭИС.
3. Перечислите и охарактеризуйте основные компоненты ИС.
4. С чего должно начинаться решение любой задачи?
5. Схема информационного процесса.
6. Сущность и особенности принципа систематических методов составления алгоритмов и программ для решения различных прикладных задач.
7. Современные подходы к проектированию программ.
8. Перечислите основные стили программирования, их особенности.
9. Объясните сущность объектно-ориентированного анализа и его парадигмы.
10. Понятие программной среды. Приведите примеры программных сред.
11. Перечислите компоненты, которые необходимо иметь для создания программы на выбранном языке программирования.
12. Приведите примеры инструментария решения функциональных задач.
13. Понятие информационной технологии.
14. В чём цель любой информационной технологии?

2. СТРУКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С ТОЧКИ ЗРЕНИЯ ПОЛЬЗОВАТЕЛЯ

В настоящее время отсутствует единая классификация состава программного обеспечения. Литературные источники по-разному трактуют структуры программных средств ВМ различных классов. По мере развития ВМ и ВС программное обеспечение постоянно усложняется по своей структуре и составу программных модулей. В настоящее время затраты на разработку и приобретение программных продуктов в несколько раз превышают стоимость технических средств (Hardware). Особенно это касается обслуживания параллельных вычислений. Иерархический модульный принцип построения ПО современных ВМ и ВС (рис. 2.1) дает возможность их адаптации к конкретным условиям применения, открытость системы для расширения состава предоставляемых услуг, способность систем к совершенствованию, наращиванию мощности и т.д.

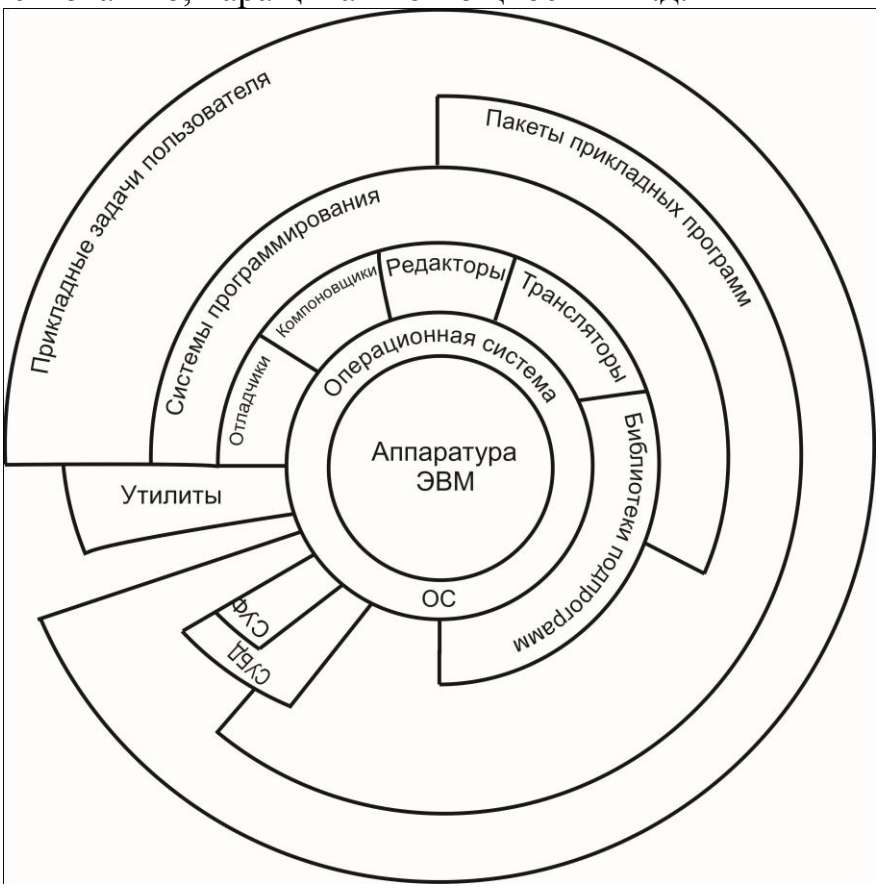


Рис. 2.1. Обобщённая структура программного обеспечения вычислительной машины

Программное обеспечение ВМ разделяют на *системное* или *базовое, общее* (general Software) и *специальное,* или *прикладное* (application or special Software).

Программные модули ПО, относящиеся к различным подсистемам, представляют для пользователя своеобразную иерархию программных компонентов, используемую им при решении своих задач.

Нижний уровень образуют программы ОС, которые играют роль посредника между техническими, аппаратными средствами системы и пользователем.

2.1. Системное программное обеспечение (СПО). Состав и назначение СПО

В англоязычной технической литературе термин *System Software* (системное программное обеспечение) означает программы и комплексы программ, являющиеся общими для всех, кто совместно использует технические средства компьютера, и применяемые как для автоматизации разработки (создания) новых программ, так и для организаций выполнения программ существующих.

Базовое или системное ПО объединяет программные компоненты, обеспечивающие многоцелевое применение ВМ и мало зависящие от специфики вычислительных работ пользователей. Сюда входят программы, организующие вычислительный процесс в различных режимах работы машин, программы контроля работоспособности ВМ, диагностики и локализации неисправностей, контроля вычислительного процесса, для управления распределением ресурсов во время функционирования вычислительной системы и для автоматизации разработки алгоритмов и программ, программы контроля заданий пользователей, их проверки, отладки и т.д. Данное ПО обычно поставляется потребителям комплектно с ВС.

Системное программное обеспечение может быть разделено на следующие пять групп:

- операционные системы;
- системы управления файлами;
- интерфейсные оболочки для взаимодействия пользователя с ОС и программные среды;
- системы программирования;
- утилиты и программы технического обслуживания.

Рассмотрим эти группы системных программ.

2.1.1. Операционные системы

Операционные системы (MS DOS, UNIX, OS/2, Windows, Linux и др.) – обязательное дополнение ВС, они предназначены для планирования и управления вычислительными ресурсами ВС, организуют выполнение пользовательских программ и взаимодействие пользователя с компьютером.

Операционные системы (ОС) выступают в роли посредника, т.е. обеспечивают интерфейс между аппаратурой компьютера и пользователем с его задачами. Любой из компонентов прикладного программного обеспечения обязательно работает под управлением ОС. На рис. 2.1 изображена обобщенная структура программного обеспечения вычислительной системы. Видно, что один из компонентов программного обеспечения, за исключением самой ОС, не имеет непосредственного доступа к аппаратуре компьютера. Пользователи взаимодействуют со своими программами через интерфейс ОС. Любые их команды, прежде чем попасть в прикладную программу, сначала проходят через ОС.

2.1.2. Системы управления файлами

Назначение *системы управления файлами* – организация более удобного доступа к данным, организованным как файлы. Именно благодаря системе управления файлами вместо низкоуровневого доступа к данным с указанием конкретных физических адресов нужной нам записи используется логический доступ с указанием имени файла и записи в нем.

Как правило, все современные ОС имеют соответствующие системы управления файлами. Однако выделение этого вида системного программного обеспечения в отдельную категорию представляется целесообразным, поскольку ряд ОС позволяет работать с несколькими файловыми системами (либо с одной из нескольких, либо сразу с несколькими одновременно). В этом случае говорят о монтируемых файловых системах (дополнительную систему управления файлами можно установить), и в этом смысле они самостоятельны.

Более того, можно назвать примеры простейших ОС, которые могут работать и без файловых систем, а значит, им необязательно иметь систему управления файлами, либо они могут работать с одной из выбранных файловых систем.

Надо, однако, понимать, что любая система управления файлами не существует сама по себе – она разработана для работы в конкретной ОС

и с определенной файловой системой. Можно сказать, что всем известная файловая система FAT (file allocation table) имеет множество реализаций как система управления файлами, например FAT-16 для самой MS-DOS, super-FAT для OS/2, FAT 32 для Windows NT, NTFS для Windows 2000 и т. д.

Подробно об эволюции операционных систем, их видах, назначении, функциях ОС, структуре файловых читайте в главе 4 данного учебника.

2.1.3. Интерфейсные оболочки и операционные среды

Для удобства взаимодействия с ОС могут использоваться дополнительные *интерфейсные оболочки*. Их основное назначение – либо расширить возможности по управлению ОС, либо изменить встроенные в систему возможности.

В качестве классических примеров *интерфейсных оболочек* и соответствующих *операционных сред* выполнения программ можно назвать различные варианты графического интерфейса X Window на системах семейства UNIX (например, К Desktop Environment в Linux), PM Shell или Object Desktop в OS/2 с графическим интерфейсом Presentation Manager; наконец, можно указать разнообразные варианты интерфейсов для семейства ОС Windows компании Microsoft, которые заменяют Explorer и могут напоминать либо UNIX с его графическим интерфейсом, либо OS/2, либо MAC OS. Следует отметить, что о семействе ОС компании Microsoft с общим интерфейсом, реализуемым программными модулями с названием Explorer (в файле system.ini, который находится в каталоге Windows, имеется строка SHELL=EXPLORER.EXE), все же можно сказать, что заменяемой в этих системах является только интерфейсная оболочка, в то время как сама операционная среда остается неизменной; она интегрирована в ОС.

Другими словами, операционная среда определяется *графическим интерфейсом прикладного программирования*, то есть API (application program interface). Интерфейс прикладного программирования (API) включает в себя управление процессами, памятью и вводом/выводом.

Например, существуют API для обеспечения визуальных функций (DirectX, OpenGL и т.п.), имеющие открытые стандарты кода, доступные каждому. DirectX – это популярный API от Microsoft. DirectX включает API для звука, музыки, устройств ввода и т.д. За 3D-графику в DirectX отвечает API Direct3D. Ранние версии DirectX – 3, 5, 6 и 7 – были относительно просты по возможностям. Разработчики могли выбирать визуальные эффекты из списка, после чего проверять их

работу, например в игре. Следующим важным шагом в программировании графики стал DirectX 8. В нём появилась возможность программировать видеокарту с помощью шейдеров, поэтому разработчики впервые получили свободу программировать эффекты так, как им нужно.

В DirectX 9 можно создавать ещё более сложные программы–шейдеры. DirectX 9c, обновлённая версия DirectX 9, включила спецификацию Pixel Shader 3.0. DirectX 10, версия API, сопровождающая Windows Vista.

Ряд операционных систем могут организовывать выполнение программ, созданных для других ОС. Например, в OS/2 можно выполнять как программы, созданные для самой OS/2, так и программы, предназначенные для выполнения в среде MS–DOS и Windows X. Соответствующая операционная среда организуется в операционной системе в рамках отдельной виртуальной машины. Аналогично, в системе Linux можно создать условия для выполнения некоторых программ, написанных для Windows X. Определенными возможностями исполнения программ, созданных для иной операционной среды, обладает и Windows NT. Эта система позволяет выполнять некоторые программы, созданные для MS–DOS, OS/2 1.x, Windows 3.x. В своем последнем семействе ОС Windows XP разработчики решили отказаться от поддержки возможности выполнения DOS-программ.

Наконец, к этому классу системного программного обеспечения следует отнести и *эмуляторы*, позволяющие смоделировать в одной операционной системе какую-либо другую машину или операционную систему. Так, известна система эмуляции WMWARE, которая позволяет запустить в среде Linux любую другую ОС, например Windows. Можно, наоборот, создать эмулятор, работающий в среде Windows, который позволит смоделировать компьютер, работающий под управлением любой ОС, в том числе и под Linux.

Таким образом, термин операционная среда означает соответствующий интерфейс, необходимый программам для обращения к ОС с целью получить определенный сервис – выполнить операцию ввода/вывода, получить или освободить участок памяти и т. д.

2.1.4. Системы программирования

О системах программирования или инструментарии программирования упоминалось в первом разделе данного пособия.

Инструментарий программирования или системы программирования можно разделить на следующие программные средства:

- *машинно-зависимые программные средства. Они* ориентированы на конкретную вычислительную машину, это языки машинных кодов, языки типа Ассемблер;
- *машинно-независимые программные средства. Они* не зависят от вычислительной машины, это алгоритмические языки высокого уровня, например, Фортран, Паскаль, Си. На язык машинных кодов программы, написанные на алгоритмических языках, переводятся специальными программами-трансляторами.

Система программирования представлена, прежде всего, такими компонентами, как транслятор с соответствующего языка, библиотеки подпрограмм, редакторы, компоновщики и отладчики. Не бывает самостоятельных (оторванных от ОС) систем программирования.

Любая система программирования может работать только в соответствующей ОС, под которую она и создана, однако при этом она может позволять разрабатывать программное обеспечение и для других ОС. Например, одна из популярных систем программирования на языке C/C++ от фирмы Watcom для OS/2 позволяет получать программы и для самой OS/2, и для DOS, и для Windows.

В том случае, когда создаваемые программы должны работать совсем на другой аппаратной базе, говорят о кроссплатформенных системах.

О системах программирования речь шла в пункте 1.6. Средства создания программ первой главы учебника.

2.1.5. Утилиты и программы технического обслуживания

Утилиты и программы технического обслуживания относят к *сервисным системам*, которые расширяют и дополняют пользовательский и программный интерфейс операционной системы.

Под *утилитами* понимают специальные системные программы, с помощью которых можно как обслуживать саму операционную систему, так и подготавливать для работы носители данных, выполнять перекодирование данных, осуществлять оптимизацию размещения данных на носителе и производить некоторые другие работы, связанные с обслуживанием вычислительной системы. К утилитам следует отнести и программу разбиения накопителя на магнитных дисках на разделы, и программу форматирования, и программу переноса основных системных файлов самой ОС. Также к утилитам относятся и

небезызвестные комплексы программ от фирмы Symantec, носящие имя Питера Нортона (создателя этой фирмы и соавтора популярного набора утилит для первых IBM PC). Естественно, что утилиты могут работать только в соответствующей операционной среде.

То есть *утилиты* – обслуживающие программы, которые предоставляют пользователю сервисные услуги. Их отличие от программ-оболочек в следующем: оболочки универсальны, а утилиты специализированы. Утилиты – программы вспомогательного назначения, они предоставляют дополнительные услуги, например, реализующие такие функции:

1) обслуживание магнитных дисков (например, UnErase – для восстановления ошибочно удаленных файлов и каталогов);

2) создание и обновление архивов: как со сжатием, так и без сжатия (например, утилиты PKZIP, PKUNZIP, архиватор-разархиватор AEJ);

3) тестирование аппаратных программных средств;

4) защита от компьютерных вирусов и т.д.

Приведём несколько примеров программ этого вида.

Advanced PDF Repair 1.1. Утилита для восстановления испорченных PDF-документов. Поддерживает работу с любыми версиями Adobe PDF-файлов. Интегрируется в оболочку Windows, что значительно упрощает и ускоряет восстановление PDF-файлов.

PicaJet Photo Transfer 1.0.4. Мощное и удобное приложение для упорядочения структуры папок с изображениями. В комплект программы входят наиболее популярные у фотографов шаблоны для организации изображений + редактор собственных шаблонов.

Icon Processor 2.0. Конвертирует BMP, JPEG, GIF, PNG, PSD, WMF, WBMP, XPM, XBM и CUR форматы файлов в иконки Windows, причем добавлять файлы и каталоги из Проводника или другой файловой оболочки можно, используя для этого метод Drag-n-Drop – перетащил–отпустил.

BB TestAssistant 1.4.2. Записывает всё, что происходит на экране компьютера, – движения мышью, перемещение и открытие окон и т.д., нажатия клавиш, а также звуки – в компактный видеофайл. Возможные форматы на выходе: Flash, AVI, EXE, WMV и PPT.

Image Analyzer 5.4.20. Программа для выявления наиболее эффективного способа сжатия изображений. Основная функция – показ информации об изображении (размеры, цвета и др.).

Iconix 1.3. Компактная программа для создания каталогов картинок (изображений) со ссылками в формате HTML. Поддерживается работа с файлами *.jpg, *.jpeg, *.png, *.gif. Умеет создавать по две ссылки на

один значок. Позволяет задавать диапазоны выбора файлов. Может сортировать значки.

A4Desk 5.73. Удобная бесплатная программа для создания различных элементов для сайта – кнопок, меню, других элементов навигации на Flash. A4Desk позволет также даже неопытному пользователю, не говоря о профессионалах, создавать целые сайты или презентации на Flash. Большие комплекты утилит, речь о которых пойдет ниже, – достаточно универсальные и вместительные пакеты. Например, в Fix–It! или Norton Utilities входит до нескольких десятков отдельных программ-утилит. Пристрастие производителей к гигантским пакетам, с одной стороны, и желание пользователей получить «все в одном» породило на свет комбинированные утилитные «офисы», объединяющие несколько наборов утилит.

Есть ли смысл выбирать для установки на свой компьютер именно утилитные «офисы»? Т. к. далеко не все программы из «офисов» реально понадобятся вам для каждодневной работы, а вместо некоторых из них лучше установить более серьезные и «умелые» пакеты. Так, программный комплекс AVP, созданный лабораторией Касперского, лучше других антивирусов приспособлен к отечественной «микрофлоре».

С другой стороны, достоинство «офисов» – в наличии единого центра управления всеми входящими в состав офиса утилитами и их хорошая уживчивость друг с другом.

Наборы утилит

Программы-«администраторы» управляют информацией на диске, чаще всего мы работаем лишь с кусочками информации на нашем винчестере – отдельными файлами или папками. Но иногда возникает ситуация, когда жесткий диск для нас вновь становится единым и мы начинаем смотреть на все его информационное пространство сразу. И тогда операция, которую нам необходимо выполнить, касается всего жесткого диска или, как минимум, одного из его разделов (например, форматирования или проверки диска). Программы для этих операций находятся в составе популярных наборов утилит – таких, как Norton SystemWorks.

Программы тонкой подстройки Windows. Разобраться в устройстве Windows не так просто, как кажется, особенно в том, что касается настроек. Хотя посредством панели управления можно получить доступ ко всевозможным настройкам, но при работе с этим инструментом обнаруживается, что имеющиеся возможности настройки и оптимизации весьма ограниченны. Эта ограниченность становится более ощутимой при установке на компьютер хотя бы одной утилиты

«тонкой подстройки». Это словосочетание в английском языке обозначается как Tweak. Дословно – «уловка, щипок». Программы-«твики» предоставляют пользователям Windows возможность изменить самые потаенные, скрытые настройки системы, которые тем не менее могут существенно улучшить внешний вид и скорость работы вашей ОС (но в неумелых руках – равно и замедлить).

Утилит такого класса существует немало. А знакомство с ними стоит начать с программы, созданной... самими разработчиками Windows (называется она TweakUI). Но созданной как бы неофициально, в свободное от работы время. Поэтому в состав самой Windows эта программа не включена, а устанавливать ее надо отдельно с лицензионного компакт-диска с установочным комплектом Windows.

Посредством TweakUI в вашем распоряжении окажется масса возможностей подстройки, хотя и несколько меньше, чем предлагают другие утилиты. С помощью этой программы вы получите возможность внести некоторые изменения во внешний вид рабочего стола, удалив с него лишние, по вашему мнению, иконки, которые обычными средствами не удаляются. Можно удалить лишние записи, оставшиеся от неправильно установленных программ, полностью автоматизировать процесс входа вашего компьютера в локальную сеть и др.

Тесты. Сколько требуется программ для того, чтобы аппаратная часть компьютера полноценно работала и имела грамотный уход? Во-первых (и это главное), необходимы драйверы для сопряжения Hard Ware с операционной системой. Они поставляются с каждым периферийным устройством или выбираются ОС из собственной библиотеки. Но немногие комплектующие имеют несколько альтернативных комплектов драйверов.

Во-вторых, часто вместе с «железом» поставляются еще и дополнительные программы-утилиты, необходимые для тонкой настройки или управления. Они гораздо менее привередливы, чем их коллеги-драйверы: одна и та же утилита может обслуживать целый модельный ряд комплектующих. Например, для каждой из звуковых карт Creative необходим свой драйвер, а проигрыватель PlayCenter, который можно найти на одном диске с драйверами, сможет осуществить сопряжение с любой картой. Но имеются программы, обладающие гораздо более широким «кругозором». Тесты – категория утилит, определяющая какое именно «железо» установлено на ПК.

Файловые менеджеры. Как известно, вся информация в компьютере хранится в виде файлов. Для облегчения различных операций с файлами имеются программы – файл-менеджеры. Они предоставляют более удобный интерфейс и различные дополнительные

утилиты. В Windows используется встроенный файл-менеджер Explorer (Проводник), но во многих случаях пользоваться им не очень удобно. Скажем, двухпанельное меню намного удобнее при копировании файлов, чем однопанельное (кстати, сама компания Microsoft рекомендует запускать два Проводника и располагать их рядом). В файл-менеджер встроены функции сортировки файлов, сравнения, архивирования и т. п.

При всем многообразии эти программы можно разделить на две большие группы. В первую входят подобия Проводника с добавлением некоторых полезных функций. А вторая группа представлена программами, имитирующими интерфейс самого популярного файлового менеджера прошлых лет – Norton Commander. Программы первой группы особенно популярны на Западе. В России отдают предпочтение файловым менеджерам второй группы, например программе Total Commander, ДИСКо Командир. Пробная версия, доступная на сайте (http://www.ghisler.com), исправно работает 30 дней, после чего при каждом запуске будет появляться окошко с предложением зарегистрироваться. Хотя функциональность программы от этого не теряется, но необходимо помнить, что данный программный продукт не является бесплатным.

Программы для работы с архивами. Поддержкой работы с архивами популярных форматов (zip, arj) сегодня удивить трудно. Редкий комплект утилит обходится без программы, помогающей пользователям просматривать файлы из архивов или самостоятельно упаковывать в файл-архив целые папки. Сегодня в ОС Windows уже встроен комплекс Compressed Folders, благодаря которому архивы становятся почти неотличимыми от обычных папок.

При наличии программ типа Compressed Folders или WinZip можно осуществлять многие операции с архивными папками, кроме установки программы из упакованной в архив папки.

Программы для просмотра графических файлов (вьюверы). Windows снабжён достаточно большим количеством возможностей по просмотру файлов различных форматов: графических, текстовых или мультимедийных. Однако встроенных программ просмотра *вьюверов* и *плееров* зачастую оказывается недостаточно.

Проигрыватели мультимедиа-файлов (плееры). Существуют отдельные программы-плееры для проигрывания различных форматов компьютерного звука и видео. Большую популярность среди музыкальных утилит завоевывают так называемые комбинированные проигрыватели. Многие из этих программ обладают довольно примечательной «внешностью», имитирующей бытовые музыкальные центры.

Win Amp (http://www.winamp.com). Наблюдается некоторый парадокс: несмотря на изобилие мощных мультимедийных комбайнов, таких как стандартный Windows Media Player, MusicMatch Jukebox или RealOne, пользователи ищут им альтернативу и находят – в виде маленькой, непритязательной на вид программы. Сегодня данный плеер распространяется на условно-бесплатной основе.

Win Amp – программа, созданная группой независимых разработчиков Nullsoft. Создавалась она для воспроизведения файлов лишь одного типа – «сжатого звука» формата MP3. Однако сегодня программа может воспроизводить почти все форматы, включая «виртуальные радиостанции», вещающие в сети Интернет, и даже видеофайлы.

Особенность Win Amp – наличие встроенного эквалайзера, с помощью которого можно довольно точно отрегулировать звучание файла. Расширить базовые функции Win Amp можно с помощью подключаемых программных модулей – плагинов (plug-ins). Одни из них влияют на качество воспроизведения звука, украшая его всевозможными спецэффектами или устраняя погрешности звучания (плагины DFX). Другие отвечают за визуальное сопровождение музыки. Третьи меняют внешний вид самого плеера (эта технология называется skins).

Win Amp можно применять в качестве «граббера», который превращает дорожки аудиодиска в подборку MP3-файлов. Подобно Windows Media Player, программа имеет менеджер MP3-файлов. Просканировав жесткие диски компьютера, вы можете автоматически добавить в библиотеку Win Amp все найденные дорожки.

Win Amp смог найти правильный баланс между качеством и «количеством», то есть объёмом программы. Win Amp в несколько раз меньше любого мультимедийного «комбайна», что же до дизайна, то, при всём его внешнем аскетизме, его нельзя не признать удобным и функциональным.

Quintessential Player (QCD) (http://quinnware.com).

Суперпопулярный универсальный проигрыватель, изрядно потеснивший в последние годы Win Amp. Однако QCD не воспроизводит видео. QCD, благодаря великолепному декодеру, имеет большие возможности при работе со звуком. Малый размер, высокое быстродействие, удобное устройство, большое количество дополнительных модулей-плагинов, поддержка смены skin-ов.

Power DVD Player (http://www.gocyberlink.com).

Power DVD Player может предоставить большое количество возможностей по управлению параметрами каждого диска.

Программа воспроизводит видео с высоким разрешением в системах NTCS и PAL. Позволяет «на лету» переключить каналы субтитров и звуковых дорожек. Power DVD поддерживает вывод звука на две, четыре или шесть колонок. Для большого числа колонок предусмотрена поддержка форматов Dolby Surround и Dolby ProLogie, но даже на двух колонках можно добиться отличного «объёмного» звука благодаря фирменной технологии TruSurround. Также удобные средства навигации, возможность автоматической установки «закладок»: прервав воспроизведение фильма на любом участке и вынув диск, вы можете быть уверены – в следующий раз программа начнёт его проигрывать именно с последней увиденной вами сцены.

Power DVD Player может воспроизводить VideoCD, KaraokeCD, MP3.

Программы технического обслуживания предназначены для облегчения тестирования оборудования и поиска неисправностей, они являются инструментом специалистов по эксплуатации аппаратной части компьютеров. Этот комплекс включает в свой состав наладочные, проверочные и диагностические тест-программы.

Наладочные программы обеспечивают автономную настройку и проверку отдельных устройств ВМ. Обычно они функционально независимы от программ ОС. *Проверочные тест-программы* предназначены для периодически проводимых проверок правильности функционирования устройств, например, после включения их в работу. *Диагностические программы* используются в тех случаях, когда необходимо классифицировать отказ оборудования и локализовать место неисправности. Инициирование работы этих программ осуществляется обычно модулями ОС после фиксации сбоев и отказов аппаратурой контроля.

У IBM PC эти средства имеют своеобразную структурную и функциональную организацию. Часть этих средств записана в ПЗУ компьютера. При каждом включении ПК и перезагрузках производится ее предварительная проверка путем выполнения тестовой программы POST (Power On Set Test), состоящей из более десятка отдельных программных фрагментов. Последовательность проверок заключается в следующем. Вначале проверяется работоспособность системного блока. Для этого все регистры машины «сбрасываются в нуль», и производится их последовательная проверка путем занесения отдельных констант, выполнения над ними простейших операций и сравнения результатов с эталонными значениями. После этого проверяются ячейки оперативной памяти. Затем проверяется стандартная периферия: клавиатура, накопители на дисках, дисплей и др. В случае каких-либо ошибок на

каждом шаге проверки формируются определенные звуковые сигналы, сопровождаемые соответствующими сообщениями на экране дисплея.

Кроме встроенных средств контроля, в ПО ПК включаются и автономные средства контроля и диагностики. Количество подобных комплектов программ достаточно велико, и каждый из них позволяет детализировать системную информацию: определение полной конфигурации ПК и характеристик отдельных ее частей (тип процессора, наличие сопроцессора, тип материнской платы, типы используемых дисков, объем оперативной памяти и ее распределение, подключение дополнительной периферии и т.д.). Помимо контроля работоспособности, они могут отразить, насколько эффективно используются ресурсы, и осуществить их перераспределение.

2.2. Прикладное программное обеспечение

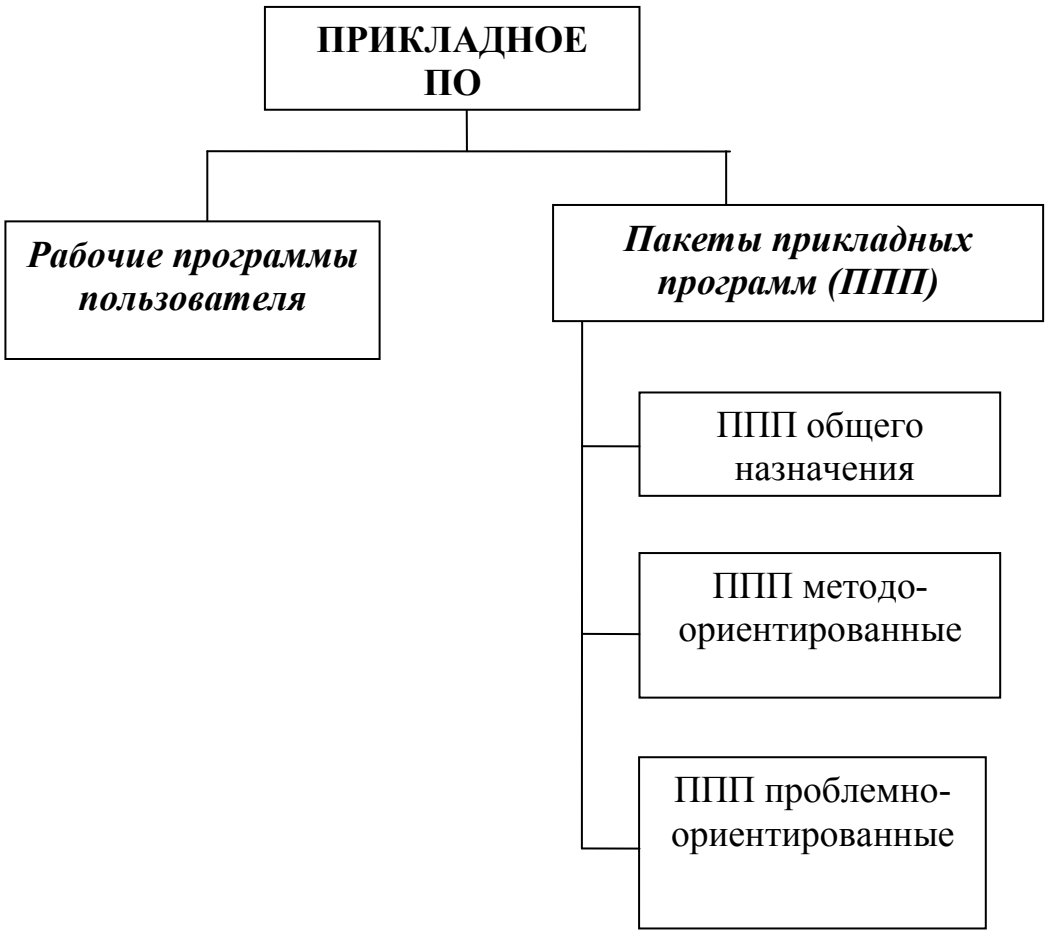


Рис. 2.2. ППО с точки зрения пользователя

Прикладное ПО (ППО) – совокупность программ для решения задач из различных сфер человеческой деятельности. ППО содержит

пакеты прикладных программ пользователей, обеспечивающие специфическое применение ВМ и ВС.

Иногда ППО разделяют на рабочие программы пользователя и пакеты прикладных программ (рис. 2.2).

Рабочие программы пользователя создаются каждым пользователем индивидуально с использованием средств программирования в соответствии с правилами той операционной системы, под управлением которой работает вычислительная система.

Обычно прикладные программы объединяются в пакеты, что является необходимым атрибутом автоматизации труда каждого специалиста-прикладника. Комплексный характер автоматизации производственных процессов предопределяет многофункциональную обработку данных и объединение отдельных практических задач в ППП. *Пакеты прикладных программ (ППП)* являются самым многочисленным классом программных продуктов. Они выполняют обработку информации различных предметных областей.

Пакет прикладных программ – комплекс взаимосвязанных программ для решения задач определенного класса конкретной предметной области.

Специализация пакета определяется характером решаемых задач (пакеты для разработки экономических документов, рекламных роликов, планирования и др.) или необходимостью управления специальной техникой (управление сложными технологическими процессами, управление бортовыми системами кораблей, самолетов и т.п.). Такие специальные пакеты программ могут использовать отдельные подразделения, службы, отделы учреждений, предприятий, фирм для разработки различных планов, проектов, документов, исследований. В некоторых случаях ППО может иметь очень сложную структуру, включающую библиотеки, каталоги, программы-диспетчеры и другие обслуживающие компоненты. Программы ППО разрабатываются с учетом интересов определенной группы пользователей, иногда даже по их заказам и при их непосредственном участии.

ППО ПК комплектуется в зависимости от места и роли автоматизированного рабочего места (АРМ) работника, использующего в своей деятельности компьютер. В ПО ПК обычно включают небольшое число пакетов программ (табличный процессор, текстовый редактор, система управления базами данных и др.). В последнее время наметилась тенденция к комплексированию и слиянию их в интегрированные программные продукты. Например, пакет MS Office фирмы Microsoft объединяет все перечисленные продукты.

В состав Microsoft Office входят следующие прикладные программы:

- Microsoft Word – универсальный редактор текстов и средство подготовки оригинал-макетов для печати;
- Microsoft Excel – (табличный процессор) – электронные таблицы с мощными средствами анализа данных и построения диаграмм, а также аналитическими функциями;
- Microsoft Access – реляционная система управления базами данных с возможностями создания запросов, отчетов;
- Microsoft Power Point – программа презентационной графики для создания слайдов и мультимедиа-презентаций;
- Microsoft Outlook – программа для управления личной и деловой информацией, адресными книгами, дневником и электронной почтой и др.

Объединив свои лучшие программы в единый прикладной комплекс, компания Microsoft создала универсальное средство для решения задач обработки данных, возникающих в современном бизнесе.

Во всех приложениях Microsoft Office используются стандартные команды, окна диалога и основные операции, в них используются похожие средства форматирования и макроязыки. Приложения проектировались для совместной работы, так что есть, например, возможность легко объединить текст из Word, диаграмму из Excel, информацию из базы данных Access в одной презентации.

ППП общего назначения автоматизируют типовые процедуры взаимодействия пользователя с компьютером при обработке информации. Сложилось несколько типов ППП данного класса:

- *текстовые процессоры* (редакторы);
- *графические редакторы;*
- *табличные процессоры* для обработки данных, представленных в виде таблиц;
- *системы управления базами данных (СУБД)* – для накопления данных в виде взаимосвязанных файлов на магнитных носителях и поиска данных по запросам;
- *интегрированные системы,* позволяющие обрабатывать текстовую информацию, таблицы, формировать базы данных и отображать рассчитанные зависимости в виде графиков;
- *сетевые пакеты,* дающие возможность пользователю работать с ресурсами вычислительной сети. *ППП глобальных сетей* обеспечивают удобный, надежный доступ пользователей к

распределенным общесетевым ресурсам, базам данных т.д. Стандартные ППП используют для организации электронной почты, телеконференций, электронной доски объявлений и др. Например, средства доступа и навигации в сети – Netscape Navigator, Explorer.

Методо-ориентированные ППП реализуют какие-либо экономико-математические методы решения задач. Например, методы математического программирования, математической статистики, сетевого планирования и управления и др.

Проблемно-ориентированные ППП предназначены для решения какой-либо задачи в конкретной функциональной области. Например, ППП бухгалтерского учета (1С:Бухгалтерия, БЭСТ и др.), банковские ППП (RS–BANK, Диасофт-БАНК и др.) и т.д.

В настоящее время на любой вид деятельности существуют, разрабатываются и совершенствуются ППП, позволяющие пользователям, даже не имеющим хорошей компьютерной подготовки, эффективно решать специфические задачи обработки информации (подготовка справок, писем, разработка документов, графическое представление данных и т.д.).

Подробно о составе прикладного программного обеспечения общего назначения, а также о пакетах проблемно-ориентированных прикладных программ читайте в главе 5 учебника.

Контрольные вопросы и задания

1. Приведите обобщённую структуру программного обеспечения вычислительной машины. Кратко опишите её суть.
2. Понятие системного программного обеспечения.
3. Состав и назначение СПО.
4. Опишите функции каждой составляющей СПО.
5. Объясните понятия: утилиты и утилитные офисы. Приведите примеры наборов утилит.
6. Классификация прикладного программного обеспечения.

3. ОРГАНИЗАЦИЯ И ПРИНЦИПЫ ЧЕЛОВЕКО-МАШИННОГО ИНТЕРФЕЙСА

3.1. Человеко-машинное взаимодействие

Человеко-машинное взаимодействие (HCI – Human-Computer Interaction) – это наука, которая изучает, как люди используют компьютерные системы, чтобы решить поставленные задачи. HCI обеспечивает нас знаниями о компьютере и человеке для того, чтобы взаимодействие между ними было более эффективным и удобным.

Область HCI включает в себя несколько дисциплин, т. к. разработчики программного обеспечения должны знать и понимать основы деятельности, поведения и ментальной специфики человека в соответствии с проектируемой системой.

Приведем некоторые из дисциплин, которые включает в себя HCI:

- эргономика;
- информатика;
- искусственный интеллект;
- лингвистика;
- психология;
- социология;
- основы разработки программного обеспечения;
- дизайн.

Взаимодействие между пользователем и компьютером

Человеко-машинный интерфейс обеспечивает связь между пользователем и компьютером: он позволяет достигать поставленных целей, успешно находить решение поставленной задачи. *Интерфейс пользователя (user interface)* – это средства взаимодействия компьютера с пользователем. Операционная система обеспечивает и поддерживает диалог пользователя с компьютером, что и достигается с помощью пользовательского интерфейса. Взаимодействие – обмен действиями и реакциями на эти действия между компьютером и пользователем.

Имеется ряд стилей взаимодействий, которые делятся на два основных вида.

Первый – это использование интерфейса языка команд, ввод команд текстовыми средствами. Это так называемые терминальные или командные системы типа MS DOS. Интерфейс в них построен на базе *командной строки (command prompt)*. Для выполнения какой-либо операции пользователю нужно было набирать в командной строке

соответствующие команды. Например, чтобы удалить документ *Текст*, требовалось ввести следующую команду: *delete c:/ Текст*.

Второй – это непосредственное манипулирование. Таким образом, имеется ряд способов, которыми пользователь мог бы связываться с компьютером:

- языки команд – пользователь управляет системой, вводя соответствующие команды в тестовом режиме (интерфейс командной строки);
- вопрос и ответ – диалог (диалоговый, интерактивный режим), где компьютер задает вопросы, а пользователь отвечает ему (или наоборот);
- формы – пользователь заполняет формы или поля диалога, вводя данные в необходимые поля;
- меню – пользователь обеспечен рядом опций и управляет системой, выбирая необходимые пункты;
- прямое манипулирование – пользователь управляет объектами на экране посредством устройства манипулирования типа мыши.

Другой термин, используемый для прямого интерфейса манипулирования,– графический интерфейс пользователя *(Graphical User Interface, GUI).* Каждый объект системы, будь то документ или программа, отображается графическим символом, называемым *пиктограммой* или *значком (icon)*. Большинство команд выполняется с помощью мыши: курсором выделяется объект, а затем пользователь выполняет команду. К примеру, чтобы удалить документ, нужно просто выделить соответствующий ему значок, нажать левую клавишу мыши и переместить пиктограмму на значок Корзина (Recycle Bin). GUI помогает пользователю при использовании многозадачного режима: каждая запущенная программа отображается на экране в отдельной области – *окне (window)*. Чтобы перейти из одного запущенного приложения в другое, достаточно щелкнуть мышью по любой части окна нужной программы. В различных операционных системах на сегодняшний день обычно используются *комбинированные* стили взаимодействия из приведенных выше. Такой подход важен для проектировщика автоматизированных систем, поскольку позволяет тщательно рассмотреть поставленную задачу заказчика (будущего пользователя), чтобы выбрать наилучший вариант решения задачи. Цель создания эргономичного интерфейса состоит в том, чтобы отобразить информацию настолько эффективно, насколько это возможно для человеческого восприятия, и структурировать отображение на дисплее таким образом, чтобы привлечь внимание к

наиболее важным единицам информации. Основная цель состоит в том, чтобы минимизировать общую информацию на экране и представить только то, что является необходимым для пользователя.

3.2. Основные принципы создания интерфейса

1. *Естественность (интуитивность).* Работа с системой не должна вызывать у пользователя сложностей в поиске необходимых директив (элементов интерфейса) для управления процессом решения поставленной задачи.

2. *Непротиворечивость.* Если в процессе работы с системой пользователем были использованы некоторые приемы работы с некоторой частью системы, то в другой части системы приемы работы должны быть идентичны. Также работа с системой через интерфейс должна соответствовать установленным, привычным для пользователя нормам (например, использование клавиши Enter).

3. *Неизбыточность.* Это означает, что пользователь должен вводить только минимальную информацию для работы или управления системой. Например, пользователь не должен вводить незначимые цифры (00010 вместо 10). Аналогично, нельзя требовать от пользователя ввести информацию, которая была предварительно введена или которая может быть автоматически получена из системы. Желательно использовать значения по умолчанию где только возможно, чтобы минимизировать процесс ввода информации.

4. *Непосредственный доступ к системе помощи.* В процессе работы необходимо, чтобы система обеспечивала пользователя необходимыми инструкциями. Система помощи отвечает трем основным аспектам: качество и количество обеспечиваемых команд; характер сообщений об ошибках и подтверждения того, что система делает. Сообщения об ошибках должны быть полезны и понятны пользователю.

5. *Гибкость.* Насколько хорошо интерфейс системы может обслуживать пользователя с различными уровнями подготовки? Для неопытных пользователей интерфейс может быть организован как иерархическая структура меню, а для опытных пользователей как команды, комбинации нажатий клавиш и параметры.

Стандартный графический интерфейс пользователя должен отвечать ряду требований:

- поддерживать информационную технологию работы пользователя с программным продуктом – содержать привычные и понятные пользователю пункты меню, соответствующие функциям

обработки, расположенные в естественной последовательности использования;

- ориентироваться на конечного пользователя, который общается с программой на *внешнем* уровне взаимодействия;
- удовлетворять правилу «шести» – в одну линейку меню включать не более 6 понятий, каждое из которых содержит не более 6 опций;
- графические объекты сохраняют свое стандартизованное назначение и по возможности местоположение на экране.

3.3. Средства управления графического интерфейса пользователя

К графическому интерфейсу пользователя предъявляются высокие требования как с чисто инженерной, так и с художественной стороны разработки, при его разработке ориентируются на возможности человека. Наиболее часто графический интерфейс реализуется в интерактивном режиме работы пользователя и строится в виде системы спускающихся *меню* с использованием в качестве средства манипуляции мыши и клавиатуры. Работа пользователя осуществляется с *экранными формами,* содержащими *объекты управления, панели инструментов* с *пиктограммами* режимов и команд обработки.

К числу типовых объектов управления интерфейса относятся:

- метка (label) – постоянный текст, не подлежащий изменению при работе пользователя с экранной формой (например, слова *Фамилия Имя Отчество);*
- текстовое окно (text box) – используется для ввода информации произвольного вида, отображения хранимой информации в базе данных (например, для ввода фамилии студента);
- рамка (frame) – объединение объектов управления в группу по функциональному или другому принципу (например, для изменения их параметров);
- командная кнопка (command button) – обеспечивает передачу управляющего воздействия, например, кнопки <*Cance1*>, <*ОК*>, <*0тмена*>; выбор режима обработки типа <*Ввод*>, <*Удаление*>, <*Редактирование*>, <*Выход*> и др.;
- кнопка-переключатель <*option button*> – для альтернативного выбора кнопки из группы однотипных кнопок (например, *семейное положение);*
- помечаемая кнопка <check button> – для аддитивного выбора несколько кнопок из группы однотипных кнопок (например, *факультатив для посещения);*

- окно-список (list box) – содержит список альтернативных значений для выбора (например, «Спортивная секция»);
- комбинированное окно (combo box) – объединяет возможности окна-списка и текстового окна (например, «Предметы по выбору» – можно указать новый предмет или выбрать один из предлагаемого списка);
- линейка горизонтальной прокрутки – для быстрого перемещения внутри длинного списка или текста по горизонтали;
- линейка вертикальной прокрутки – для быстрого перемещения внутри длинного списка или текста по вертикали;
- окно-список каталогов;
- окно-список накопителей;
- окно-список файлов и др.

Управление – общий термин для компонентов интерфейса типа слайдеров, кнопок, кадров (фреймов), переключателей и т.д., которые служат, чтобы заместить объекты, являющиеся знакомыми пользователям из реального мира.

Кнопки используются, чтобы выбрать опцию или вызвать событие (например, запуск подпрограммы).

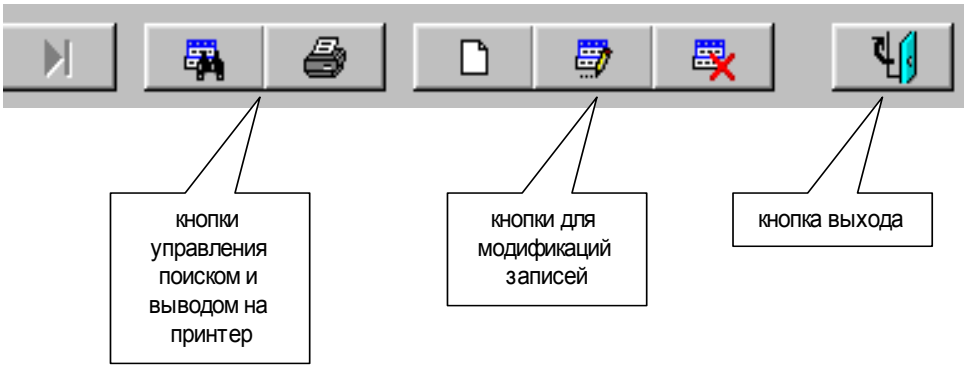


Рис. 3.1. Пример кнопок из разрабатываемой системы

Переключатели подобны кнопкам выбора, в которых пользователь выбирает значение из фиксированного списка.

Слайдеры – обычно это элемент, «полоса прокрутки», они могут быть помещены или в горизонтальную или вертикальную линейку на экране.

Метки и текстовые блоки используются для текстовой информации. Различие между ними – текстовые поля, которые позволяют пользователю вводить текстовые данные в поля, в то время как метки – поля, не редактируемые, используемые только для отображения текста типа подсказок, команд пользователя и т.д.

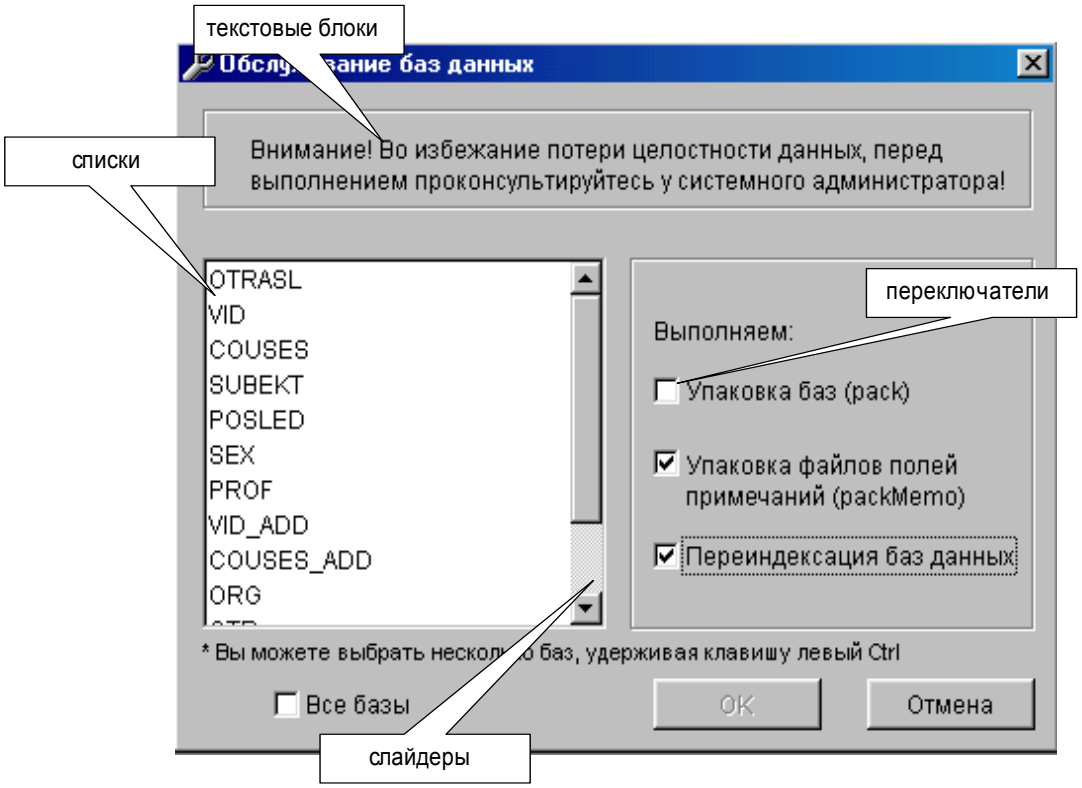


Рис. 3.2. Пример средств графического интерфейса пользователя

Списки – специализированные средства управления, которые отображают раскрывающиеся списки значений (часто с присоединенными слайдерами, чтобы перемещаться вверх или вниз по списку) и позволяют пользователю выбирать значение из списка, или вводить другое значение в присоединенное текстовое поле. Списки – удобный и компактный элемент интерфейса, который занимает минимум места на экране и в то же время несет большую информационную нагрузку (рис. 3.2).

Изображения (Иконки)

В интерфейсе непосредственного манипулирования пользователи выполняют действия непосредственно на видимых объектах. Этими объектами могут быть кнопки, метки, меню или изображения (иконки). Все иконки можно классифицировать согласно тому, насколько точно они отображают несущую функцию:

- *иконки Подобия* – иконки похожи на объекты, которые они отображают (типа ножниц, чтобы отобразить операцию «вырезки» на рис. 3.3);

- *иконки по образцу* – представляют пример типа объекта (например, иконкой, показывающей линию, чтобы представить средство рисования, представленной на рис. 3.4);

Рис. 3.3. Пример иконки подобия

Рис. 3.4. Пример иконки по образцу

- *символические иконки* используются, чтобы представить действие или состояние в символической форме (например, разорванная линия между двумя компьютерами для того, чтобы показать разорванное сетевое соединение);
- *произвольные иконки* не несут никакой информации по поводу их представления, поэтому их назначение должно быть описано (например, обратная круговая стрелка, чтобы представить действие «отмена последней команды»).

Меню

Необходимый элемент автоматизированной системы – меню, позволяющее пользователю выполнять задачи внутри приложения и управлять процессом решения. *Меню* – набор опций, отображаемых на экране, где пользователи могут выбирать и выполнять действия, тем самым производя изменения в состоянии интерфейса. Достоинство меню в том, что пользователи не должны помнить название элемента или действия, которое они хотят выполнить – они должны только распознать его среди пунктов меню. Таким образом, меню может использовать даже неопытный пользователь. Однако проект меню должен быть тщательно продуман. Чтобы меню было эффективным, названия пунктов меню должны быть очевидными.

Меню может занимать много экранного места, но есть решение для этой проблемы – использование всплывающего или ниспадающего меню. При нажатии на иконку, строку меню или другой объект вызывается всплывающее или ниспадающее меню.

Основные принципы создания меню

В процессе проектирования системы меню приложения необходимо принять наилучший способ отображения меню, чтобы оно было понятно и легко в использовании. Обычно команды меню упорядочены некоторым иерархическим способом. Основная проблема состоит в том, чтобы правильно распределить различные пункты меню по различным уровням и правильно их сгруппировать. Исследования показывают, что имеются четыре варианта для организации меню:

- алфавитный;

- категоричный;
- в соответствии с нормальными соглашениями;
- в соответствии с частотой использования.

Принципы проектирования меню:

- структура меню должна соответствовать структуре решаемой системой задачи, организация меню должна отразить наиболее эффективную последовательность шагов, чтобы достичь решения поставленной задачи;
- пункты меню должны быть краткими, грамматически правильными и соответствовать своему заголовку в меню. Порядок пунктов меню выбирается согласно соглашению, частоте использования, порядку использования, в зависимости от потребностей задачи или пользователя;
- выбор пунктов меню должен быть обеспечен несколькими способами – с помощью клавиатуры, с помощью мыши, а также через другие объекты пользовательского интерфейса. Необходимо использовать легко запоминаемые сочетания клавиш для более быстрого доступа к пунктам меню, поскольку это очень экономит время.

Формы

Формы – основной элемент интерфейса. Назначение форм – удобный ввод и просмотр данных, состояния, сообщений автоматизированной системы.

Основные *принципы проектирования форм*:

- форма проектируется для более удобного, более понятного и скорейшего достижения решения поставленной задачи. Если форма переносится из бумажной формы, то передвижение по смежным полям не должно вызывать затруднений у пользователя;
- размещение информационных единиц на пространстве формы должно соответствовать логике ее будущего использования: это зависит от необходимой последовательности доступа к информационным единицам, частоты их использования, а также от относительной важности элементов;
- важно использовать незаполненное пространство, чтобы создать равновесие и симметрию среди информационных элементов формы, для фиксации внимания пользователя в нужном направлении;
- логические группы элементов необходимо отделять пробелами, строками, цветовыми или другими визуальными средствами;

- взаимозависимые или связанные элементы должны отображаться в одной форме.

Дизайн заголовков и полей

Для отдельных полей заголовок должен быть выровнен по левому краю; для полей списков заголовок должен быть выше и левее по отношению к основному полю, числовые поля выравниваются по правому полю.

Длинные колоночные поля или длинные столбцы информационных единиц с одиночными полями необходимо объединять в группы пять элементов, разделяемых пустой строкой, – это помогает пользователю мысленно обрабатывать информацию по выделенным группам.

В формах с большим количеством информации необходимо использовать названия разделов, которые однозначно свидетельствуют о характере принадлежащей им информации.

Необходимо четко разделить отображение заголовков и непосредственно полей ввода, поскольку такая путаница может вызвать дискомфорт у пользователя.

Заголовки должны быть краткими, знакомыми и содержательными.

Поля, необязательные для заполнения либо не имеющие особой важности, должны отличаться визуально (цветом или другими эффектами) от полей важных и обязательных для заполнения.

Тексты и диалоги

Приведем некоторые принципы, которыми необходимо руководствоваться при создании текстовых диалогов и отображений:

- текст в нижнем регистре читается приблизительно на 13 % быстрее, чем текст, который напечатан полностью в верхнем регистре;
- символы верхнего регистра наиболее эффективны для информации, которая должна привлечь внимание. НЕ ИСПОЛЬЗУЙТЕ ВЕРХНИЙ РЕГИСТР, ЕСЛИ ВЫ НЕ ХОТИТЕ ВЫДЕЛЯТЬ КАКУЮ-ЛИБО ИНФОРМАЦИЮ;
- выровненный по правому краю текст труднее читать, чем равномерно распределенный текст с невыровненным правым полем;
- оптимальный интервал между строками равен или немного больше, чем высота символов.

Панель задач

Переход из одного запущенного приложения в другое в Windows XP осуществляется через панель задач (рис. 3.5).

Рис. 3.5. Пример панели задач в Windows XP

В операционной системе Linux интерфейс рабочего стола позволяет иным способом переключать открытые окна (рис 3.6). ОС Linux имеет несколько рабочих столов.

Рис. 3.6. Пример интерфейс рабочего стола ОС Linux

3.4. Общие принципы проектирования интерфейса

Существует ряд навигационных средств и приемов, которые помогают пользователю ориентироваться в системе. Они включают: использование заголовков страниц для каждого экрана; использование номеров страниц; номеров строк и столбцов; отображение текущего имени файла вверху экрана. Тип системы навигации зависит от принятого стиля интерфейса. Для интерфейсов языка команд очень мало способов обеспечения полноценной навигации. В интерфейсах с меню можно использовать иерархически – структурированное меню. Для выхода из подменю нужно применять не сложные действия. Диалоговые интерфейсы сами по себе защищают пользователя от ошибочных действий. Информация Состояния обычно отображается внизу экрана и содержит в себе данные количеств записей, числе обработанных единиц, процессе печати, очереди печати и т.д.

Проектирование сообщений

Сообщения необходимы для направления пользователя в нужную сторону, подсказок и предупреждений для выполнения необходимых действий на пути решения задачи. Они также включают подтверждения действий со стороны пользователя и подтверждения, что задачи были выполнены системой успешно либо по каким-то причинам не выполнены. Сообщения могут быть обеспечены в форме диалога, экранных заставок и т.п.

Сообщения могут предложить пользователю:

- выбрать из предложенных альтернатив некую опцию или набор опций;
- ввести некоторую информацию;
- выбрать опцию из набора опций, которые могут изменяться в зависимости от текущего контекста;
- подтвердите фрагмент введенной информации перед продолжением ввода.

Сообщения могут быть помещены в модальные диалоговые окна, которые вынуждают пользователя ответить на вопрос прежде, чем может быть предпринято любое другое действие, потому что все другие средства управления заморожены. Это может быть полезно, когда система должна вынудить пользователя принять решение перед продолжением работы. Немодальные диалоговые окна позволяют работать с другими элементами интерфейса, в то время как само окно может игнорироваться.

Обработка ошибок в формах ввода

Основные принципы:

- обеспечить возможность посимвольного редактирования введенных записей для исправления ошибок ввода (опечаток);
- если ошибка была обнаружена системой, желательно вернуть курсор в поле с ошибочными данными и каким-либо образом выделить это поле визуально;
- обеспечить значимые сообщения об ошибках, использующие стиль языка пользователя и соответствующую терминологию;
- обеспечить сообщения об ошибках, которые объясняют и предлагают пути ее устранения.

3.5. Диалоговый режим

Большинство программных продуктов, особенно прикладного характера, ориентированных на конечного пользователя, работают в *диалоговом режиме* взаимодействия с пользователем таким образом,

что ведется обмен *сообщениями,* влияющими на обработку данных. В диалоговом режиме под воздействием пользователя осуществляются запуск функций (методов) обработки, изменение свойств объектов, производится настройка параметров выдачи информации на печать и т.п.

Системы, поддерживающие диалоговые процессы, классифицируются следующим образом:

- системы с *жестким сценарием диалога* – стандартизированное представление информации обмена;
- *дескрипторные системы* – формат ключевых слов сообщений;
- *тезаурусные системы* – семантическая сеть дескрипторов, образующих словарь системы (аналог – гипертекстовые системы);
- системы с *языком деловой прозы* – представление сообщений на языке, естественном для профессионального пользования.

Наиболее просты для реализации и распространены диалоговые системы с жестким сценарием диалога, которые представлены в виде:

- *меню,* когда диалог инициируется программой; пользователю предлагается выбор альтернативы функций обработки из фиксированного перечня; предоставляемое меню может быть иерархическим и содержать вложенные подменю следующего уровня;
- действия *запрос-ответ,* когда фиксирован перечень возможных значений, выбираемых из списка, или ответы *типа Да/Нет;*
- *запрос по формату,* в нем с помощью ключевых слов, фраз или путем заполнения экранной формы с регламентированным по составу и структуре набором реквизитов осуществляется подготовка сообщений.

Диалоговый процесс управляется согласно созданному *сценарию,* для которого определяются:

- точки (момент, условие) начала диалога;
- инициатор диалога – человек или программный продукт;
- параметры и содержание диалога – сообщения, состав и структура меню, экранные формы и т.п.;
- реакция программного продукта на завершение диалога.

Описание сценария диалога выполняют:

- *блок-схема,* в которой предусмотрены блоки выдачи сообщений и обработки полученных ответов;
- *ориентированный граф,* вершины которого – сообщения и выполняемые действия, дуги – связь сообщений; словесное описание;

- специализированные объектно-ориентированные языки построения сценариев.

Средства создания диалоговых процессов интерфейса пользователя

Для создания диалоговых процессов и интерфейса конечного пользователя наиболее применимы *объектно-ориентированные инструментальные средства разработки программ.* Например, в составе инструментальных средств СУБД содержатся *построители меню,* с помощью которых создается ориентированная на конечного пользователя совокупность режимов команд в виде *главного меню* и *вложенных подменю.* В ряде СУБД и электронных таблиц, текстовых редакторов существуют различные типы *диалоговых окон,* содержащих разнообразные объекты управления, перечисленные выше. В среде электронных таблиц и текстовых редакторов имеются возможности настройки главных меню (удаление ненужных, добавление новых режимов и команд), создания системы подсказок с помощью встроенных средств и языков программирования.

Контрольные вопросы и задания

1. В чём суть человеко-машинного взаимодействия?
2. Объясните понятие «Интерфейс пользователя».
3. Назовите основные виды интерфейса пользователя.
4. Основные принципы создания интерфейса пользователя.
5. Назовите средства управления графического интерфейса пользователя.
6. Общие принципы проектирования интерфейса пользователя.
7. Принципы проектирования меню и форм.
8. Дайте понятие «Диалоговый режим работы».

4. ОПЕРАЦИОННЫЕ СИСТЕМЫ: ИСТОРИЯ РАЗВИТИЯ, КЛАССИФИКАЦИЯ, ФУНКЦИИ

4.1. Эволюция операционных систем

История ОС насчитывает примерно полвека. Огромное влияние на развитие операционных систем оказали успехи в совершенствовании элементной базы и вычислительной аппаратуры, поэтому многие этапы развития ОС тесно связаны с появлением новых типов аппаратных платформ, таких, как мини-компьютеры или персональные компьютеры. Серьезную эволюцию операционные системы претерпели в связи с новой ролью компьютеров в локальных и глобальных сетях. Важнейшим фактором развития ОС стал Интернет. По мере того как эта Сеть приобретает черты универсального средства массовых коммуникаций, ОС становятся все более простыми и удобными в использовании, включают развитые средства поддержки мультимедийной информации, снабжаются надежными средствами защиты.

Появление первых операционных систем

Настоящее рождение цифровых вычислительных машин произошло вскоре после окончания Второй мировой войны. В середине 1940-х годов были созданы первые ламповые вычислительные устройства. В то время одна и та же группа людей участвовала и в проектировании, и в эксплуатации, и в программировании вычислительной машины. Это была скорее научно-исследовательская работа в области вычислительной техники, а не использование компьютеров в качестве инструмента решения каких-либо практических задач из других прикладных областей. Программирование осуществлялось исключительно на машинном языке. Не было никакого системного программного обеспечения, кроме библиотек математических и служебных подпрограмм, которые программист мог использовать для того, чтобы не писать каждый раз коды, вычисляющие значение какой-либо математической функции или управляющие стандартным устройством ввода–вывода.

Операционные системы все еще не появились, все задачи организации вычислительного процесса решались вручную каждым программистом с пульта управления, который представлял собой примитивное устройство ввода–вывода, состоящее из кнопок, переключателей и индикаторов. С середины 1950-х годов начался

новый период в развитии вычислительной техники, связанный с появлением новой технической базы – полупроводниковых элементов.

Первые цифровые вычислительные машины работали без операционных систем, все задачи организации вычислительного процесса решались вручную каждым программистом с пульта управления.

Наряду с совершенствованием аппаратуры заметный прогресс наблюдался также в области автоматизации программирования и организации вычислительных работ. В эти годы появились первые алгоритмические языки, и таким образом к библиотекам математических и служебных подпрограмм добавился *новый тип системного программного* обеспечения – *трансляторы.*

Для организации эффективного совместного использования трансляторов, библиотечных программ и загрузчиков в штат многих вычислительных центров были введены должности операторов. Но как бы быстро и надежно ни работали операторы, они никак не могли состязаться в производительности с работой устройств компьютера. Большую часть времени процессор простаивал в ожидании, пока оператор запустит очередную задачу. А поскольку процессор представлял собой весьма дорогое устройство, то низкая эффективность его использования означала низкую эффективность использования компьютера в целом.

Для решения этой проблемы были разработаны *первые системы пакетной обработки* (середины 1950-х годов), которые автоматизировали всю последовательность действий оператора по организации вычислительного процесса. Ранние системы пакетной обработки явились прообразом современных операционных систем, они стали первыми системными программами, предназначенными не для обработки данных, а для управления вычислительным процессом.

Оператор составлял пакет заданий, которые в дальнейшем без его участия последовательно запускались на выполнение управляющей программой – монитором.

Пакет обычно представлял собой набор перфокарт, но для ускорения работы он мог переноситься на более удобный и емкий носитель, например на магнитную ленту или магнитный диск.

Однако при этом программисты-пользователи лишились непосредственного доступа к компьютеру, что снижало эффективность их работы, – внесение любого исправления требовало значительно больше времени, чем при интерактивной работе за пультом машины.

Появление мультипрограммных операционных систем для мэйнфреймов

Следующий важный период развития операционных систем относится к 1965–1975 годам. В это время в технической базе вычислительных машин произошел переход от отдельных полупроводниковых элементов типа транзисторов к интегральным микросхемам, что открыло путь к появлению следующего поколения компьютеров. Большие функциональные возможности интегральных схем сделали возможным реализацию на практике сложных компьютерных архитектур, таких, например, как IBM/360.

В этот период были реализованы практически все основные механизмы, присущие современным ОС: *мультипрограммирование, мультипроцессирование, поддержка многотерминального многопользовательского режима, виртуальная память, файловые системы, разграничение доступа и сетевая работа.*

В эти годы начинается расцвет системного программирования. Из направления прикладной математики, представляющго интерес для узкого круга специалистов, системное программирование превращается в отрасль индустрии, оказывающую непосредственное влияние на практическую деятельность миллионов людей. Революционным событием данного этапа явилась промышленная реализация *мультипрограммирования* – способа организации вычислительного процесса, при котором в памяти компьютера находилось одновременно несколько программ, попеременно выполняющихся на одном процессоре.

Вариант мультипрограммирования, применяемый в системах разделения времени, был нацелен на создание для каждого отдельного пользователя иллюзии единоличного владения вычислительной машиной за счет периодического выделения каждой программе своей доли процессорного времени. В системах разделения времени при *многотерминальном режиме* эффективность использования оборудования ниже, чем в системах пакетной обработки, что явилось платой за удобства работы пользователя. Терминальные комплексы могли располагаться на большом расстоянии от процессорных стоек, соединяясь с ними с помощью различных глобальных связей, – модемных соединений телефонных сетей или выделенных каналов. Для поддержания удаленной работы терминалов в операционных системах появились специальные программные модули, реализующие различные (в то время, как правило, нестандартные) протоколы связи. Такие вычислительные системы с удаленными терминалами, сохраняя централизованный характер обработки данных, в какой-то степени

являлись прообразом современных сетей, а соответствующее системное программное обеспечение – прообразом сетевых ОС.

К этому времени можно констатировать существенное изменение в распределении функций между аппаратными и программными средствами компьютера. ОС становились неотъемлемыми элементами компьютеров, играя роль «продолжения» аппаратуры. В первых вычислительных машинах программист, напрямую взаимодействуя с аппаратурой, мог выполнить загрузку программных кодов, используя пультовые переключатели и лампочки индикаторов, а затем вручную запустить программу на выполнение, нажав кнопку «пуск». В компьютерах 1960-х годов большую часть действий по организации вычислительного процесса взяла на себя ОС. Реализация мультипрограммирования потребовала внесения очень важных изменений в аппаратуру компьютера, непосредственно направленных на поддержку нового способа организации вычислительного процесса. В процессорах появился *привилегированный и пользовательский режим* работы, специальные регистры для быстрого переключения с одной программы на другую, средства защиты областей памяти, а также развитая *система прерываний.*

Еще одной важной тенденцией этого периода является *создание семейств программно-совместимых машин* и операционных систем для них. Примерами семейств программно-совместимых машин, построенных на интегральных микросхемах, являются серии машин IBM/360 и IBM/370. Вскоре идея программно-совместимых машин стала общепризнанной. ОС, построенные с намерением удовлетворить всем требованиям, оказались чрезвычайно сложными. Они состояли из многих миллионов ассемблерных строк, написанных тысячами программистов, и содержали тысячи ошибок, вызывающих нескончаемый поток исправлений. ОС этого поколения были очень дорогими. Так, разработка OS/360, объем кода, для которой составил 8 Мбайт, стоила компании IBM 80 миллионов долларов.

Операционные системы и глобальные сети

В начале 1970-х годов появились первые *сетевые операционные системы*, которые, в отличие от многотерминальных ОС, позволяли не только рассредоточить пользователей, но и организовать распределенное хранение и обработку данных между несколькими компьютерами, связанными электрическими связями. Любая сетевая ОС, с одной стороны, выполняет все функции локальной ОС, а с другой стороны, обладает некоторыми дополнительными средствами,

позволяющими ей взаимодействовать по сети с операционными системами других компьютеров. Программные модули, реализующие сетевые функции, появлялись в ОС постепенно, по мере развития сетевых технологий, аппаратной базы компьютеров и возникновения новых задач, требующих сетевой обработки.

В 1969 году Министерство обороны США инициировало работы по объединению суперкомпьютеров оборонных и научно-исследовательских центров в единую сеть. Эта сеть получила название ARPANET и явилась отправной точкой для создания самой известной ныне глобальной сети – *Интернета*. В 1974 году компания IBM объявила о создании собственной сетевой архитектуры для своих мэйнфреймов, получившей название SNA (System Network Architecture). Эта многоуровневая архитектура, во многом подобная стандартной модели OSI. Функции верхних уровней SNA выполнялись программными модулями. Один из них составлял основу программного обеспечения процессора телеобработки. Другие модули работали на ЦП в составе стандартной ОС IBM для мэйнфреймов.

В это же время в Европе велись активные работы по созданию и стандартизации *сетей X.25*. Эти сети с коммутацией пакетов не были привязаны к какой–либо конкретной операционной системе. После получения статуса международного стандарта в 1974 году протоколы X.25 стали поддерживаться многими операционными системами. С 1980 года компания IBM включила поддержку протоколов X.25 в архитектуру SNA и в свои ОС.

Важной вехой в истории мини-компьютеров и вообще в истории операционных систем явилось создание *ОС UNIX*. Первоначально эта ОС предназначалась для поддержания режима разделения времени в мини-компьютере PDP-7. С середины 70-х годов началось массовое использование ОС UNIX. К этому времени программный код для UNIX был на 90 % написан на языке высокого уровня С. Широкое распространение эффективных С-компиляторов сделало UNIX уникальной для того времени ОС, обладающей возможностью сравнительно легкого переноса на различные типы компьютеров. Поскольку эта ОС поставлялась вместе с исходными кодами, то она стала первой открытой ОС, которую могли совершенствовать простые пользователи-энтузиасты. Хотя UNIX была первоначально разработана для мини-компьютеров, гибкость, элегантность, мощные функциональные возможности и открытость позволили ей занять прочные позиции во всех классах компьютеров: суперкомпьютерах, мэйнфреймах, мини-компьютерах, серверах и рабочих станциях на базе RISC-процессоров, персональных компьютерах.

Первые локальные сети строились с помощью нестандартного коммуникационного оборудования, в простейшем случае – путем прямого соединения последовательных портов компьютеров. Программное обеспечение также было нестандартным и реализовывалось в виде пользовательских приложений. Первое сетевое приложение для ОС UNIX – программа UUCP (UNIX–to–UNIX Copy program) – появилась в 1976 году и начала распространяться с версией 7 AT&T UNIX с 1978 года. Эта программа позволяла копировать файлы с одного компьютера на другой в пределах локальной сети через различные аппаратные интерфейсы, могла работать через глобальные связи, например модемные.

Развитие операционных систем в 80-е годы

К наиболее важным событиям этого десятилетия можно отнести разработку *стека TCP/IP,* становление Интернета, стандартизацию технологий локальных сетей, появление персональных компьютеров и ОС для них.

Рабочий вариант стека протоколов TCP/IP был создан в конце 1970-х годов. Этот стек представлял собой набор общих протоколов для разнородной вычислительной среды и предназначался для связи экспериментальной сети ARPANET с другими «сателлитными» сетями. В 1983 году стек протоколов TCP/IP был принят Министерством обороны США в качестве военного стандарта. Переход компьютеров сети ARPANET на стек TCP/IP ускорила его реализация для операционной системы BSD UNIX. С этого времени началось совместное существование UNIX и протоколов TCP/IP, а практически все многочисленные версии Unix стали сетевыми.

Все десятилетие было отмечено постоянным появлением новых, все более совершенных версий ОС UNIX. Среди них были и фирменные версии UNIX: SunOS, HP–UX, Irix, AIX и многие другие, в которых производители компьютеров адаптировали код ядра и системных утилит для своей аппаратуры. Разнообразие версий породило проблему их совместимости, которую периодически пытались решить различные организации. В результате были приняты стандарты POSIX и XPG, определяющие интерфейсы ОС для приложений, а специальное подразделение компании AT&T выпустило несколько версий UNIX System III и UNIX System V, призванных консолидировать разработчиков на уровне кода ядра.

Начало 1980-х годов связано с еще одним знаменательным для истории ОС событием – появлением персональных компьютеров. Компьютеры стали широко использоваться неспециалистами, что

потребовало разработки *«дружественного»* программного обеспечения, и предоставление этих «дружественных» функций стало прямой обязанностью ОС. Персональные компьютеры послужили также мощным катализатором для бурного роста локальных сетей, создав для этого отличную материальную основу в виде десятков и сотен компьютеров, принадлежащих одному предприятию и расположенных в пределах одного здания. В результате поддержка сетевых функций стала для ОС ПК необходимым условием.

Однако и *дружественный интерфейс*, и сетевые функции появились у ОС ПК не сразу. Первая версия наиболее популярной ОС раннего этапа развития персональных компьютеров – *MS-DOS* компании Microsoft была однопрограммная, однопользовательская ОС с интерфейсом командной строки, способная стартовать с дискеты. Основными задачами для нее были управление файлами, расположенными на гибких и жестких дисках в UNIX подобной иерархической файловой системе, а также поочередный запуск программ. MS-DOS не была защищена от программ пользователя, так как процессор Intel 8088 не поддерживал привилегированного режима. Разработчики первых ПК считали, что при индивидуальном использовании компьютера и ограниченных возможностях аппаратуры нет смысла в поддержке мультипрограммирования, поэтому в процессоре не были предусмотрены привилегированный режим и другие механизмы поддержки мультипрограммных систем.

Недостающие функции для MS-DOS и подобных ей ОС компенсировались внешними программами, предоставлявшими пользователю удобный графический интерфейс (например, Norton Commander) или средства тонкого управления дисками (например, PC Tools). Наибольшее влияние на развитие программного обеспечения для персональных компьютеров оказала *операционная среда Windows* компании Microsoft, представлявшая собой *надстройку над MS-DOS*.

Сетевые функции также реализовывались в основном сетевыми оболочками, работавшими поверх ОС. При сетевой работе всегда необходимо поддерживать многопользовательский режим, при котором один пользователь − интерактивный, а остальные получают доступ к ресурсам компьютера по сети. В таком случае от ОС требуется хотя бы некоторый минимум функциональной поддержки многопользовательского режима. История сетевых средств MS-DOS началась с версии 3.1. Эта версия MS-DOS добавила к файловой системе необходимые средства блокировки файлов и записей, которые позволили более чем одному пользователю иметь доступ к файлу.

Пользуясь этими функциями, сетевые оболочки могли обеспечить разделение файлов между сетевыми пользователями.

Вместе с выпуском версии MS-DOS 3.1 в 1984 году компания Microsoft также выпустила продукт, называемый Microsoft Networks, который обычно неформально называют MS-NET. Некоторые концепции, заложенные в MS-NET, такие как введение в структуру базовых сетевых компонентов – редиректора и сетевого сервера, успешно перешли в более поздние сетевые продукты Microsoft: *LAN Manager*, *Windows for Workgroups*, а затем и в *Windows NT.*

Сетевые оболочки для ПК выпускали и другие компании: IBM, Artisoft, Performance Technology. Иной путь выбрала компания Novell. Она изначально сделала ставку на разработку ОС со встроенными сетевыми функциями и добилась на этом пути выдающихся успехов. Ее сетевые ОС NetWare на долгое время стали эталоном производительности, надежности и защищенности для ЛВС.

Первая сетевая ОС компании Novell появилась на рынке в 1983 году и называлась *OS-Net.* Эта ОС предназначалась для сетей, имевших звездообразную топологию, центральным элементом которых был специализированный компьютер на базе микропроцессора Motorola 68000. Позже IBM выпустила персональные компьютеры PC XT, компания Novell разработала новый продукт – *NetWare 86,* рассчитанный на архитектуру микропроцессоров семейства Intel 8088.

С самой первой версии ОС NetWare распространялась как ОС для центрального сервера ЛВС, которая за счет специализации на выполнении функций файл-сервера обеспечивает максимально возможную для данного класса компьютеров скорость удаленного доступа к файлам и повышенную безопасность данных. За высокую производительность пользователи сетей Novell NetWare расплачиваются стоимостью – выделенный файл-сервер не может использоваться в качестве рабочей станции, а его специализированная ОС имеет весьма специфический прикладной программный интерфейс (API). В отличие от Novell большинство других компаний развивали сетевые средства для персональных компьютеров в рамках операционных систем с универсальным интерфейсом API, то есть ОС общего назначения. Такие системы по мере развития аппаратных платформ ПК стали все больше приобретать черты ОС мини-компьютеров.

В 1987 году в результате совместных усилий Microsoft и IBM появилась первая многозадачная операционная система для персональных компьютеров с процессором Intel 80286, в полной мере

использующая возможности защищенного режима – *OS/2*. Эта система была хорошо продумана. Она поддерживала вытесняющую многозадачность, виртуальную память, графический пользовательский интерфейс (не с первой версии) и виртуальную машину для выполнения DOS-приложений. Фактически она выходила за пределы простой многозадачности с ее концепцией распараллеливания отдельных процессов, получившей название многопоточности.

OS/2 с ее развитыми функциями многозадачности и файловой системой HPFS со встроенными средствами многопользовательской защиты оказалась хорошей платформой для построения локальных сетей персональных компьютеров. Наибольшее распространение получили сетевые оболочки LAN Manager компании Microsoft и LAN Server компании IBM, разработанные этими компаниями на основе одного базового кода. Эти оболочки уступали по производительности файловому серверу NetWare и потребляли больше аппаратных ресурсов, но имели важные достоинства: они позволяли, во-первых, выполнять на сервере любые программы, разработанные для OS/2, MS–DOS и Windows, а во-вторых, использовать компьютер, на котором они работали, в качестве рабочей станции.

Сетевые разработки компаний Microsoft и IBM привели к появлению *NetBIOS* – очень популярного транспортного протокола и одновременно интерфейса прикладного программирования для локальных сетей, получившего применение практически во всех сетевых операционных системах для персональных компьютеров. Этот протокол и сегодня применяется для создания небольших локальных сетей. Не очень удачная рыночная судьба OS/2 не позволила системам LAN Manager и LAN Server захватить заметную долю рынка, но принципы работы этих сетевых систем во многом нашли свое воплощение в более удачливой операционной системе 90-х годов – Microsoft Windows NT, содержащей встроенные сетевые компоненты, некоторые из которых имеют приставку LM – от LAN Manager.

В 1980-е годы были приняты основные стандарты на коммуникационные технологии для локальных сетей: в 1980 году – *Ethernet*, в 1985 – *Token Ring*, в конце 80-х – *FDDI.* Это позволило обеспечить совместимость сетевых операционных систем на нижних уровнях, а также стандартизовать интерфейс ОС с драйверами сетевых адаптеров. Для ПК применялись не только специально разработанные для них ОС, подобные MS–DOS, NetWare и OS/2, но и адаптировались уже существующие ОС. Появление процессоров Intel 80286 и особенно 80386 с поддержкой мультипрограммирования позволило перенести на

платформу ПК ОС UNIX. Наиболее известной системой этого типа была версия UNIX компании Santa Cruz Operation (SCO UNIX).

Особенности современного этапа развития операционных систем

В 90-е годы практически *все ОС*, занимающие заметное место на рынке, *стали сетевыми*. Сетевые функции сегодня встраиваются в ядро ОС, являясь ее неотъемлемой частью. Операционные системы получили средства для работы со всеми основными технологиями локальных (Ethernet, Fast Ethernet, Gigabit Ethernet, Token Ring, FDDI, ATM) и глобальных (X.25, frame relay, ISDN, ATM) сетей, а также средства для создания составных сетей (IP, IPX, AppleTalk, RIP, OSPF, NLSP). В ОС используются средства мультиплексирования нескольких стеков протоколов, за счет которого компьютеры могут поддерживать одновременную сетевую работу с разнородными клиентами и серверами. Появились специализированные ОС, которые предназначены исключительно для выполнения коммуникационных задач. Например, сетевая операционная система *IOS* компании Cisco Systems, работающая в маршрутизаторах, организует в мультипрограммном режиме выполнение набора программ, каждая из которых реализует один из коммуникационных протоколов.

К началу 2000 года все производители ОС резко усилили поддержку средств работы с Интернетом (кроме производителей UNIX-систем, в которых эта поддержка всегда была существенной). Кроме самого стека TCP/IP, в комплект поставки начали включать утилиты, реализующие такие популярные сервисы Интернета, как telnet, ftp, DNS и Web. Влияние Интернета проявилось и в том, что компьютер превратился из чисто вычислительного устройства в *средство коммуникаций с развитыми вычислительными возможностями*.

Особое внимание сегодня уделяется *корпоративным* сетевым ОС. Их дальнейшее развитие представляет одну из наиболее важных задач и в обозримом будущем. Корпоративная ОС отличается способностью хорошо и устойчиво работать в крупных сетях, которые характерны для больших предприятий, имеющих отделения в десятках городов и, возможно, в разных странах. Таким сетям органически присуща высокая степень гетерогенности программных и аппаратных средств, поэтому корпоративная ОС должна беспроблемно взаимодействовать с операционными системами разных типов и работать на различных аппаратных платформах. В начале 21 века достаточно явно определилась тройка лидеров в классе корпоративных ОС – это Novell NetWare 5.0, Microsoft Windows NT 4.0 и Windows 2000, а также UNIX-системы различных производителей аппаратных платформ.

Для корпоративной ОС очень важно наличие средств централизованного администрирования и управления, позволяющих в единой базе данных хранить учетные записи о десятках тысяч пользователей компьютеров, коммуникационных устройств и модулей программного обеспечения, имеющихся в корпоративной сети. В современных операционных системах средства централизованного администрирования обычно базируются на единой справочной службе. Первой успешной реализацией справочной службы корпоративного масштаба была система StreetTalk компании Banyan. К настоящему времени наибольшее признание получила справочная служба NDS компании Novell, выпущенная впервые в 1993 году для первой корпоративной версии NetWare 4.O. Роль централизованной справочной службы настолько велика, что именно по качеству справочной службы оценивают пригодность операционной системы для работы в корпоративном масштабе. Длительная задержка выпуска Windows NT 2000 во многом была связана с созданием для этой ОС масштабируемой справочной службы Active Directory, без которой этому семейству ОС трудно было претендовать на звание истинно корпоративной ОС.

Создание многофункциональной масштабируемой справочной службы является стратегическим направлением эволюции ОС. От успехов этого направления во многом зависит и дальнейшее развитие Интернета. Такая служба нужна для превращения Интернета в предсказуемую и управляемую систему, например для обеспечения требуемого качества обслуживания трафика пользователей, поддержки крупных распределенных приложений, построения эффективной почтовой системы и т. п.

На современном этапе развития операционных систем на первый план вышли средства обеспечения безопасности. Это связано с возросшей ценностью информации, обрабатываемой компьютерами, а также с повышенным уровнем угроз, существующих при передаче данных по сетям, особенно по публичным, таким, как Интернет. Многие операционные системы обладают сегодня развитыми средствами защиты информации, основанными на шифрации данных, аутентификации и авторизации. Современным операционным системам присуща *многоплатформенностъ*, то есть способность работать на совершенно различных типах компьютеров. Многие операционные системы имеют специальные версии для поддержки *кластерных* архитектур, обеспечивающих высокую производительность и отказоустойчивость. Исключением пока является ОС NetWare, все версии которой разработаны для платформы Intel, а реализации

функций NetWare в виде оболочки для других ОС, например NetWare for AIX, успеха не имели.

В последние годы получила дальнейшее развитие долговременная тенденция повышения удобства работы человека с компьютером. Эффективность работы человека становится основным фактором, определяющим эффективность вычислительной системы в целом. Усилия человека не должны тратиться на настройку параметров вычислительного процесса, как это происходило в ОС предыдущих поколений. Например, в системах пакетной обработки для мэйнфреймов каждый пользователь должен был с помощью языка управления заданиями определить большое количество параметров, относящихся к организации вычислительных процессов в компьютере. Так, для системы OS/360 язык управления заданиями JCL предусматривал возможность определения пользователем более 40 параметров, среди которых были приоритетные задания, требования к основной памяти, перечень используемых устройств ввода–вывода и режимы их работы.

Современная операционная система берет на себя выполнение задачи выбора параметров операционной среды, используя для этой цели различные адаптивные алгоритмы. Например, тайм–ауты в коммуникационных протоколах часто определяются в зависимости от условий работы сети. Распределение оперативной памяти между процессами осуществляется автоматически с помощью механизмов виртуальной памяти в зависимости от активности этих процессов и информации о частоте использования ими той или иной страницы. Мгновенные приоритеты процессов определяются динамически в зависимости от предыстории, включающей, например, время нахождения процесса в очереди, процент использования выделенного кванта времени, интенсивность ввода–вывода и т. п. Даже в процессе установки большинство ОС предлагают режим выбора параметров по умолчанию, который гарантирует пусть не оптимальное, но всегда приемлемое качество работы систем.

Постоянно повышается удобство интерактивной работы с компьютером путем включения в операционную систему развитых графических интерфейсов, использующих наряду с графикой звук и видеоизображение. Это особенно важно для превращения компьютера в терминал новой публичной сети, которой постепенно становится Интернет, так как для массового пользователя, терминал должен быть почти таким же понятным и удобным, как телефонный аппарат. Пользовательский интерфейс операционной системы становится все более интеллектуальным, направляя действия человека в типовых ситуациях и принимая за него рутинные решения.

Уровень удобств в использовании ресурсов, которые сегодня предоставляют пользователям, администраторам и разработчикам приложений ОС изолированных компьютеров, для сетевых операционных систем является только заманчивой перспективой. Пока пользователи и администраторы сети тратят значительное время на попытки выяснить, где находится тот или иной ресурс, разработчики сетевых приложений прилагают много усилий для определения местоположения данных и программных модулей в сети. ОС будущего должны обеспечить высокий уровень прозрачности сетевых ресурсов, взяв на себя задачу организации распределенных вычислений, превратив сеть в виртуальный компьютер. Именно этот смысл вкладывают в лаконичный лозунг «Сеть – это компьютер» специалисты компании Sun, но для превращения лозунга в жизнь разработчикам операционных систем нужно пройти еще немалый путь.

На сегодняшний день наиболее известными ОС являются семейства операционных систем *Microsoft Windows* (например, Windows XP, Windows Server 2003 и 2007) и *UNIX* (различные ОС на ядре Linux, например Linux Mandriva 2007).

4.2. Классификация операционных систем

Широко известно высказывание, согласно которому любая наука начинается с классификации. Само собой, что вариантов классификации может быть очень много: все будет зависеть от выбранного признака, по которому один объект мы будем отличать от другого. Однако, что касается ОС, здесь уже давно сформировалось относительно небольшое количество классификаций (рис. 4.1):

1) по назначению;
2) по режиму обработки задач;
3) по способу взаимодействия с системой;
4) по способу построения (архитектурным особенностям систем).

Прежде всего, различают *ОС общего и специального назначения*. ОС специального назначения, в свою очередь, подразделяются на следующие: для переносимых микрокомпьютеров и различных встроенных систем, организации и ведения баз данных, решения задач реального времени и т. п.

Основной особенностью *операционных систем реальною времени* (ОСРВ) является обеспечение обработки поступающих заданий в течение заданных интервалов времени, которые нельзя превышать.

Рис. 4.1. Классификация ОС

Поток заданий в общем случае не является планомерным и не может регулироваться оператором (характер следования событий можно предсказать лишь в редких случаях), то есть задания поступают и непредсказуемые моменты времени и без всякой очередности. В ОС, не предназначенных для решения задач реального времени, имеются некоторые накладные расходы процессорного времени на этапе инициирования (при выполнении которого ОС распознает все пожелания пользователей относительно решения своей задачи, загружает в оперативную память нужную программу и выделяет другие необходимые для ее выполнения ресурсы).

В ОСРВ подобные затраты могут отсутствовать, так как набор задач обычно фиксирован и вся информация о задачах известна еще до поступления запросов. Для подлинной реализации режима реального времени необходима (хотя этого и недостаточно) организация мультипрограммирования. *Мультипрограммирование* является основным средством повышения производительности вычислительной системы, а для решения задач реального времени производительность становится важнейшим фактором. Лучшие характеристики по производительности для систем реального времени обеспечиваются

однотерминальными ОСРВ. Средства организации мультитерминального режима всегда замедляют работу системы в целом, но расширяют функциональные возможности системы. Одной из наиболее известных ОСРВ для ПК является ОС QNX.

По режиму обработки задач и в зависимости от особенностей использованного алгоритма управления процессором операционные системы делят на многозадачные и однозадачные, *обеспечивающие однопрограммный и мультипрограммный режимы*, на системы, поддерживающие многонитевую обработку и не поддерживающие её, на многопроцессорные и однопроцессорные системы.

Поддержка многозадачности. По числу одновременно выполняемых задач операционные системы могут быть разделены на два класса:

- однозадачные (например, MS DOS, MSX);
- многозадачные (ОС ЕС, OS/2, UNIX, Windows).

Однозадачные ОС в основном выполняют функцию предоставления пользователю виртуальной машины, делая более простым и удобным процесс взаимодействия пользователя с компьютером. Однозадачные ОС включают средства управления периферийными устройствами, средства управления файлами, средства общения с пользователем.

Многозадачные ОС, кроме выполнения вышеперечисленных функций, управляют разделением совместно используемых ресурсов, таких, как процессор, оперативная память, файлы и внешние устройства.

Под *мультипрограммированием* понимается способ организации вычислений, когда на однопроцессорной вычислительной системе создается видимость одновременного выполнения нескольких программ. Любая задержка в решении программы (например, для осуществления операций ввода/вывода данных) используется для выполнения других (таких же либо менее важных) программ. Иногда при этом говорят о мультизадачном режиме. При этом, вообще говоря, мультипрограммный и мультизадачный режимы – это не синонимы, хотя и близкие понятия. Основное принципиальное отличие в этих терминах заключается в том, что мультипрограммный режим обеспечивает параллельное выполнение нескольких приложений и при этом программисты, создающие эти программы, не должны заботиться о механизмах организации их параллельной работы. Эти функции берет на себя сама ОС; именно она распределяет между выполняющимися приложениями ресурсы вычислительной системы, осуществляет необходимую синхронизацию вычислений и взаимодействие.

Мультизадачный режим, наоборот, предполагает, что забота о параллельном выполнении и взаимодействии приложений ложится как раз на прикладных программистов. В современной технической и, тем более, научно-популярной литературе об этом различии часто забывают, тем самым внося некоторую путаницу. Можно, однако, заметить, что современные ОС для ПК реализуют и мультипрограммный, и мультизадачный режимы.

Различают ОС с вытесняющей и невытесняющей многозадачностью. Важнейшим разделяемым ресурсом является процессорное время. Способ распределения процессорного времени между несколькими одновременно существующими в системе *процессами* (или *нитями*) во многом определяет специфику ОС. Среди множества существующих вариантов реализации многозадачности можно выделить две группы алгоритмов:

- невытесняющая многозадачность (NetWare, Windows 3.x);
- вытесняющая многозадачность (Windows, OS/2, UNIX).

При невытесняющей многозадачности активный процесс выполняется до тех пор, пока он сам, по собственной инициативе, не отдаст управление операционной системе для того, чтобы та выбрала из очереди другой готовый к выполнений процесс. При вытесняющей многозадачности решение о переключении процессора с одного процесса на другой принимается операционной системой, а не самим активным процессом.

Поддержка *многонитевости*.

Важным свойством операционных систем является возможность распараллеливания вычислений в рамках одной задачи. Многонитевая ОС разделяет процессорное время не между задачами, а между их отдельными ветвями (*нитями*).

Многопроцессорная обработка.

Другим важным свойством ОС является отсутствие или наличие в ней средств поддержки многопроцессорной обработки – *мулътипроцессирование.* Мультипроцессирование приводит к усложнению всех алгоритмов управления ресурсами.

Симметричная многопроцессорная обработка (Symmetric MultiProcessing, SMP) – это способность операционной системы работать с компьютером, в котором установлены два и более процессора. Операционная система в данном случае должна обеспечивать балансировку нагрузки, чтобы дать работу каждому из процессоров. Механизм SMP может использоваться как при

выполнении одной программы, так и нескольких приложений – в любом случае нагрузка распределяется равномерно.

В наши дни становится общепринятым введение в ОС функций поддержки многопроцессорной обработки данных. Такие функции имеются в операционных системах Solaris 2.x фирмы Sun, Open Server 3.x фирмы Santa Cms Operations, OS/2 фирмы IBM, Windows NT фирмы Microsoft и NetWare фирмы Novell.

При организации работы с вычислительной системой в диалоговом режиме можно говорить об *однопользовательских,* однотерминальных (MS DOS, Windows 3.x, ранние версии OS/2) и *мультитерминальных* ОС (UNIX, Windows). В мультитерминальных ОС с одной вычислительной системой одновременно могут работать несколько пользователей, каждый со своего терминала. При этом у пользователей возникает иллюзия, что у каждого из них имеется своя собственная вычислительная система. Очевидно, что для организации мультитерминального доступа к вычислительной системе необходимо обеспечить мультипрограммный режим работы. В качестве одного из примеров мультитерминальных ОС для ПК можно назвать Linux.

По основному архитектурному принципу ОС разделяются на *микроядерные* и *монолитные.* В некоторой степени это разделение тоже условно, однако можно в качестве яркого примера микроядерной ОС привести ОСРВ QNX, тогда как в качестве монолитной можно назвать Windows или ОС Linux.

Ядро ОС Windows мы не можем изменить: нам не доступны его исходные коды и у нас нет программы для сборки (компиляции) этого ядра. А вот в случае с Linux мы можем сами собрать ядро, которое нам необходимо, включив в него те необходимые программные модули и драйверы, которые мы считаем целесообразным включить именно в ядро (а не обращаться к ним из ядра).

4.3. Примеры операционных систем

На сегодняшний день наиболее известными ОС являются *семейства операционных систем Microsoft Windows и UNIX*. Первые ведут свою «родословную» от операционной системы MS–DOS, которой оснащались первые персональные компьютеры (ПК) фирмы IBM. ОС UNIX была разработана группой сотрудников Bell Labs под руководством Денниса Ричи, Кена Томпсона и Брайана Кернигана (Dennis Ritchie, Ken Thompson, Brian Kernighan) в 1969 году.

ОС MS DOS

В 80–90-е годы одной из самых популярных операционных систем была операционная система, разработанная фирмой Microsoft, – *MS DOS*, имеющая интерфейс командной строки (рис. 4.2).

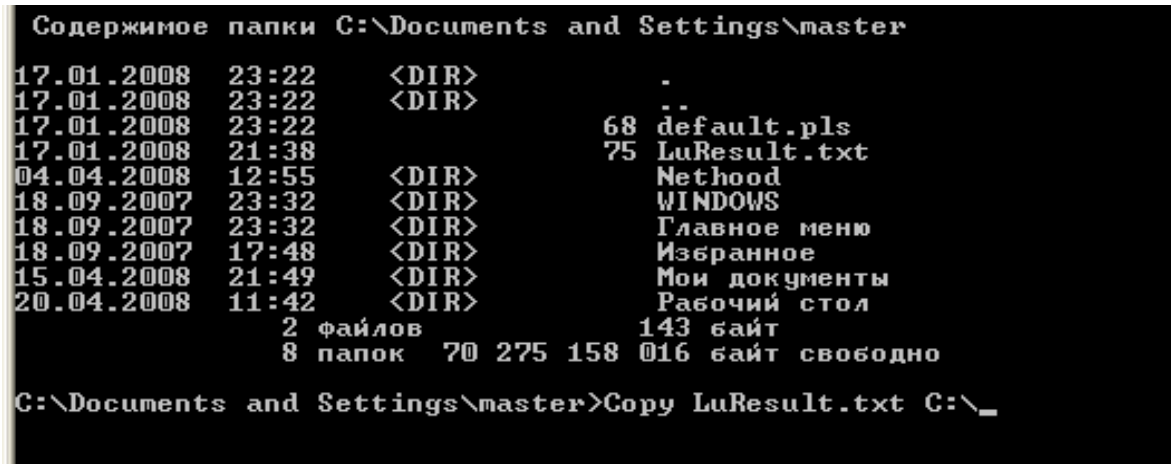


Рис. 4.2. Вид экрана в MS DOS

Несмотря на то, что в среде MS DOS работали тысячи программ, разработка таких программ была связана со значительными трудностями. Программистам приходилось либо самим разрабатывать средства для создания диалогового интерфейса (меню, запросов, окон и т.д.), либо использовать библиотеки программ. И крупным фирмам приходилось содержать множество сотрудников, занятых разработкой программ графического интерфейса, поддержки десятков типов мониторов и сотен типов принтеров. Это увеличивало сроки и финансовые расходы на создание и сопровождение программ, замедляло развитие всей отрасли разработки программного обеспечения.

В отличие от компьютеров фирмы IBM, для ПК типа Macintosh фирмы Apple была разработана операционная система, которая представляла пользователям удобный графический интерфейс, средства взаимодействия с внешними устройствами. В 1985 г. и фирма Microsoft выпустила собственную *операционную среду Windows.* В 1987–1989 гг. стали появляться и удобные программы, работающие в среде Windows – Microsoft Word для Windows, Excel.

Windows предлагает пользователю *оконный интерфейс,* когда каждой выполняемой программе отводится *экранное окно.* То есть появилась возможность одновременного использования нескольких программ. Причем информацию из окна одной программы в окно другой программы можно было передавать (через копирование) при помощи *буфера обмена.*

Элементы окон, кнопок, значков для Windows-приложений стандартизированы. Для подготовки документов, содержащих текст, Windows позволяет использовать масштабируемые шрифты, применяемые как для экранного вывода, так и для распечатки на принтере. Благодаря этому в процессе подготовки документа можно видеть на экране практически то же, что будет получено на бумаге – так называемый принцип WYSIWYG (What You See Is What You Get).

Выпуск графической операционной оболочки Microsoft Windows 3.0 стал главным событием 1990 г. на программном рынке. В 1992 г. появилась версия Windows 3.1. Интерфейс был улучшен, в частности были усилены возможности управления экранными объектами мышью (drag–and–drop – метод перетаскивания). Что же пользователь получает при использовании Windows и Windows-приложений?

1. *Единый пользовательский интерфейс,* т.е. программисты не изобретают собственные средства для создания пользовательского интерфейса.

2. *Многозадачность* – Windows обеспечивает возможность одновременного выполнения нескольких задач и простого переключения с одной задачи на другую.

3. *Совместимость с DOS-приложениями,* т.е. для выполнения DOS-программ, как правило, нет необходимости выходить из Windows.

4. Средства обмена данными:

- *буфер обмена данными;*
- *динамический обмен данными* между приложениями (DDE – Dynamic Data Exchange) – одна программа может использовать данные, созданные другой программой;
- *механизм связи и внедрения объектов* (OLE) является усовершенствованием средств DDE. Здесь приложение, использующее данные, может запустить программу, с помощью которой были созданы «внедренные» данные, для их изменения.

5. *Поддержка масштабируемых шрифтов.*

6. *Удобство поддержки устройств.* Для подключения к ПК любого нового устройства достаточно установить драйвер этого устройства, предназначенный для Windows, после чего все Windows–приложения смогут работать с этим устройством.

7. *Поддержка мультимедиа.* При подключении соответствующих устройств Windows может воспринимать звуковую информацию от микрофона, компакт-диска, выводить звуки и движущиеся изображения.

Принято считать версии Windows 3.0 и Windows 3.1, предшествующие Windows 95, *операционными оболочками:* т.е.

программами, облегчающими работу с MS DOS. Загрузка этих версий Windows производится после загрузки MS DOS соответствующей командой. Новая версия *Windows 95* была принципиально новой операционной системой фирмы Microsoft с колоссальным числом особенностей и возможностей. Windows 95 заменила MS DOS, так как она включала в себя все, что ей нужно от MS DOS, и могла запускать программы MS DOS так, как будто это программы Windows.

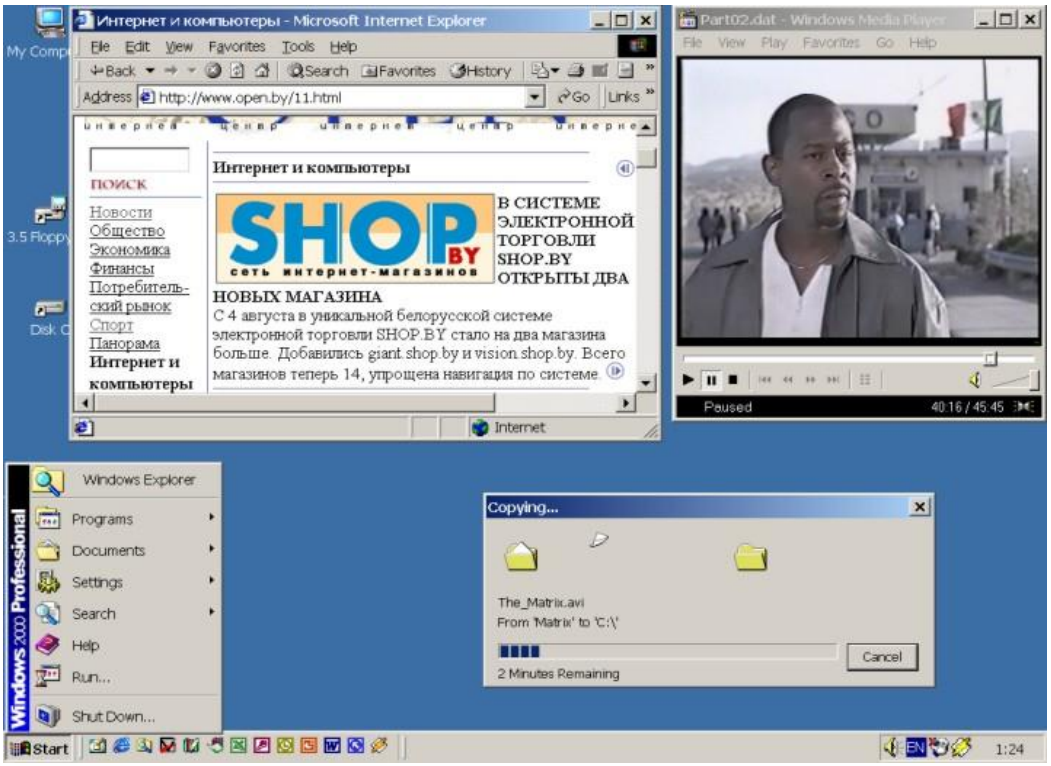


Рис. 4.3. Пример экрана ОС Windows

В переводе с английского Windows – это «окна». Программа называется так не случайно – при работе с этой программой используется так называемый *многооконный графический интерфейс.* Пользователь общается с компьютером посредством *графических символов,* т.е. маленьких картинок (которые называются *«пиктограммами»* или *«иконками»).* Причем в Windows в любой момент можно перейти к работе с программой одного из открытых окон (рис. 4.3).

ОС Windows

Windows – это высокопроизводительная, многозадачная и многопотоковая операционная система с расширенными сетевыми возможностями. Работа над несколькими задачами возможна благодаря новым программам, которые обеспечивают:

- *вытеснение,* т.е. операционная система в любой момент может прервать их выполнение и переключиться на другую задачу;
- *отдельное адресное пространство.* Приложения выполняются в своей защищенной области памяти, что делает невозможным нарушение их целостности со стороны других программ;
- *поддержку потоков.* Обеспечивается многозадачность в пределах одного приложения (т.е. приложения могут одновременно запускать несколько потоков);
- *технологию OLE.*

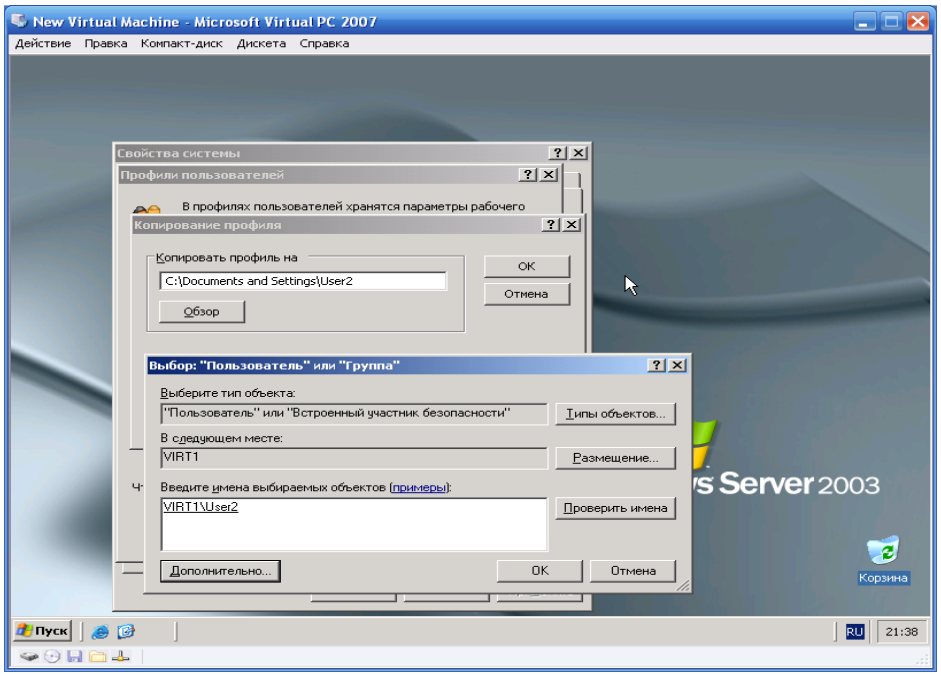


Рис. 4.4. Установленная ОС Microsoft Windows Server 2003 на VirtualPC 2007

Все эти средства операционной среды Windows сохраняются и в последующих версиях ОС Windows (Windows 2000, Windows XP, Windows Server 2003 и 2007, представленная на рис. 4.4) . Кроме того, в новые версии входят программы, применение которых повышает производительность компьютера, надежность его работы, в том числе в вычислительных сетях.

UNIX-системы

В наши дни, когда говорят об операционной системе *UNIX*, чаще всего имеют в виду не конкретную ОС, а скорее целое семейство UNIX-подобных операционных систем. Само же слово UNIX стало зарегистрированной торговой маркой корпорации AT&T. В настоящее время Unix используется в корпоративной среде, а также нередко применяется в системах клиент-сервер сетей intranet. Пример окна ОС Unix можно увидеть на рис. 4.5.

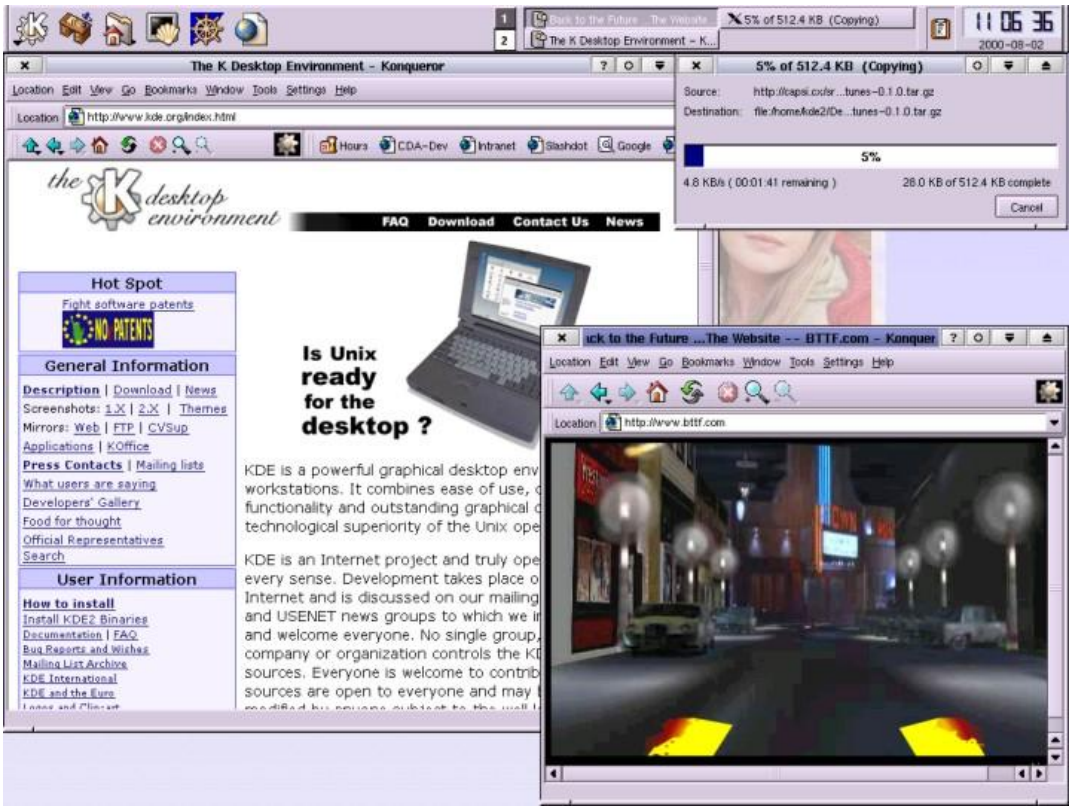


Рис. 4.5. ОС UNIX с XWindow

В конце 1970-х годов (теперь уже прошлого столетия) сотрудники Калифорнийского университета в Беркли внесли ряд усовершенствований в исходные коды UNIX, включая работу с протоколами семейства TCP/IP. Их разработка стала известна под именем BSD («Berkeley Systems Distribution»). Она распространялась под лицензией, которая позволяла дорабатывать и усовершенствовать продукт и передавать результат третьим лицам (с исходными кодами или без них) при условии, что будет указано, какая часть кода разработана в Беркли. Исторический недостаток Unix – недоступность системы для программистов, работающих вне промышленных или университетских вычислительных центров. Несмотря на то, что версии Unix для ПК существуют уже давно, они не обладают изяществом и мощью, отличающими ОС для миникомпьютеров, мэйнфреймов или современных серверов. Кроме того, ранние коммерческие версии Unix были слишком дороги – зачастую, дороже компьютеров, на которых им предстояло работать.

До недавнего времени у этой ОС не было собственного графического интерфейса, но даже появление такового (XWindow, рис. 4.5) не облегчило работу пользователям: все-таки UNIX остается системой для инженеров и настоящих профессионалов.

ОС LINUX

Linux – продукт культуры Unix. Linux – свободно распространяемое ядро Unix-подобной системы, написанное Линусом Торвальдсом 17 сентября 1991 года. Эта система обновляется и по сей день при помощи большого числа добровольцев по всей Сети. Linux обладает всеми свойствами современной Unix-системы, включая настоящую многозадачность, развитую подсистему управления памятью и сетевую подсистему. Основное отличие Linux от UNIX в том, что Linux несколько проще и, самое главное, является абсолютно бесплатной. Сегодня Linux становится конкурентом операционных систем Microsoft. Ядро Linux, поставляемое вместе со свободно распространяемыми прикладными и системными программами, образует полнофункциональную универсальную операционную систему. На сегодняшний день существует множество различных поставок Linux-дистрибутивов, которые можно разделить на дистрибутивы общего назначения и специализированные. К специализированным дистрибутивам относятся такие, как LinuxRouter, – урезанная поставка Linux для создания дешевого маршрутизатора и др. Обычно под словосочетанием «ОС Linux» понимают именно дистрибутивы Linux общего назначения, они образуют лицо ОС Linux такой, какой ее знают большинство пользователей ОС. ОС на основе ядра Linux имеет широкие возможности:

Совместимость. Большая часть ядра Linux написана на языке Си, благодаря чему система достаточно легко переносится на различные аппаратные архитектуры. Сегодня официальное ядро Linux работает в среднем на около 40 платформах. Кроме того, существует много портов Linux, распространяемых отдельно от официального ядра. Ядро Linux способно работать на многопроцессорных системах, обеспечивая эффективное использование всех процессоров.

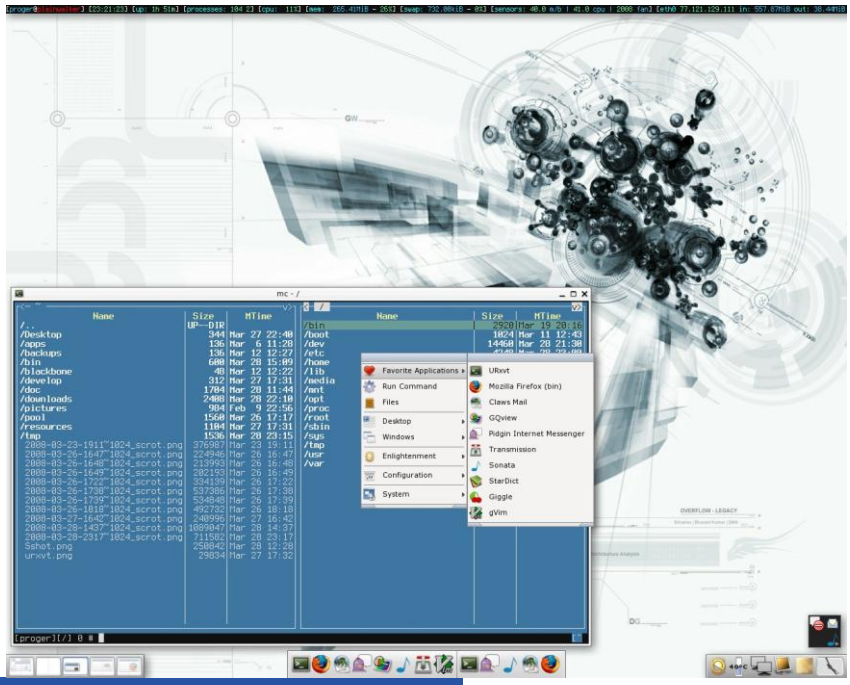
Реальная многозадачность. Все процессы независимы – ни один из них не должен мешать выполнению других задач. Для этого ядро Linux осуществляет режим разделения времени центрального процессора, поочередно выделяя каждому процессу интервалы времени для выполнения. Это существенно отличается от режима вытесняющей многозадачности, реализованной в Windows, когда процесс должен сам «уступить» процессор другим процессам, что может задержать их выполнение.

Свопирование оперативной памяти на диск. Свопирование оперативной памяти на диск позволяет работать при ограниченном объеме физической оперативной памяти. Для этого содержимое некоторых частей (страниц) оперативной памяти записываются в

выделенную область на жестком диске, которая трактуется как дополнительная оперативная память. Это несколько снижает скорость работы, но позволяет организовать работу программ, требующих большего объема ОЗУ, чем фактически имеется в компьютере.

Страничная организация памяти. Системная память Linux организована в виде страниц объемом 4KB. Если оперативная память полностью исчерпана, ОС будет искать давно не использованные страницы памяти для их перемещения из памяти на жесткий диск. Если какие-либо из этих страниц становятся нужны, Linux восстанавливает их с диска. Некоторые старые Unix-системы и некоторые современные платформы (включая Microsoft Windows) переносят на диск все содержимое ОП, относящееся к неработающему в данный момент приложению, (т. е. все страницы памяти, относящиеся к приложению, сохраняются на диске при нехватке памяти) что менее эффективно.

Динамические библиотеки. В Linux есть механизмы динамической подгрузки библиотек. Суть динамической подгрузки состоит в том, что запущенная программа может по собственному усмотрению подключить к себе какую-либо библиотеку. В Linux библиотеки используются повсеместно, поскольку это очень удобный способ «не изобретать велосипеды». Даже ядро Linux в каком-то смысле представляет собой библиотеку механизмов, называемых системными вызовами.



...са KDE и Gnome

Графический интерфейс. Linux использует стандартную оконную систему X. В большинстве дистрибутивов используется свободно распространяемая реализация X'ов–Xorg. Xorg поддерживает почти все популярные графические адаптеры на платформе Intel и некоторых других. Наиболее популярные графические среды сегодня – Gnome (GNU Network Object Model Environment) и KDE (см. рис. 4.6).

Сетевые возможности. Linux можно интегрировать в любую локальную сеть. Поддерживаются все службы Unix, включая Networked File System (NFS), удаленный доступ (telnet, rlogin), работа в TCP/IP сетях, dial–up-доступ по протоколам SLIP и PPP, и т. д.

Также поддерживается включение Linux-машины как сервера или клиента для другой сети, в частности, работает общее использование (sharing) файлов и удаленная печать в Macintosh, NetWare и Windows. Linux может служить файл-сервером по протоколам NFS (как правило используемом только на Unix машинах), SMB (Netbios over TCP/IP, используемый на различных Windows платформах), FTP (File Transfer Protocol, один из наиболее распространённых протоколов передачи файлов по сети), AppleShare и IPX (Novell).

Поддержка различных форматов файловых систем. Основной файловой системой Linux является его собственная журналируемая ext3fs, которая позволяет эффективно использовать дисковое пространство. Официальное ядро содержит поддержку более 20 различных файловых систем.

В последнее время новые версии продуктов, основанных на ядре Linux, поражают своей функциональностью и красотой. Например, графический интерфейс Compiz оставляет по красоте и функциональности далеко позади новый интерфейс, не оправдавший надежды разработчиков и пользователей Windows Vista, – Aero.

Многопользовательский доступ. Linux – это не только многозадачная ОС, она поддерживает возможность одновременной работы многих пользователей. При этом Linux может предоставлять все системные ресурсы пользователям, работающим с хостом через различные удаленные терминалы, как показано на рис. 4.7.

Возможность запуска исполняемых файлов других ОС. Linux не является первой в истории операционной системой. Для ранее разработанных ОС, включая DOS, Windows, FreeBSD или OS/2, разработана масса различного, в том числе очень полезного и очень неплохого программного обеспечения. Для запуска таких программ под Linux разработаны эмуляторы DOS, Windows. Более того, фирмой Vmware разработана система виртуальных машин, представляющая

собой эмулятор компьютера, в котором можно запустить любую операционную систему. Имеются аналогичные разработки и у других фирм. ОС Linux способна также выполнять бинарные файлы других Intel-ориентированных Unix-платформ, соответствующих стандарту iBCS2 (intel Binary Compatibility).

Рис. 4.7. Совместимость ядра Linux с различными архитектурами

Для подключения дополнительного оборудования необязательно иметь к нему драйвера, в большинстве случаев достаточно лишь подключить дэвайс и система сама найдет для него нужный драйвер в репозитории. Многие специалисты считают, что ОС на базе ядра Linux в скором времени смогут заменить привычный нам Windows.

ОС для Macintosh

Первые Macintosh'n, появившиеся в далеком 1984 году, в корне изменили восприятие компьютера с точки зрения обыкновенного человека, заложили основы того, что в дальнейшем стали применять практически все компьютерные фирмы. В течение времени наряду с совершенствованием самих компьютеров Macintosh развивалась и их программная основа – операционная система Mac OS (System). Mac OS является ближайшим конкурентом Windows и Linux (рис. 4.8).

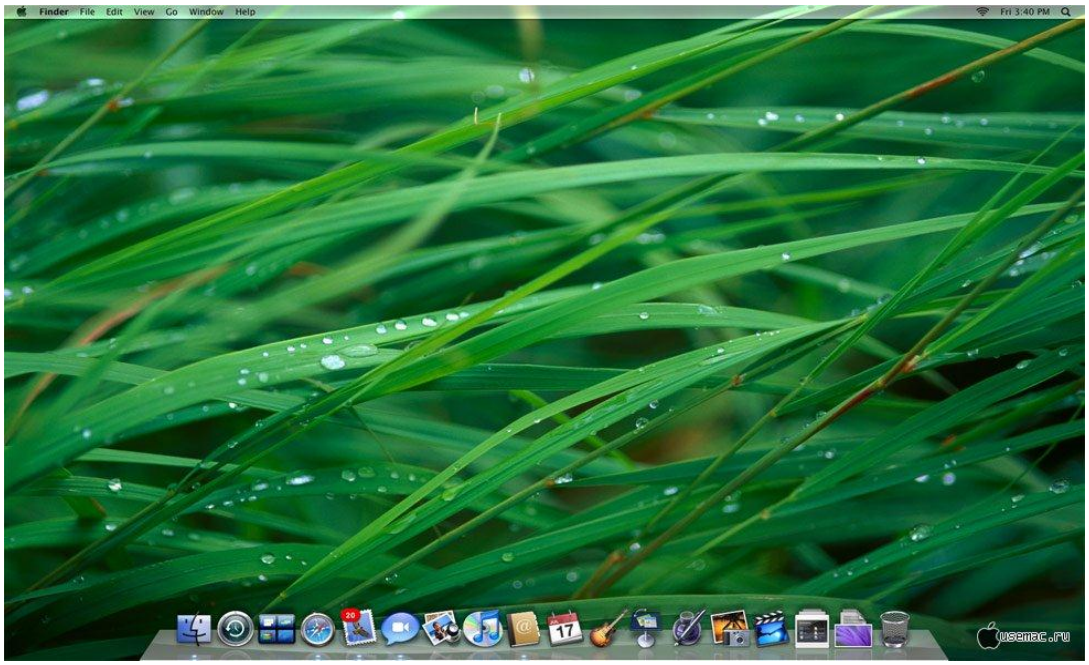


Рис. 4.8. Вид экрана OS Mac

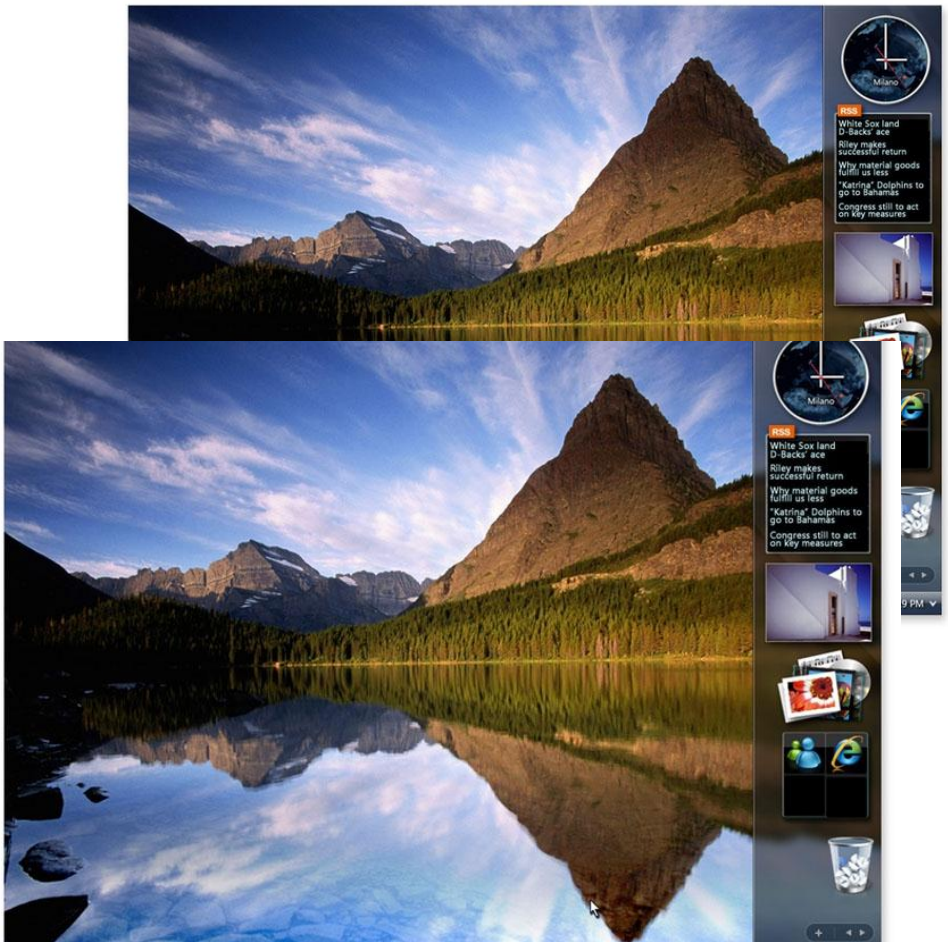


Рис. 4.9. Визуальное сравнение экрана OS Mac и WindowsVista

Большую популярность Mac OS завоевал в США. В США более 30 % компьютеров имеют платформу Mac OS. Такая популярность объясняется её мультимедийными возможностями и приятным внешним видом. Но так как Mac OS не использует BIOS, из-за этого она не может запускаться на IBM совместимых компьютерах.

В России Mac OS является новым, мало известным продуктом, поэтому и не имеет такого числа пользователей, как в Европе и США. Для некоторых пользователей переключение с PC на Mac кажется весьма непростым этапом адаптации к иной опреционной системе Mac Os, но те, кто перешел на Apple Mac, дают положительные отзывы, а именно, что Macintosh довольно-таки опережает по комфорту, простоте, быстроте и своей безопасности. Производительные программы макинтош для Apple Mac доступны так же, как и для Windows (рис. 4.9).

System 7 – операционная система для компьютеров Macintosh, весьма распространенных в США. Традиционно компьютеры фирмы Apple используются в издательской деятельности, поэтому System 7 оснащена хорошим GUI и поддержкой мультимедиа.

4.4. Функции операционных систем

Сегодня существует большое количество разных типов операционных систем, отличающихся областями применения, аппаратными платформами и методами реализации. Естественно, это обуславливает и значительные функциональные различия этих ОС. Даже у конкретной операционной системы набор выполняемых функций зачастую определить не так просто – та функция, которая сегодня выполняется внешним по отношению к ОС компонентом, завтра может стать ее неотъемлемой частью и наоборот. Поэтому при изучении операционных систем очень важно из всего многообразия выделить те функции, которые присущи всем ОС.

Операционная система представляет комплекс системных и служебных программных средств. С одной стороны, она опирается на базовое программное обеспечение компьютера, входящее в его систему BIOS (базовая система ввода–вывода), с другой стороны, она сама является опорой для программного обеспечения более высоких уровней – прикладных и большинства служебных приложений. Приложениями операционной системы принято называть программы, предназначенные для работы под управлением данной системы.

Основная функция всех операционных систем – посредническая. Она заключаются в обеспечении нескольких видов интерфейса:

- интерфейса между пользователем и программно-аппаратными средствами компьютера (интерфейс пользователя);
- интерфейса между программным и аппаратным обеспечением (аппаратно-программный интерфейс);
- интерфейса между разными видами программного обеспечения (программный интерфейс).

Даже для одной аппаратной платформы, например такой, как IBM PC, существует несколько операционных систем. Различия между ними рассматривают в двух категориях: внутренние и внешние. Внутренние различия характеризуются методами реализации основных функций. Внешние различия определяются наличием и доступностью приложений данной системы, необходимых для удовлетворения технических требований, предъявляемых к конкретному рабочему месту.

Таким образом, ОС компьютера представляет собой комплекс взаимосвязанных программ, который действует как интерфейс между приложениями и пользователями с одной стороны, и аппаратурой компьютера с другой стороны. В соответствии с этим определением ОС выполняет две группы функций:

- предоставление пользователю или программисту вместо реальной аппаратуры компьютера расширенной виртуальной машины, с которой удобней работать и которую легче программировать;
- повышение эффективности использования компьютера путем рационального управления его ресурсами в соответствии с некоторым критерием.

4.4.1. Операционные системы для автономного компьютера

ОС как виртуальная машина

Для того чтобы успешно решать свои задачи, современный пользователь или даже прикладной программист может обойтись без досконального знания аппаратного устройства компьютера. Так, например, при работе с диском программисту, пишущему приложение для работы под управлением ОС, или конечному пользователю ОС достаточно представлять его в виде некоторого набора файлов, каждый из которых имеет имя. Последовательность действий при работе с файлом заключается в его открытии, выполнении одной или нескольких операций чтения или записи, а затем в закрытии файла. Если бы программист работал непосредственно с аппаратурой компьютера, без участия ОС, то для организации чтения блока данных с диска программисту пришлось бы использовать более десятка команд с

указанием множества параметров: номера блока на диске, номера сектора на дорожке и т. п. А после завершения операции обмена с диском он должен был бы предусмотреть в своей программе анализ результата выполненной операции.

Операционная система избавляет программистов не только от необходимости напрямую работать с аппаратурой дискового накопителя, предоставляя им простой файловый интерфейс, но и берет на себя все другие рутинные операции, связанные с управлением другими аппаратными устройствами компьютера: физической памятью, таймерами, принтерами и т. д.

В результате реальная машина, способная выполнять только небольшой набор элементарных действий, определяемых ее системой команд, превращается в виртуальную машину, выполняющую широкий набор гораздо более мощных функций. Виртуальная машина тоже управляется командами, но это уже команды другого, более высокого уровня: удалить файл с определенным именем, запустить на выполнение некоторую прикладную программу, повысить приоритет задачи, вывести текст из файла на печать. Таким образом, назначение ОС состоит в предоставлении пользователю/программисту некоторой расширенной виртуальной машины, которую легче программировать и с которой легче работать, чем непосредственно с аппаратурой, составляющей реальный компьютер или реальную сеть.

ОС как система управления ресурсами

Операционная система не только предоставляет пользователям и программистам удобный интерфейс к аппаратным средствам компьютера, но и является механизмом, распределяющим ресурсы компьютера. К числу основных ресурсов современных вычислительных систем могут быть отнесены такие ресурсы, как процессоры, основная память, таймеры, наборы данных, диски, накопители на магнитных лентах, принтеры, сетевые устройства и некоторые другие. Ресурсы распределяются между процессами.

Процесс (задача) представляет собой базовое понятие большинства современных ОС и часто кратко определяется как программа в стадии выполнения. Программа – это статический объект, представляющий собой файл с кодами и данными. Процесс – это динамический объект, который возникает в операционной системе после того, как пользователь или сама ОС решает «запустить программу на выполнение», то есть создать новую единицу вычислительной работы. Например, ОС может создать процесс в ответ на команду пользователя

run prgl. exe, где prgl. exe – это имя файла, в котором хранится код программы.

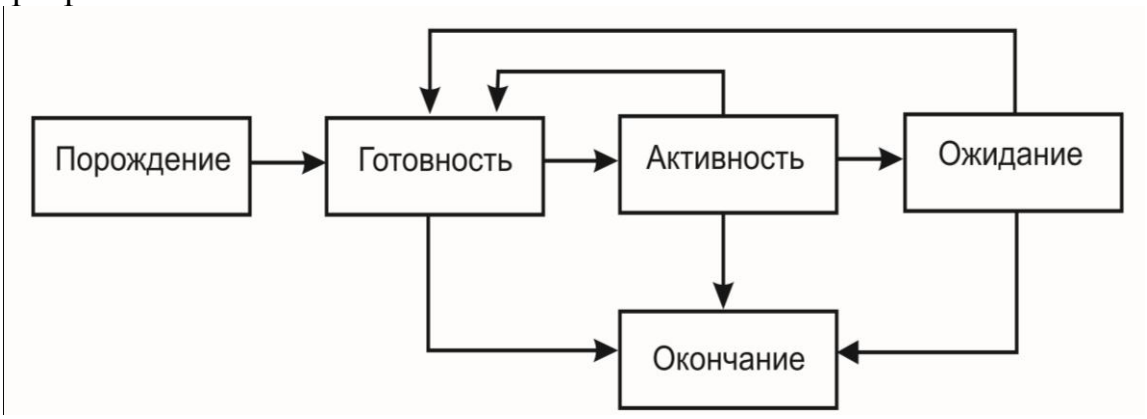


Рис. 4.10. Графа состояний переходов процесса из одной фазы в другую

При исполнении программ на центральном процессоре следует различать следующие характерные состояния (рис. 4.10):

- *порождение* – подготовку условий для исполнения процессором задачи;
- *активное состояние* (или «Счет») – непосредственное исполнение процессором задачи;
- *ожидание* по причине занятости какого-либо требуемого ресурса;
- *готовность* – программа не исполняется, но все необходимые для исполнения программы ресурсы, кроме центрального процессора, предоставлены;
- *окончание* – нормальное или аварийное завершение исполнения программы, после которого процессор и другие ресурсы ей не предоставляются.

Остановимся более подробно на понятиях: «процесс», «нить», «поток».

Во многих современных ОС для обозначения минимальной единицы работы ОС используют термин «нить», или «поток», при этом изменяется суть термина «процесс».

Понятия «процесс» и «поток»

Чтобы поддерживать мультипрограммирование, ОС должна определить и оформить для себя те внутренние единицы работы, между которыми будет разделяться процессор и другие ресурсы компьютера. В настоящее время в большинстве операционных систем определены два типа единиц работы. Более крупная единица работы, обычно носящая название процесса, или задачи, требует для своего выполнения

нескольких более мелких работ, для обозначения которых используют термины «поток», или «нить».

При использовании этих терминов часто возникают сложности. Это происходит в силу нескольких причин. Во-первых, специфика различных ОС, когда совпадающие по сути понятия получили разные названия, например задача (task) в OS/2, OS/360 и процесс (process) в UNIX, Windows NT, NetWare. Во-вторых, по мере развития системного программирования и методов организации вычислений некоторые из этих терминов получили новое смысловое значение, особенно это касается понятия «процесс», который уступил многие свои свойства новому понятию «поток». В-третьих, терминологические сложности порождаются наличием нескольких вариантов перевода англоязычных терминов на русский язык. Например, термин «thread» переводится как «нить», «поток», «облегченный процесс», «минизадача» и др. Далее в качестве названия единиц работы ОС будут использоваться термины «процесс» и «поток». В тех же случаях, когда различия между этими понятиями не будут играть существенной роли, они объединяются под обобщенным термином «задача».

Итак, в чем же состоят принципиальные отличия в понятиях «процесс» и «поток»?

Очевидно, что любая работа вычислительной системы заключается в выполнении некоторой программы. Поэтому и с процессом, и с потоком связывается определенный программный код, который для этих целей оформляется в виде исполняемого модуля. Чтобы этот программный код мог быть выполнен, его необходимо загрузить в оперативную память, возможно, выделить некоторое место на диске для хранения данных, предоставить доступ к устройствам ввода–вывода, например, к последовательному порту, для получения данных по подключенному к этому порту модему; и т. д. В ходе выполнения программе может также понадобиться доступ к информационным ресурсам, например, файлам, портам TCP/UPD. И, конечно же, невозможно выполнение программы без предоставления ей процессорного времени, то есть времени, в течение которого процессор выполняет коды данной программы.

В операционных системах, где существуют и процессы, и потоки, процесс рассматривается операционной системой как заявка на потребление всех видов ресурсов, кроме одного − процессорного времени. Этот последний важнейший ресурс распределяется операционной системой между другими единицами работы − потоками, которые и получили свое название благодаря тому, что они представляют собой последовательности (потоки выполнения) команд.

В простейшем случае процесс состоит из одного потока, и именно таким образом трактовалось понятие «процесс» до середины 80-х годов (например, в ранних версиях UNIX) и в таком же виде оно сохранилось в некоторых современных ОС. В таких системах понятие «поток» полностью поглощается понятием «процесс», то есть остается только одна единица работы и потребления ресурсов − процесс. Мультипрограммирование осуществляется в таких ОС на уровне процессов.

Для того чтобы процессы не могли вмешаться в распределение ресурсов, а также не могли повредить коды и данные друг друга, важнейшей задачей ОС является изоляция одного процесса от другого. Для этого операционная система обеспечивает каждый процесс отдельным виртуальным адресным пространством, так что ни один процесс не может получить прямого доступа к командам и данным другого процесса.

Виртуальное адресное пространство процесса − это совокупность адресов, которыми может манипулировать программный модуль процесса. Операционная система отображает виртуальное адресное пространство процесса на отведенную процессу физическую память.

При необходимости взаимодействия процессы обращаются к операционной системе, которая, выполняя функции посредника, предоставляет им средства межпроцессной связи − конвейеры, почтовые ящики, разделяемые секции памяти и некоторые другие.

Однако в системах, в которых отсутствует понятие потока, возникают проблемы при организации параллельных вычислений в рамках процесса. Действительно, при мультипрограммировании повышается пропускная способность системы, но отдельный процесс никогда не может быть выполнен быстрее, чем в однопрограммном режиме (всякое разделение ресурсов только замедляет работу одного из участников за счет дополнительных затрат времени на ожидание освобождения ресурса). Однако приложение, выполняемое в рамках одного процесса, может обладать внутренним параллелизмом, который в принципе мог бы позволить ускорить его решение. Если, например, в программе предусмотрено обращение к внешнему устройству, то на время этой операции можно не блокировать выполнение всего процесса. Параллельное выполнение нескольких работ в рамках одного интерактивного приложения повышает эффективность работы пользователя. Так, при работе с текстовым редактором желательно иметь возможность совмещать набор нового текста с такими продолжительными по времени операциями, как переформатирование

значительной части текста, печать документа или его сохранение на локальном или удаленном диске. Еще одним примером необходимости распараллеливания является сетевой сервер баз данных. В этом случае параллелизм желателен как для обслуживания различных запросов к базе данных, так и для более быстрого выполнения отдельного запроса за счет одновременного просмотра различных записей базы.

Потоки возникли в операционных системах как *средство распараллеливания вычислений.* Конечно, задача распараллеливания вычислений в рамках одного приложения может быть решена и традиционными способами.

Во-первых, прикладной программист может взять на себя сложную задачу организации параллелизма, выделив в приложении некоторую подпрограмму – диспетчер, которая периодически передает управление той или иной ветви вычислений. При этом программа получается логически весьма запутанной, с многочисленными передачами управления, что существенно затрудняет ее отладку и модификацию.

Во-вторых, решением является создание для одного приложения нескольких процессов для каждой из параллельных работ. Однако использование для создания процессов стандартных средств ОС не позволяет учесть тот факт, что эти процессы решают единую задачу, а значит, имеют много общего между собой: они могут работать с одними и теми же данными, использовать один и тот же кодовый сегмент, наделяться одними и теми же правами доступа к ресурсам вычислительной системы. Так, если в примере с сервером баз данных создавать отдельные процессы для каждого запроса, поступающего из сети, то все процессы будут выполнять один и тот же программный код и выполнять поиск в записях, общих для всех процессов файлов данных. ОС при таком подходе будет рассматривать эти процессы наравне со всеми остальными процессами и с помощью универсальных механизмов обеспечивать их изоляцию друг от друга. В данном случае все эти достаточно громоздкие механизмы используются явно не по назначению, выполняя не только бесполезную, но и вредную работу, затрудняющую обмен данными между различными частями приложения. Кроме того, на создание каждого процесса ОС тратит определенные системные ресурсы, которые в данном случае неоправданно дублируются, – каждому процессу выделяются собственное виртуальное адресное пространство, физическая память, закрепляются устройства ввода–вывода и т. п.

Из всего вышеизложенного следует, что в операционной системе наряду с процессами нужен другой механизм распараллеливания вычислений, который учитывал бы тесные связи между отдельными

ветвями вычислений одного и того же приложения. Для этих целей современные ОС предлагают механизм многопоточной обработки (multithreading). При этом вводится новая единица работы – *поток выполнения*, а понятие «процесс» в значительной степени меняет смысл. Понятию «поток» соответствует последовательный переход процессора от одной команды программы к другой. ОС распределяет процессорное время между потоками. Процессу ОС назначает адресное пространство и набор ресурсов, которые совместно используются всеми его потоками.

Создание потоков требует от ОС меньших накладных расходов, чем процессов. В отличие от процессов, которые принадлежат разным, вообще говоря, конкурирующим приложениям, все потоки одного процесса всегда принадлежат одному приложению, поэтому ОС изолирует потоки в гораздо меньшей степени, нежели процессы в традиционной мультипрограммной системе. Все потоки одного процесса используют общие файлы, таймеры, устройства, одну и ту же область оперативной памяти, одно и то же адресное пространство. Это означает, что они разделяют одни и те же глобальные переменные. Поскольку каждый поток может иметь доступ к любому виртуальному адресу процесса, один поток может использовать стек другого потока. Между потоками одного процесса нет полной защиты, потому что, во-первых, это невозможно, а во-вторых, не нужно. Чтобы организовать взаимодействие и обмен данными, потокам вовсе не требуется обращаться к ОС, им достаточно использовать общую память – один поток записывает данные, а другой читает их. С другой стороны, потоки разных процессов по-прежнему хорошо защищены друг от друга.

Итак, мультипрограммирование более эффективно на уровне потоков, а не процессов. Каждый поток имеет собственный счетчик команд и стек. Задача, оформленная в виде нескольких потоков в рамках одного процесса, может быть выполнена быстрее за счет псевдопараллельного (или параллельного в мультипроцессорной системе) выполнения ее отдельных частей. Например, если электронная таблица была разработана с учетом возможностей многопоточной обработки, то пользователь может запросить пересчет своего рабочего листа и одновременно продолжать заполнять таблицу. Особенно эффективно можно использовать многопоточность для выполнения распределенных приложений, например, многопоточный сервер может параллельно выполнять запросы сразу нескольких клиентов. Использование потоков связано не только со стремлением повысить производительность системы за счет параллельных вычислений, но и с целью создания более читабельных, логичных программ. Наибольший

эффект от введения многопоточной обработки достигается на разных процессорах действительно параллельно (а не псевдопараллельно).

Вытесняющие и невытесняющие алгоритмы планирования

С самых общих позиций все множество алгоритмов планирования можно разделить на два класса: вытесняющие и невытесняющие.

Невытесняющие (non-preemptive) алгоритмы основаны на том, что активному потоку позволяется выполняться, пока он сам, по собственной инициативе, не отдаст управление операционной системе для того, чтобы та выбрала из очереди другой готовый к выполнению поток.

Вытесняющие (preemptive) алгоритмы – это такие способы планирования потоков, в которых решение о переключении процессора с выполнения одного потока на выполнение другого потока принимается операционной системой, а не активной задачей.

Основным различием между вытесняющими и невытесняющими алгоритмами является степень централизации механизма планирования потоков. При вытесняющем мультипрограммировании функции планирования потоков целиком сосредоточены в операционной системе и программист пишет свое приложение, не заботясь о том, что оно будет выполняться одновременно с другими задачами. При этом операционная система выполняет следующие функции: определяет момент снятия с выполнения активного потока, запоминает его контекст, выбирает из очереди готовых потоков следующий, запускает новый поток на выполнение, загружая его контекст.

При невытесняющем мультипрограммировании механизм планирования распределен между операционной системой и прикладными программами. Прикладная программа, получив управление от операционной системы, сама определяет момент завершения очередного цикла своего выполнения и только затем передает управление ОС с помощью какого-либо системного вызова. ОС формирует очереди потоков и выбирает в соответствии с некоторым правилом (например, с учетом приоритетов) следующий поток на выполнение. Такой механизм создает проблемы как для пользователей, так и для разработчиков приложений. Для пользователей это означает, что управление системой теряется на произвольный период времени, который определяется приложением (а не пользователем). Если приложение тратит слишком много времени на выполнение какой-либо работы, например на форматирование диска, пользователь не может переключиться с этой задачи на другую задачу, например на текстовый редактор, в то время как форматирование продолжалось бы в фоновом режиме.

Однако распределение функций планирования потоков между системой и приложениями не всегда является недостатком, а при определенных условиях может быть и преимуществом, потому что дает возможность разработчику приложений самому проектировать алгоритм планирования, наиболее подходящий для данного фиксированного набора задач. Так как разработчик сам определяет в программе момент возвращения управления, то при этом исключаются нерациональные прерывания программ в «неудобные» для них моменты времени. Кроме того, легко разрешаются проблемы совместного использования данных: задача во время каждого цикла выполнения использует их монопольно и уверена, что на протяжении этого периода никто другой не изменит данные. Существенным преимуществом невытесняющего планирования является более высокая скорость переключения с потока на поток. Почти во всех современных операционных системах, ориентированных на высокопроизводительное выполнение приложений (UNIX, Windows NT/2000, OS/2, VAX/VMS), реализованы вытесняющие алгоритмы планирования потоков (процессов). В последнее время дошла очередь и до ОС класса настольных систем, например OS/2 Warp и Windows.

Мультипрограммирование на основе прерываний

Прерывания являются основной движущей силой любой операционной системы. Отключите систему прерываний – и «жизнь» в операционной системе немедленно остановится. Периодические прерывания от таймера вызывают смену процессов в мультипрограммной ОС, а прерывания от устройств ввода–вывода управляют потоками данных, которыми вычислительная система обменивается с внешним миром.

Как верно было замечено: «Прерывания названы так весьма удачно, поскольку они прерывают нормальную работу системы» (Скотт Максвелл. Ядро Linux в комментариях. – К. ДиаСофт, 2000). Другими словами, система прерываний переводит процессор на выполнение потока команд, отличного от того, который выполнялся до сих пор, с последующим возвратом к исходному коду. Из сказанного можно сделать вывод о том, что механизм прерываний очень похож на механизм выполнения процедур. Это на самом деле так, хотя между этими механизмами имеется важное отличие. Переход по команде происходит в заранее определенных программистом точках программы в зависимости от исходных данных, обрабатываемых программой. Прерывание же происходит в произвольной точке потока команд программы, которую программист не может прогнозировать. Прерывание возникает либо в зависимости от внешних по отношению к

процессу выполнения программы событий, либо при появлении непредвиденных аварийных ситуаций в процессе выполнения данной программы. Сходство же прерываний с процедурами состоит в том, что в обоих случаях выполняется некоторая подпрограмма, обрабатывающая специальную ситуацию, а затем продолжается выполнение основной ветви программы.

Назначение и типы прерываний

В зависимости от источника прерывания делятся на три больших класса:

- внешние;
- внутренние;
- программные.

Внешние прерывания могут возникать в результате действий пользователя или оператора за терминалом или же в результате поступления сигналов от аппаратных устройств (сигналов завершения операций ввода–вывода, вырабатываемых контроллерами внешних устройств компьютера, такими, как принтер или накопитель на жестких дисках, или же сигналов от датчиков управляемых компьютером технических объектов). Внешние прерывания называют также *аппаратными,* отражая тот факт, что прерывание возникает вследствие подачи некоторой аппаратурой (например, контроллером принтера) электрического сигнала, который передается (возможно, проходя через другие блоки компьютера, например контроллер прерываний) на специальный вход прерывания процессора.

Внутренние прерывания, называемые также *исключениями* (exeption), происходят синхронно выполнению программы при появлении аварийной ситуации в ходе исполнения некоторой инструкции программы. Примерами исключений являются деление на нуль, ошибки защиты памяти, обращения по несуществующему адресу, попытка выполнить привилегированную инструкцию в пользовательском режиме и т. п. Исключения возникают непосредственно в ходе выполнения тактов команды («внутри» выполнения).

Программные прерывания отличаются от предыдущих двух классов тем, что они по своей сути не являются «истинными» прерываниями. Программное прерывание возникает при выполнении особой команды процессора, выполнение которой имитирует прерывание, то есть переход на новую последовательность инструкций. Причины использования программных прерываний вместо обычных инструкций

вызова процедур будут изложены ниже, после рассмотрения механизма прерываний.

Прерываниям приписывается *приоритет*, с помощью которого они ранжируются по степени важности и срочности. О прерываниях, имеющих одинаковое значение приоритета, говорят, что они относятся к одному уровню приоритета.

Прерывания обычно обрабатываются модулями ОС, так как действия, выполняемые по прерыванию, относятся к управлению разделяемыми ресурсами ВС. Процедуры, вызываемые по прерываниям, обычно называют обработчиками прерываний, или процедурами обслуживания прерываний. Аппаратные прерывания обрабатываются драйверами соответствующих внешних устройств, исключения – специальными модулями ядра, а программные прерывания – процедурами ОС, обслуживающими системные вызовы. Кроме этих модулей, в операционной системе может находиться так называемый диспетчер прерываний, который координирует работу отдельных обработчиков прерываний.

Механизм прерываний

Механизм прерываний поддерживается аппаратными средствами компьютера и программными средствами операционной системы. Аппаратная поддержка прерываний имеет свои особенности, зависящие от типа процессора и других аппаратных компонентов, передающих сигнал запроса прерывания от внешнего устройства к процессору (таких, как контроллер внешнего устройства, шины подключения внешних устройств, контроллер прерываний, являющийся посредником между сигналами шины и сигналами процессора). Существуют два основных способа, с помощью которых шины выполняют прерывания: *векторный* (vectored) и *опрашиваемый* (polled). В обоих способах процессору предоставляется информация об уровне приоритета прерывания на шине подключения внешних устройств. В случае векторных прерываний в процессор передается также информация о начальном адресе программы обработки возникшего прерывания – обработчика прерываний.

Устройствам, которые используют векторные прерывания, назначается вектор прерываний. Он представляет собой электрический сигнал, выставляемый на соответствующие шины процессора и несущий в себе информацию об определенном, закрепленном за данным устройством номере, который идентифицирует соответствующий обработчик прерываний. Этот вектор может быть фиксированным, конфигурируемым (например, с использованием переключателей) или

программируемым. При получении сигнала запроса прерывания процессор выполняет специальный цикл подтверждения прерывания, в котором устройство должно идентифицировать себя. В течение этого цикла устройство отвечает, выставляя на шину вектор прерываний. Затем процессор использует этот вектор для нахождения обработчика данного прерывания.

При использовании опрашиваемых прерываний процессор получает от запросившего прерывание устройства только информацию об уровне приоритета прерывания. С каждым уровнем прерываний может быть связано несколько устройств и соответственно несколько программ – обработчиков прерываний. При возникновении прерывания процессор должен определить, какое устройство из тех, которые связаны с данным уровнем прерываний, действительно запросило прерывание. Это достигается вызовом всех обработчиков прерываний для данного уровня приоритета, пока один из обработчиков не подтвердит, что прерывание пришло от обслуживаемого им устройства. Если же с каждым уровнем прерываний связано только одно устройство, то определение нужной программы обработки прерывания происходит немедленно, как и при векторном прерывании.

Механизм прерываний некоторой аппаратной платформы может сочетать векторный и опрашиваемый типы прерываний. Типичным примером такой реализации является платформа персональных компьютеров на основе процессоров Intel Pentium. Шины PCI, используемые в этой платформе в качестве шин подключения внешних устройств, они поддерживают механизм опрашиваемых прерываний. Контроллеры периферийных устройств выставляют на шину не вектор, а сигнал запроса прерывания определенного уровня IRQ. Однако в процессоре Pentium система прерываний является векторной. Вектор прерываний в процессор Pentium поставляет контроллер прерываний, который отображает поступающий от шины сигнал IRQ на определенный номер вектора.

Вектор прерываний, передаваемый в процессор, представляет собой целое число в диапазоне от 0 до 255, указывающее на одну из 256 программ обработки прерываний, адреса которых хранятся в таблице обработчиков прерываний. В том случае, когда к каждой линии IRQ подключается только одно устройство, процедура обработки прерываний работает так, как если бы система прерываний была чисто векторной, то есть процедура не выполняет никаких дополнительных опросов для выяснения того, какое именно устройство запросило прерывание. Однако при совместном использовании одного уровня IRQ несколькими устройствами программа обработки прерываний должна

работать в соответствии со схемой опрашиваемых прерываний, то есть дополнительно выполнить опрос всех устройств, подключенных к данному уровню IRQ.

Механизм прерываний чаще всего поддерживает *приоритезацию* и *маскирование* прерываний. Приоритезация означает, что все источники прерываний делятся на классы и каждому классу назначается свой уровень приоритета запроса на прерывание. Приоритеты могут обслуживаться как относительные и абсолютные. Обслуживание запросов прерываний по схеме с относительными приоритетами заключается в том, что при одновременном поступлении запросов прерываний из разных классов выбирается запрос, имеющий высший приоритет. Однако в дальнейшем при обслуживании этого запроса процедура обработки прерывания уже не откладывается даже в том случае, когда появляются более приоритетные запросы – решение о выборе нового запроса принимается только в момент завершения обслуживания очередного прерывания. Если же более приоритетным прерываниям разрешается приостанавливать работу процедур обслуживания менее приоритетных прерываний, то это означает, что работает схема приоритезации с абсолютными приоритетами.

Если процессор (или компьютер, когда поддержка приоритезации прерываний вынесена во внешний по отношению к процессору блок) работает по схеме с абсолютными приоритетами, то он поддерживает в одном из своих внутренних регистров переменную, фиксирующую уровень приоритета обслуживаемого в данный момент прерывания. При поступлении запроса из определенного класса его приоритет сравнивается с текущим приоритетом процессора, и если приоритет запроса выше, то текущая процедура обработки прерываний вытесняется, а по завершении обслуживания нового прерывания происходит возврат к прерванной процедуре.

Обобщенно последовательность действий аппаратных и программных средств по обработке прерывания можно описать следующим образом:

1. При возникновении сигнала (для аппаратных прерываний) или условия (для внутренних прерываний) прерывания происходит первичное аппаратное распознавание типа прерывания. Если прерывания данного типа в настоящий момент запрещены (приоритетной схемой или механизмом маскирования), то процессор продолжает поддерживать естественный ход выполнения команд. В противном случае, в зависимости от поступившей в процессор информации (уровень прерывания, вектор прерывания или тип условия внутреннего прерывания), происходит автоматический вызов

процедуры обработки прерывания, адрес которой находится в специальной таблице операционной системы, размещаемой либо в регистрах процессора, либо в определенном месте оперативной памяти.

2. Автоматически сохраняется некоторая часть контекста прерванного потока, которая позволит ядру возобновить исполнение потока процесса после обработки прерывания. В это подмножество обычно включаются значения счетчика команд, слова состояния машины, хранящего признаки основных режимов работы процессора (пример такого слова – регистр EFLA6S в Intel Pentium), а также нескольких регистров общего назначения, которые требуются программе обработки прерывания. Может быть сохранен и полный контекст процесса, если ОС обслуживает данное прерывание со сменой процесса. Однако в общем случае это не обязательно, часто обработка прерываний выполняется без вытеснения текущего процесса.

3. Одновременно с загрузкой адреса процедуры обработки прерываний в счетчик команд может автоматически выполняться загрузка нового значения слова состояния машины (или другой системной структуры, например селектора кодового сегмента в процессоре Pentium), которое определяет режимы работы процессора при обработке прерывания, в том числе работу в привилегированном режиме. В некоторых моделях процессоров переход в привилегированный режим за счет смены состояния машины при обработке прерывания является единственным способом смены режима. Прерывания практически во всех мультипрограммных ОС обрабатываются в привилегированном режиме модулями ядра, так как при этом обычно нужно выполнить ряд критических операций, от которых зависит жизнеспособность системы, – управлять внешними устройствами, перепланировать потоки и т. п.

4. Временно запрещаются прерывания данного типа, чтобы не образовалась очередь вложенных друг в друга потоков одной и той же процедуры. Детали выполнения этой операции зависят от особенностей аппаратной платформы, например, может использоваться механизм маскирования прерываний. Многие процессоры автоматически устанавливают признак запрета прерываний в начале цикла обработки прерывания, в противном случае это делает программа обработки прерываний.

5. После того как прерывание обработано ядром операционной системы, прерванный контекст восстанавливается, и работа потока возобновляется с прерванного места. Часть контекста восстанавливается аппаратно по команде возврата из прерываний (например, адрес

следующей команды и слово состояния машины), а часть – программным способом с помощью явных команд извлечения данных из стека. При возврате из прерывания блокировка повторных прерываний данного типа снимается.

Программные прерывания

Программное прерывание реализует один из способов перехода на подпрограмму с помощью специальной инструкции процессора, такой, как INT в процессорах Intel Pentium, trap в процессорах Motorola, syscall в процессорах MIPS или Ticc в процессорах SPARC. При выполнении команды программного прерывания процессор отрабатывает ту же последовательность действий, что и при возникновении внешнего или внутреннего прерывания, но только происходит это в предсказуемой точке программы – там, где программист поместил данную команду.

Практически все современные процессоры имеют в системе команд инструкции программных прерываний. Одной из причин появления инструкций программных прерываний в системе команд процессоров является то, что их использование часто приводит к более компактному коду программ по сравнению с использованием стандартных команд выполнения процедур. Это объясняется тем, что разработчики процессора обычно резервируют для обработки прерываний небольшое число возможных подпрограмм, так что длина операнда в команде программного прерывания, который указывает на нужную подпрограмму, меньше, чем в команде перехода на подпрограмму. Например, в процессоре x86 предусмотрена возможность применения 256 программ обработки прерываний, поэтому в инструкции INT операнд имеет длину в один байт (а инструкция INT 3, которая предназначена для вызова отладчика, вся имеет длину один байт). Значение операнда команды INT просто является индексом в таблице из 256 адресов подпрограмм обработки прерываний, один из которых и используется для перехода по команде INT. В результате программные прерывания часто используются для выполнения ограниченного количества вызовов функций ядра операционной системы, то есть системных вызовов.

Диспетчеризация и приоритезация прерываний в ОС

Операционная система должна играть активную роль в организации обработки прерываний. Прерывания выполняют очень полезную для вычислительной системы функцию – они позволяют реагировать на асинхронные но отношению к вычислительному процессу события. В то же время прерывания создают дополнительные трудности для ОС в организации вычислительного процесса. Эти трудности связаны с

непредвиденными переходами управления от одной процедуры к другой, возникающими в результате прерываний от контроллеров внешних устройств. Возможно также возникновение в непредвиденные моменты времени исключений, связанных с ошибками во время выполнения инструкций. Усложняют задачу планирования вычислительных работ и запросы на выполнение системных функций (системные вызовы) от пользовательских приложений, выполняемые с помощью программных прерываний. Сами модули ОС также часто вызывают друг друга с помощью программных прерываний, еще больше запутывая картину вычислительного процесса.

Операционная система не может терять контроль над ходом выполнения системных процедур, вызываемых по прерываниям. Она должна упорядочивать их во времени так же, как планировщик упорядочивает многочисленные пользовательские потоки. Кроме того, сам планировщик потоков является системной процедурой, вызываемой по прерываниям (аппаратным – от таймера или контроллера устройства ввода–вывода, или программным – от приложения или модуля ОС). Поэтому правильное планирование процедур, вызываемых по прерываниям, является необходимым условием правильного планирования пользовательских потоков. В противном случае, в системе могут возникать, например, такие ситуации, когда операционная система будет длительное время заниматься не требующей мгновенной реакции задачей – архивированием данных на внешнем носителе – в то время, когда высокоскоростной диск будет простаивать и тормозить работу многочисленных приложений, обменивающихся данными с этим диском. Еще один пример такой ситуации иллюстрирует рис. 4.11. В данном случае обработчик прерываний принтера блокирует на длительное время обработку прерывания от таймера, в результате чего системное время на некоторое время «замирает» и поток 2, критически важный для пользователя, не получает управление в запланированное время. Остроту проблемы несколько смягчает то обстоятельство, что во многих случаях обработка прерывания связана с выполнением всего нескольких операций ввода–вывода и поэтому имеет очень небольшую продолжительность. Тем не менее, ОС всегда должна контролировать ситуацию и выполнять критичную работу вовремя, а не полагаться на волю случая.

Для упорядочения работы обработчиков прерываний в операционных системах применяется тот же механизм, что и для упорядочения работы пользовательских процессов – механизм приоритетных очередей. Все источники прерываний обычно делятся на

несколько классов, причем каждому классу присваивается приоритет. В операционной системе выделяется программный модуль, который занимается диспетчеризацией обработчиков прерываний. Этот модуль в разных ОС называется по-разному, но для определенности будем его называть диспетчером прерываний.

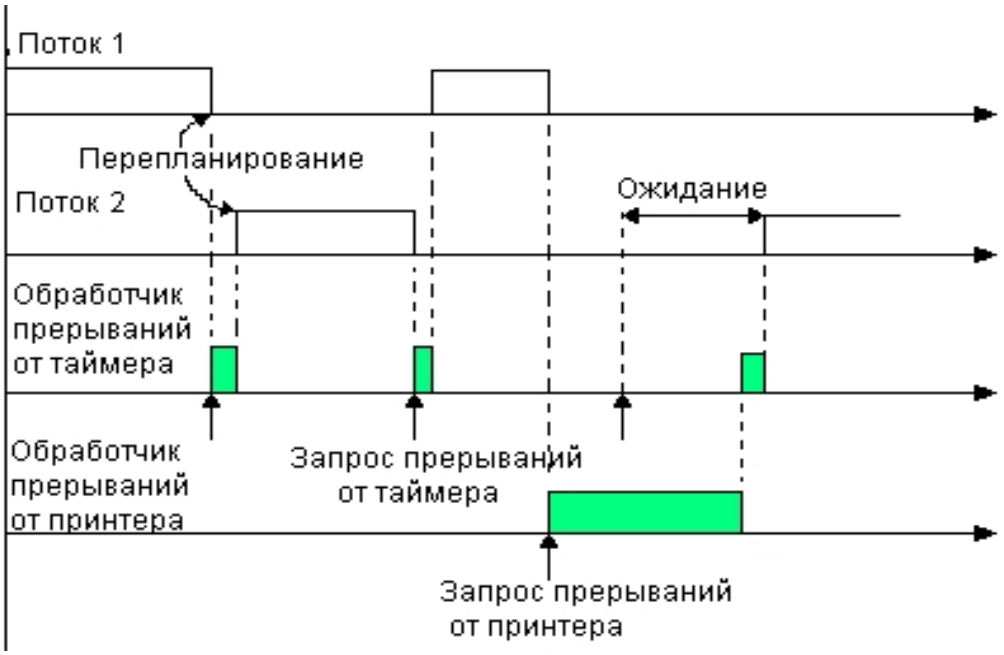


Рис. 4.11. Неупорядоченная обработка прерываний

При возникновении прерывания диспетчер прерываний вызывается первым. Он запрещает ненадолго все прерывания, а затем выясняет причину прерывания. После этого диспетчер сравнивает назначенный данному источнику прерывания приоритет и сравнивает его с текущим приоритетом потока команд, выполняемого процессором. В этот момент времени процессор уже может выполнять инструкции другого обработчика прерываний, также имеющего некоторый приоритет. Если приоритет нового запроса выше текущего, то выполнение текущего обработчика приостанавливается, и он помещается в соответствующую очередь обработчиков прерываний. В противном случае, в очередь помещается обработчик нового запроса. Итак, *управление ресурсами* вычислительной системы с целью наиболее эффективного их использования *является назначением ОС*. Управление ресурсами включает решение следующих общих, не зависящих от типа ресурса задач:

- планирование ресурса, то есть определение, какому процессу, когда и в каком количестве (если ресурс может выделяться частями) следует выделить данный ресурс;

- удовлетворение запросов на ресурсы;
- отслеживание состояния и учет использования ресурса, то есть поддержание оперативной информации о том, занят или свободен ресурс и какая доля ресурса уже распределена;
- разрешение конфликтов между процессами.

Для решения этих общих задач управления ресурсами разные ОС используют различные алгоритмы, особенности которых, в конечном счете, и определяют облик ОС в целом, включая характеристики производительности, область применения и даже пользовательский интерфейс.

Задача организации эффективного совместного использования ресурсов несколькими процессами является весьма сложной, и сложность эта порождается в основном случайным характером возникновения запросов на потребление ресурсов. В мультипрограммной системе образуются очереди заявок от одновременно выполняемых программ к разделяемым ресурсам компьютера: процессору, странице памяти, к принтеру, к диску. Операционная система организует обслуживание этих очередей по разным алгоритмам: в порядке поступления, на основе приоритетов, кругового обслуживания и т. д.

Наиболее важными подсистемами управления ресурсами являются *подсистемы управления процессами, памятью, файлами и внешними устройствами*, а подсистемами, общими для всех ресурсов, являются *подсистемы пользовательского интерфейса, защиты данных и администрирования.*

На протяжении периода существования процесса его выполнение может быть многократно прервано и продолжено.

Операционная система берет на себя также функции синхронизации процессов, позволяющие процессу приостанавливать свое выполнение до наступления какого-либо события в системе, например завершения операции ввода–вывода, осуществляемой по его запросу операционной системой.

В операционной системе нет однозначного соответствия между процессами и программами. Один и тот же программный файл может породить несколько параллельно выполняемых процессов, а процесс может в ходе своего выполнения сменить программный файл и начать выполнять другую программу.

Таким образом, подсистема управления процессами планирует выполнение процессов, то есть распределяет процессорное время между несколькими одновременно существующими в системе процессами,

занимается созданием и уничтожением процессов, обеспечивает процессы необходимыми системными ресурсами, поддерживает синхронизацию процессов, а также обеспечивает взаимодействие между процессами.

Подсистемы управления ресурсами

Память является для процесса таким же важным ресурсом, как и процессор, так как процесс может выполняться процессором только в том случае, если его коды и данные (не обязательно все) находятся в оперативной памяти.

Управление памятью включает распределение имеющейся физической памяти между всеми существующими в системе в данный момент процессами, загрузку кодов и данных процессов в отведенные им области памяти, настройку адресно-зависимых частей кодов процесса на физические адреса выделенной области, а также защиту областей памяти каждого процесса. Одним из наиболее популярных способов управления памятью в современных ОС является так называемая *виртуальная память.* Наличие в ОС механизма виртуальной памяти позволяет программисту писать программу так, как будто в его распоряжении имеется однородная оперативная память большого объема, часто существенно превышающего объем имеющейся физической памяти. В действительности все данные, используемые программой, хранятся на диске и при необходимости частями (сегментами или страницами) отображаются в физическую память.

Защита памяти – это избирательная способность предохранять выполняемую задачу от записи или чтения памяти, назначенной другой задаче. Правильно написанные программы не пытаются обращаться к памяти, назначенной другим. Однако реальные программы часто содержат ошибки, в результате которых такие попытки иногда предпринимаются. Средства защиты памяти, реализованные в операционной системе, должны пресекать несанкционированный доступ процессов к чужим областям памяти.

Таким образом, функциями ОС по управлению памятью являются: отслеживание свободной и занятой памяти; выделение памяти процессам и освобождение памяти при завершении процессов; защита памяти; вытеснение процессов из оперативной памяти на диск, когда размеры основной памяти недостаточны для размещения в ней всех процессов, и возвращение их в оперативную память, когда в ней освобождается место, а также настройка адресов программы на конкретную область физической памяти.

Управление файлами и внешними устройствами. Способность ОС к «экранированию» сложностей реальной аппаратуры очень ярко проявляется в одной из основных подсистем ОС – файловой системе. Операционная система виртуализирует отдельный набор данных, хранящихся на внешнем накопителе, в виде файла – простой неструктурированной последовательности байтов, имеющей символьное имя. Для удобства работы с данными файлы группируются в каталоги, которые, в свою очередь, образуют группы – каталоги более высокого уровня. Пользователь может с помощью ОС выполнять над файлами и каталогами такие действия, как поиск по имени, удаление, вывод содержимого на внешнее устройство (например, на дисплей), изменение и сохранение содержимого.

Чтобы представить большое количество наборов данных, разбросанных случайным образом по цилиндрам и поверхностям дисков различных типов, в виде хорошо всем знакомой и удобной иерархической структуры файлов и каталогов, операционная система должна решить множество задач. Файловая система ОС выполняет преобразование символьных имен файлов, с которыми работает пользователь или прикладной программист, в физические адреса данных на диске, организует совместный доступ к файлам, защищает их от несанкционированного доступа. При выполнении своих функций файловая система тесно взаимодействует с подсистемой управления внешними устройствами, которая по запросам файловой системы осуществляет передачу данных между дисками и оперативной памятью.

Подсистема управления внешними устройствами, называемая также подсистемой ввода–вывода, исполняет роль интерфейса ко всем устройствам, подключенным к компьютеру. Спектр этих устройств очень обширен. Номенклатура выпускаемых накопителей на жестких, гибких и оптических дисках принтеров, сканеров, мониторов, плоттеров, модемов, сетевых адаптеров и более специальных устройств ввода–вывода, таких как, например, аналого-цифровые преобразователи, может насчитывать сотни моделей. Программа, управляющая конкретной моделью ВУ и учитывающая все его особенности, обычно называется *драйвером* этого устройства (от английского drive – управлять, вести). Драйвер может управлять единственной моделью устройства, например модемом U–1496E компании ZyXEL, или же группой устройств определенного типа, например любыми Hayes-совместимыми модемами. Для пользователя очень важно, чтобы ОС включала как можно больше разнообразных драйверов, так как это гарантирует возможность подключения к

компьютеру большого числа внешних устройств различных производителей.

Поддержание высокоуровневого унифицированного интерфейса прикладного программирования к разнородным устройствам ввода–вывода является одной из наиболее важных задач ОС. Со времени появления ОС UNIX такой унифицированный интерфейс в большинстве операционных систем строится на основе концепции файлового доступа. Эта концепция заключается в том, что обмен с любым внешним устройством выглядит как обмен с файлом, имеющим имя и представляющим собой неструктурированную последовательность байтов. В качестве файла может выступать как реальный файл на диске, так и алфавитно-цифровой терминал, печатающее устройство или сетевой адаптер. Здесь мы опять имеем дело со свойством операционной системы подменять реальную аппаратуру удобными для пользователя и программиста абстракциями.

Защита данных и администрирование. Безопасность данных вычислительной системы обеспечивается средствами отказоустойчивости ОС, направленными на защиту от сбоев и отказов аппаратуры и ошибок программного обеспечения, а также средствами защиты от несанкционированного доступа. В последнем случае ОС защищает данные от ошибочного или злонамеренного поведения пользователей системы.

Первым рубежом обороны при защите данных от несанкционированного доступа является процедура логического входа. Операционная система должна убедиться, что в систему пытается войти пользователь, вход которого разрешен администратором. Функции защиты ОС вообще очень тесно связаны с функциями администрирования, так как именно администратор определяет права пользователей при их обращении к разным ресурсам системы – файлам, каталогам, принтерам, сканерам и т. п. Кроме того, администратор ограничивает возможности пользователей в выполнении тех или иных системных действий. Например, пользователю может быть запрещено выполнять процедуру завершения работы ОС, устанавливать системное время, завершать чужие процессы, создавать учетные записи пользователей, изменять права доступа к некоторым каталогам и файлам. Администратор может также урезать возможности пользовательского интерфейса, убрав, например, некоторые пункты из меню операционной системы выводимого на дисплей пользователя.

Важным средством защиты данных являются функции аудита ОС, заключающиеся в фиксации всех событий, от которых зависит безопасность системы. Например, попытки удачного и неудачного

логического входа в систему, операции доступа к некоторым каталогам и файлам, использование принтеров и т. п. Список событий, которые необходимо отслеживать, определяет администратор ОС.

Поддержка отказоустойчивости реализуется операционной системой, как правило, на основе резервирования. Чаще всего в функции ОС входит поддержание нескольких копий данных на разных дисках или разных дисковых накопителях. Резервируются также принтеры и другие устройства ввода–вывода. При отказе одного из избыточных устройств операционная система должна быстро и прозрачным для пользователя образом произвести реконфигурацию системы и продолжить работу с резервным устройством.

Особым случаем обеспечения отказоустойчивости является использование нескольких процессоров, то есть мультипроцессирование, когда система продолжает работу при отказе одного из процессоров, хотя и с меньшей производительностью. (Необходимо отметить, что многие ОС использует мультипроцессорную конфигурацию компьютера только для ускорения работы и при отказе одного из процессоров прекращают работу.)

Поддержка отказоустойчивости также входит в обязанности системного администратора. В состав ОС обычно входят утилиты, позволяющие администратору выполнять регулярные операции резервного копирования для обеспечения быстрого восстановления важных данных.

Интерфейс прикладного программирования. Прикладные программисты используют в своих приложениях обращения к ОС, когда для выполнения тех или иных действий им требуется особый статус, которым обладает только операционная система. Например, в большинстве современных ОС все действия, связанные с управлением аппаратными средствами компьютера, может выполнять только ОС. Помимо этих функций прикладной программист может воспользоваться набором сервисных функций ОС, которые упрощают написание приложений. Функции такого типа реализуют универсальные действия, часто требующиеся в различных приложениях, такие, например, как обработка текстовых строк. Эти функции могли бы быть выполнены и самим приложением, однако гораздо проще использовать уже готовые, отлаженные процедуры, включенные в состав операционной системы. В то же время даже при наличии в ОС соответствующей функции программист может реализовать ее самостоятельно в рамках приложения, если предложенный операционной системой вариант его не вполне устраивает.

Возможности операционной системы доступны прикладному программисту в виде набора функций, называющегося интерфейсом прикладного программирования (Application Programming Interface, API). От конечного пользователя эти функции скрыты за оболочкой алфавитно-цифрового или графического пользовательского интерфейса.

Для разработчиков приложений все особенности конкретной операционной системы представлены особенностями ее API. Поэтому операционные системы с различной внутренней организацией, но с одинаковым набором функций API кажутся им одной и той же ОС, что упрощает стандартизацию операционных систем и обеспечивает переносимость приложений между внутренне различными ОС, соответствующими определенному стандарту на API. Например, следование общим стандартам API UNIX, одним из которых является стандарт Posix, позволяет говорить о некоторой обобщенной операционной системе UNIX, хотя многочисленные версии этой ОС от разных производителей иногда существенно отличаются внутренней организацией.

Приложения выполняют обращения к функциям API с помощью системных вызовов. Способ, которым приложение получает услуги операционной системы, очень похож на вызов подпрограмм. Информация, нужная ОС и состоящая обычно из идентификатора команды и данных, помещается в определенное место памяти, в регистры и/или стек. Затем управление передается операционной системе, которая выполняет требуемую функцию и возвращает результаты через память, регистры или стеки. Если операция проведена неудачно, то результат включает индикацию ошибки.

Способ реализации системных вызовов зависит от структурной организации ОС, которая, в свою очередь, тесно связана с особенностями аппаратной платформы. Кроме того, он зависит от языка программирования. При использовании ассемблера программист устанавливает значения регистров и/или областей памяти, а затем выполняет специальную инструкцию вызова сервиса или программного прерывания для обращения к некоторой функции ОС. При использовании языков высокого уровня функции ОС вызываются тем же способом, что и написанные пользователем подпрограммы, требуя задания определенных аргументов в определенном порядке.

Пользовательский интерфейс. Операционная система должна обеспечивать удобный интерфейс не только для прикладных программ, но и для человека, работающего за терминалом. Этот человек может быть конечным пользователем, администратором ОС или программистом.

Современные ОС поддерживают развитые функции пользовательского интерфейса для интерактивной работы за терминалами двух типов: алфавитно-цифровыми (или командными) и графическими.

При работе за алфавитно-цифровым терминалом пользователь имеет в своем распоряжении систему команд, мощность которой отражает функциональные возможности данной ОС. Команды могут вводиться не только в интерактивном режиме с терминала, но и считываться из так называемого командного файла, содержащего некоторую последовательность команд.

Программный модуль ОС, ответственный за чтение отдельных команд или же последовательность команд из командного файла, иногда называют *командным интерпретатором*.

Ввод команды может быть упрощен, если операционная система поддерживает графический пользовательский интерфейс. В этом случае пользователь для выполнения нужного действия с помощью мыши выбирает на экране нужный пункт меню или графический символ.

4.4.2. Ввод-вывод и файловая система

Рассмотрим подробнее одну из главных задач ОС – обеспечение обмена данными между приложениями и периферийными устройствами компьютера. Собственно, ради выполнения этой задачи и были разработаны первые системные программы, послужившие прототипами операционных систем.

В современной ОС функции обмена данными с периферийными устройствами выполняет подсистема ввода-вывода. Клиентами этой подсистемы являются не только пользователи и приложения, но и некоторые компоненты самой ОС, которым требуется получение системных данных или их вывод, например, подсистеме управления процессами при смене активного процесса необходимо записать на диск контекст приостанавливаемого процесса и считать с диска контекст активизируемого процесса.

Основными компонентами подсистемы ввода–вывода являются *драйверы*, управляющие внешними устройствами, и *файловая система*.

Файловая система ввиду ее сложности, специфичности и важности как основного хранилища всей информации вычислительной системы заслуживает рассмотрения в отдельной главе. Тем не менее, здесь файловая система рассматривается совместно с другими компонентами подсистемы ввода-вывода по двум причинам.

Во-первых, файловая система активно использует остальные части подсистемы ввода-вывода, а во-вторых, модель файла лежит в основе большинства механизмов доступа к устройствам, используемых в современной подсистеме ввода-вывода.

Задачи ОС по управлению файлами и устройствами

Подсистема ввода-вывода (Input–Output Subsystem) мультипрограммной ОС при обмене данными с внешними устройствами компьютера должна решать ряд общих задач, из которых наиболее важными являются следующие:

- организация параллельной работы устройств ввода–вывода и процессора;
- согласование скоростей обмена и кэширование данных;
- разделение устройств и данных между процессами;
- обеспечение удобного логического интерфейса между устройствами и остальной частью системы;
- поддержка широкого спектра драйверов с возможностью простого включения в систему нового драйвера;
- динамическая загрузка и выгрузка драйверов;
- поддержка нескольких файловых систем;
- поддержка синхронных и асинхронных операций ввода–вывода.

*Организация параллельной работы устройств ввода-вывода и процессора***.** Каждое устройство ввода-вывода вычислительной системы – диск, принтер, терминал и т. п. – снабжено специализированным блоком управления, называемым контроллером. Контроллер взаимодействует с драйвером – системным программным модулем, предназначенным для управления данным устройством. Контроллер периодически принимает от драйвера выводимую на устройство информацию, а также команды управления, которые говорят о том, что с этой информацией нужно сделать (например, вывести в виде текста в определенную область терминала или записать в определенный сектор диска). Под управлением контроллера устройство может некоторое время выполнять свои операции автономно, не требуя внимания со стороны центрального процессора

Процессы, происходящие в контроллерах, протекают в периоды между выдачами команд независимо от ОС. От подсистемы ввода–вывода требуется спланировать в реальном масштабе времени (в котором работают внешние устройства) запуск и приостановку большого количества разнообразных драйверов, обеспечив приемлемое время реакции каждого драйвера на независимые события контроллера. С другой стороны, необходимо минимизировать загрузку процессора

задачами ввода–вывода, оставив как можно больше процессорного времени на выполнение пользовательских потоков.

Данная задача является классической задачей планирования систем реального времени и обычно решается на основе многоуровневой приоритетной схемы обслуживания по прерываниям. Для обеспечения приемлемого уровня реакции все драйверы (или части драйверов) распределяются по нескольким приоритетным уровням в соответствии с требованиями ко времени реакции и временем использования процессора. Для реализации приоритетной схемы обычно задействуется общий диспетчер прерываний ОС.

Согласование скоростей обмена и кэширование данных. При обмене данными всегда возникает задача согласования скорости. Например, если один пользовательский процесс вырабатывает некоторые данные и передает их другому пользовательскому процессу через оперативную память, то в общем случае скорости генерации данных и их чтения не совпадают. Согласование скорости обычно достигается за счет буферизации данных в оперативной памяти и синхронизации доступа процессов к буферу.

В подсистеме ввода-вывода для согласования скоростей обмена также широко используется буферизация данных в оперативной памяти. В тех специализированных операционных системах, в которых обеспечение высокой скорости ввода-вывода является первоочередной задачей (управление в реальном времени, услуги сетевой файловой службы и т. п.), большая часть оперативной памяти отводится не подкоды прикладных программ, а под буферизацию данных. Однако буферизация только на основе оперативной памяти в подсистеме ввода-вывода оказывается недостаточной. Для таких случаев применяется спул-файл (от spool – шпулька, тоже буфер, только для ниток). Типичный пример применения спулинга дает организация вывода данных на принтер. Для печатаемых документов объем в несколько десятков мегабайт не редкость, поэтому для их временного хранения (а печать каждого документа занимает от нескольких минут до десятков минут) объема оперативной памяти явно недостаточно.

Другим решением этой проблемы является использование большой буферной памяти в контроллерах внешних устройств. Такой подход особенно полезен в тех случаях, когда помещение данных на диск слишком замедляет обмен (или когда данные выводятся на сам диск). Например, в контроллерах графических дисплеев применяется буферная память, соизмеримая по объему с оперативной, и это существенно ускоряет вывод графики на экран.

Буферизация данных позволяет не только согласовать скорости работы процессора и внешнего устройства, но и решить другую задачу – сократить количество реальных операций ввода-вывода за счет кэширования данных. Дисковый кэш является непременным атрибутом подсистем ввода-вывода практически всех операционных систем, значительно сокращая время доступа к хранимым данным.

Разделение устройств и данных между процессами. Устройства ввода–вывода могут предоставляться процессам как в монопольное, так и в совместное (разделяемое) использование. При этом ОС должна обеспечивать контроль доступа теми же способами, что и при доступе процессов к другим ресурсам вычислительной системы – путем проверки прав пользователя или группы пользователей, от имени которых действует процесс, на выполнение той или иной операции над устройством. Например, определенной группе пользователей последовательный порт разрешено захватывать в монопольное владение, а другим пользователям это запрещено.

Операционная система может контролировать доступ не только к устройству в целом, но и к отдельным порциям данных, хранимых или отображаемых этим устройством. Диск является типичным примером устройства, для которого важно контролировать доступ не к устройству в целом, а к отдельным каталогам и файлам. Так, в файловой системе обычно для каждого каталога и файла можно задать индивидуальные права доступа. Очевидно, что для организации совместного доступа к частям устройства или частям данных, хранящихся на нем, непременным условием является задание режима совместного использования устройства в целом.

При разделении устройства между процессами может возникнуть необходимость в разграничении порции данных двух процессов друг от друга. Обычно такая потребность возникает при совместном использовании так называемых последовательных устройств. Типичным представителем такого рода устройства является принтер, который не выделяется в монопольное владение процессам, и в то же время каждый документ должен быть напечатан в виде последовательного набора страниц. Для подобных устройств организуется очередь заданий на вывод, при этом каждое задание представляет собой порцию данных, которую нельзя разрывать, например документ для печати. Для хранения очереди заданий используется спул–файл, который одновременно согласует скорости работы принтера и оперативной памяти и позволяет организовать разбиение данных на логические порции. Так как спул–файл находится на разделяемом устройстве

прямого доступа, то процессы могут одновременно выполнять вывод на принтер, помещая данные в свой раздел спул-файла.

Обеспечение удобного логического интерфейса между устройствами и остальной частью системы.

Разнообразие устройств ввода-вывода делают особенно актуальной функцию ОС по созданию экранирующего логического интерфейса между периферийными устройствами и приложениями. Практически все современные операционные системы поддерживают в качестве основы такого интерфейса файловую модель периферийных устройств, когда любое устройство выглядит для прикладного программиста последовательным набором байт, с которым можно работать с помощью унифицированных системных вызовов (например, read и write), задавая имя файла-устройства и смещение от начала последовательности байт. Для поддержания такого интерфейса подсистема ввода-вывода должна проделать немалую работу, учитывая разницу в организации операций обмена данными, например, с жестким диском и графическим терминалом.

Привлекательность модели файла-устройства состоит в ее простоте и унифицированности для устройств любого типа, однако во многих случаях для программирования операций ввода-вывода некоторого устройства она является слишком бедной. Поэтому данная модель часто используется только в качестве базиса, над которым подсистема ввода-вывода строит более содержательную модель устройств конкретного типа. Подсистема ввода-вывода предоставляет, как правило, специфический интерфейс для вывода графической информации на дисплей или принтер для программирования операций сетевого обмена и т. п. При этом разработчик специфического интерфейса всегда может опираться на имеющийся базовый интерфейс.

Поддержка широкого спектра драйверов и простота включения нового драйвера в систему. Достоинством подсистемы ввода-вывода любой универсальной ОС является наличие разнообразного набора драйверов для наиболее популярных периферийных устройств. Прекрасно спланированная и реализованная операционная система может потерпеть неудачу на рынке только из-за того, что в ее состав не включен достаточный набор драйверов, и администраторы и пользователи вынуждены искать нужный им драйвер для имеющегося у них внешнего устройства у производителей оборудования или, что еще хуже, заниматься его разработкой. Именно в такой ситуации оказались пользователи первых версий OS/2, и, возможно, это обстоятельство послужило в свое время не последней причиной сдачи позиций этой неплохой операционной системы.

Чтобы операционная система не испытывала недостатка в драйверах, необходимо наличие четкого, удобного и открытого интерфейса между драйверами и другими компонентами ОС. Такой интерфейс нужен для того, чтобы драйверы писали не только непосредственные разработчики данной операционной системы, но и большая армия программистов по всему миру, в первую очередь – тех предприятий, которые выпускают внешние устройства для компьютеров. Открытость интерфейса драйверов, то есть доступность его описания для независимых разработчиков программного обеспечения, является необходимым условием успешного развития операционной системы.

Драйвер взаимодействует, с одной стороны, с модулями ядра ОС (модулями подсистемы ввода-вывода, модулями системных вызовов, модулями подсистем управления процессами и памятью и т. д.), а с другой стороны – с контроллерами внешних устройств. Поэтому существуют два типа интерфейсов: интерфейс «драйвер-ядро» (Driver Kernel Interface, DKI) и интерфейс «драйвер-устройство» (Driver Device Interface, DDF). Подсистема ввода-вывода может поддерживать несколько различных типов интерфейсов DKI/DDI, предоставляя специфический интерфейс для устройств определенного класса. Так, в ОС Windows NT для драйверов сетевых адаптеров существует интерфейс стандарта NDIS (Network Driver Interface Specification), в то время как драйверы сетевых транспортных протоколов взаимодействуют с верхними слоями сетевого программного обеспечения по интерфейсу TDI (Transport Driver Interface).

Для поддержки процесса разработки драйверов операционной системы обычно выпускается так называемый пакет DDK (Driver Development Kit), представляющий собой набор соответствующих инструментальных средств – библиотек, компиляторов и отладчиков.

Динамическая загрузка и выгрузка драйверов. Кроме проблемы разработки новых драйверов, существует также проблема включения драйвера в состав модулей работающей ОС, то есть динамической загрузки-выгрузки драйвера. Так как набор потенциально поддерживаемых данной ОС периферийных устройств всегда существенно шире набора устройств, которыми ОС должна управлять при установке на конкретной машине, то ценным свойством ОС является возможность динамически загружать в оперативную память требуемый драйвер (без останова ОС) и выгружать его после того, как потребность в поддержке устройства миновала, что может существенно сэкономить системную область памяти. Поддержка динамической

загрузки драйверов является практически обязательным требованием для современных универсальных операционных систем.

Поддержка нескольких файловых систем. Диски представляют особый род периферийных устройств, так как именно на них хранится большая часть как пользовательских, так и системных данных. Данные на дисках организуются в файловые системы, и свойства файловой системы во многом определяют свойства самой ОС – ее отказоустойчивость, быстродействие, максимальный объем хранимых данных. Популярность файловой системы часто приводит к ее миграции из «родной» ОС в другие операционные системы. Например, файловая система FAT появилась первоначально в MS–DOS, но затем была реализована в OS/2, семействе MS Windows и многих реализациях UNIX. Ввиду этого поддержка нескольких популярных файловых систем для подсистемы ввода-вывода также важна, как и поддержка широкого спектра периферийных устройств. Важно также, чтобы архитектура подсистемы ввода-вывода позволяла достаточно просто включать в ее состав новые типы файловых систем без необходимости переписывания кода. Обычно в операционной системе имеется специальный слой программного обеспечения, отвечающий за решение данной задачи, например слой VFS (Virtual File System) в версиях UNIX на основе кода System V Release 4.

Поддержка синхронных и асинхронных операций ввода–вывода

Операция ввода-вывода может выполняться по отношению к программному модулю, запросившему операцию, в синхронном или асинхронном режимах (рис. 4.12).

Смысл этих режимов тот же, что и для рассмотренных выше системных вызовов, – синхронный режим означает, что программный модуль приостанавливает свою работу до тех пор, пока операция ввода-вывода не будет завершена (рис. 4.12, *а*), а при асинхронном режиме программный модуль продолжает выполняться в мультипрограммном режиме одновременно с операцией ввода–вывода (рис. 4.12, *б*). Отличие же заключается в том, что операция ввода-вывода может быть инициирована не только пользовательским процессом – в этом случае операция выполняется в рамках системного вызова, но и кодом ядра, например кодом подсистемы виртуальной памяти для считывания отсутствующей в памяти страницы.

Подсистема ввода-вывода должна предоставлять своим клиентам (пользовательским процессам и кодам ядра) возможность выполнять как синхронные, так и асинхронные операции ввода–вывода, в зависимости от потребностей вызывающей стороны.

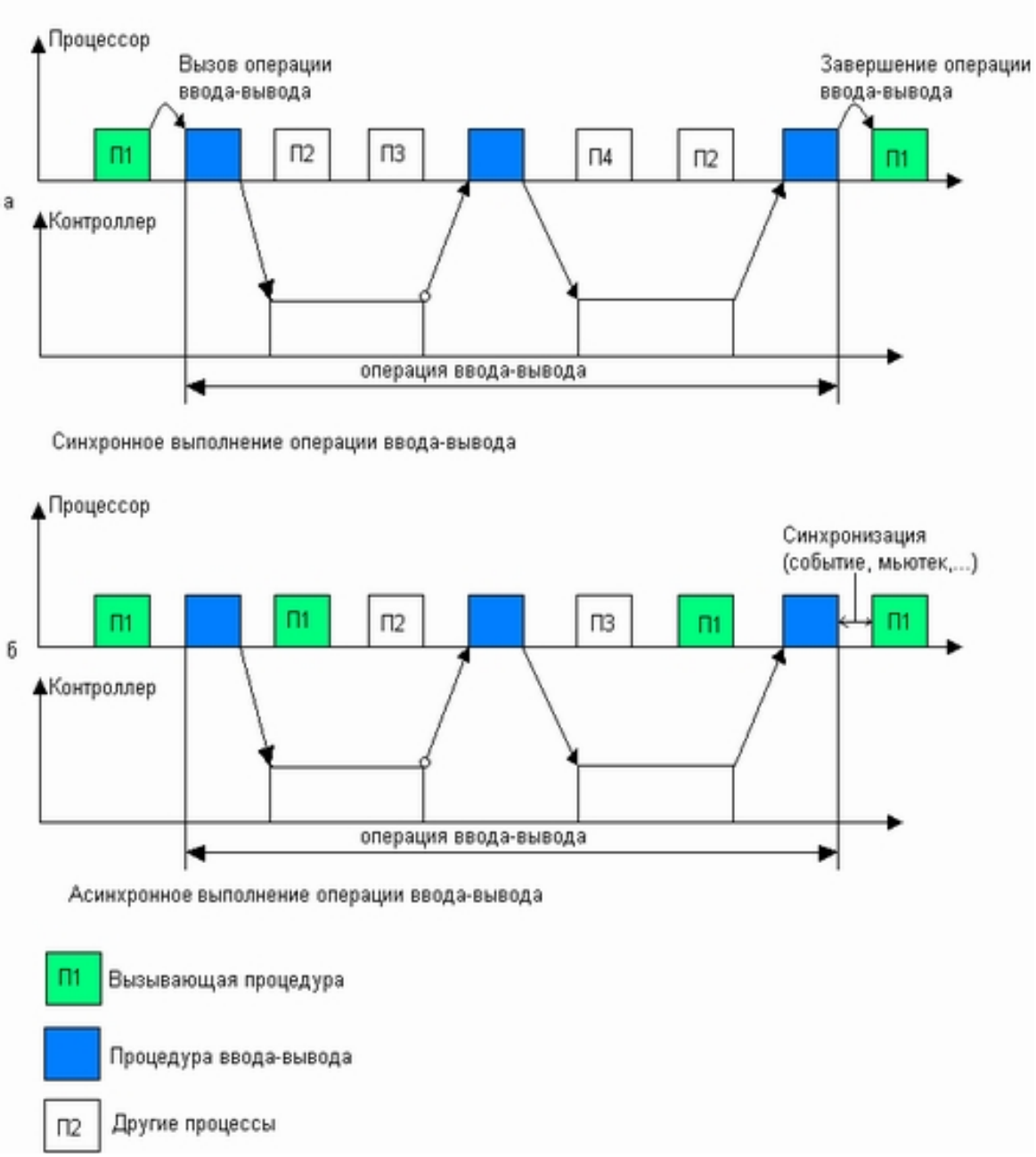


Рис. 4.12. Два режима выполнения операций ввода–вывода

Логическая организация файловой системы

Одной из основных задач операционной системы является предоставление удобств пользователю при работе с данными, хранящимися на дисках. Для этого ОС подменяет физическую структуру хранящихся данных некоторой удобной для пользователя логической моделью. Логическая модель файловой системы материализуется в виде дерева каталогов, выводимого на экран такими утилитами, как Norton или Windows Commander, или Windows Explorer, в символьных составных именах файлов, в командах работы с файлами.

Базовым элементом этой модели является файл, который так же, как и файловая система в целом, может характеризоваться как логической, так и физической структурой.

Цели и задачи файловой системы

Файл – это именованная область внешней памяти, в которую можно записывать и из которой можно считывать данные. Файлы хранятся в памяти, не зависящей от энергопитания, обычно – на магнитных дисках. Однако нет правил без исключения. Одним из таких исключений является так называемый электронный диск, когда в оперативной памяти создается структура, имитирующая файловую систему.

Основные цели использования файла перечислены ниже.

- Долговременное и надежное хранение информации. Долговременность достигается за счет использования запоминающих устройств, не зависящих от питания, а высокая надежность определяется средствами защиты доступа к файлам и общей организацией программного кода ОС, при которой сбои аппаратуры чаще всего не разрушают информацию, хранящуюся в файлах.
- Совместное использование информации. Файлы обеспечивают естественный и легкий способ разделения информации между приложениями и пользователями за счет наличия понятного человеку символьного имени и постоянства хранимой информации и расположения файла. Пользователь должен иметь удобные средства работы с файлами, включая каталоги-справочники, объединяющие файлы в группы, средства поиска файлов по признакам, набор команд для создания, модификации и удаления файлов.
- Файл может быть создан одним пользователем, а затем использоваться совсем другим пользователем, при этом создатель файла или администратор могут определить права доступа к нему других пользователей. Эти цели реализуются в ОС файловой системой.

Файловая система включает в себя:

- совокупность всех файлов на диске;
- наборы структур данных, используемых для управления файлами, такие, например, как каталоги файлов, дескрипторы файлов, таблицы распределения свободного и занятого пространства на диске;
- комплекс системных программных средств, реализующих различные операции над файлами, такие, как создание, уничтожение, чтение, запись, именование и поиск файлов. Файловая система позволяет программам обходиться набором достаточно простых операций для выполнения действий над некоторым абстрактным объектом, представляющим файл. Файловая система распределяет

дисковую память, поддерживает именование файлов, отображает имена файлов в соответствующие адреса во внешней памяти, обеспечивает доступ к данным, поддерживает разделение, защиту и восстановление файлов. Таким образом, файловая система играет роль промежуточного слоя, экранирующего все сложности физической организации долговременного хранилища данных и создающего для программ более простую логическую модель этого хранилища, а также предоставляя им набор удобных в использовании команд для манипулирования файлами.

Задачи, решаемые ФС, зависят от способа организации вычислительного процесса в целом. Основные функции ФС нацелены на решение следующих задач:

- именование файлов;
- программный интерфейс для приложений;
- отображения логической модели файловой системы на физическую организацию хранилища данных;
- устойчивость файловой системы к сбоям питания, ошибкам аппаратных и программных средств;
- задача совместного доступа к файлу из нескольких процессов;
- защита файлов одного пользователя от несанкционированного доступа другого пользователя.

Типы файлов. Файловые системы поддерживают несколько функционально различных типов файлов, в число которых, как правило, входят обычные файлы, файлы-каталоги, специальные файлы, именованные конвейеры, отображаемые в память файлы и другие.

Обычные файлы, или просто файлы, содержат информацию произвольного характера, которую заносит в них пользователь или которая образуется в результате работы системных и пользовательских программ. Большинство современных операционных систем (например, UNIX, Windows, OS/2) никак не ограничивает и не контролирует содержимое и структуру обычного файла. Содержание обычного файла определяется приложением, которое с ним работает.

Например, текстовый редактор создает текстовые файлы, состоящие из строк символов, представленных в каком–либо коде. Это могут быть документы, исходные тексты программ и т. п. Текстовые файлы можно прочитать на экране и распечатать на принтере. Двоичные файлы не используют коды символов, они часто имеют сложную внутреннюю структуру, например исполняемый код программы или архивный файл. Все операционные системы должны

уметь распознавать хотя бы один тип файлов – их собственные исполняемые файлы.

Каталоги – это особый тип файлов, которые содержат системную справочную информацию о наборе файлов, сгруппированных пользователями по какому-либо неформальному признаку (например, в одну группу объединяются файлы, содержащие документы одного договора, или файлы, составляющие один программный пакет).

Во многих ОС в каталог могут входить файлы любых типов, в том числе другие каталоги, за счет чего образуется древовидная структура, удобная для поиска. Каталоги устанавливают соответствие между именами файлов и их характеристиками, используемыми ФС для управления файлами. В число таких характеристик входит, в частности, информация (или указатель на другую структуру, содержащую эти данные) о типе файла и расположении его на диске, правах доступа к файлу и датах его создания и модификации. Во всех остальных отношениях каталоги рассматриваются ФС как обычные файлы.

Специальные файлы – это фиктивные файлы, ассоциированные с устройствами ввода-вывода, которые используются для унификации механизма доступа к файлам и внешним устройствам. Специальные файлы позволяют пользователю выполнять операции ввода–вывода посредством обычных команд записи в файл или чтения из файла.

Эти команды обрабатываются сначала программами файловой системы, а затем на некотором этапе выполнения запроса преобразуются операционной системой в команды управления соответствующим устройством.

Современные файловые системы поддерживают и другие типы файлов, такие, как *символьные связи, именованные конвейеры*, *отображаемые в память файлы*.

Иерархическая структура файловой системы. Пользователи обращаются к файлам по символьным именам. Однако способности человеческой памяти ограничивают количество имен объектов, к которым пользователь может обращаться по имени. Иерархическая организация пространства имен позволяет значительно расширить эти границы. Именно поэтому большинство файловых систем имеет иерархическую структуру, в которой уровни создаются за счет того, что каталог более низкого уровня может входить в каталог более высокого уровня (рис. 4.13).

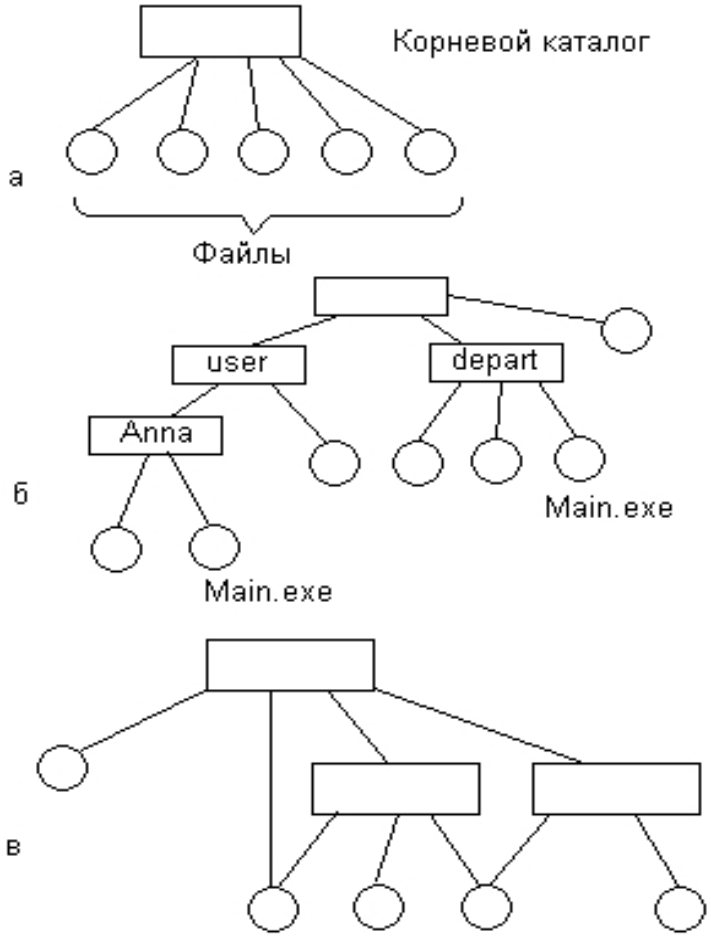


Рис. 4.13. Иерархия файловых систем

Граф, описывающий иерархию каталогов, может быть деревом или сетью. Каталоги образуют дерево, если файлу разрешено входить только в один каталог (рис. 4.13, *б*), и сетью, если файл может входить сразу в несколько каталогов (рис. 4.13, *в*). Например, в MS–DOS и Windows каталоги образуют древовидную структуру, а в UNIX – сетевую. В древовидной структуре каждый файл является листом. Каталог самого верхнего уровня называется *корневым каталогом*, или *корнем* (root). При такой организации пользователь освобожден от запоминания имен всех файлов, ему достаточно примерно представлять, к какой группе может быть отнесен тот или иной файл, чтобы путем последовательного просмотра каталогов найти его. Иерархическая структура удобна для многопользовательской работы: каждый пользователь со своими файлами локализуется в своем каталоге или поддереве каталогов, и вместе с тем все файлы в системе логически связаны. Частным случаем иерархической структуры является одноуровневая организация, когда все файлы входят в один каталог (рис. 4.13, *а*).

Имена файлов. Все типы файлов имеют символьные имена. В иерархически организованных файловых системах обычно используются три типа имен файлов: простые, составные и относительные.

Простое, или *короткое*, символьное имя идентифицирует файл в пределах одного каталога. Простые имена присваивают файлам пользователи и программисты, при этом они должны учитывать ограничения ОС как на номенклатуру символов, так и на длину имени. До сравнительно недавнего времени эти границы были весьма узкими. Так, в популярной файловой системе FAT длина имен ограничивались схемой 8.3 (8 символов – собственно имя, 3 символа – расширение имени), а в файловой системе s5, поддерживаемой многими версиями ОС UNIX, простое символьное имя не могло содержать более 14 символов. Однако пользователю гораздо удобнее работать с длинными именами, поскольку они позволяют дать файлам легко запоминающиеся названия, ясно говорящие о том, что содержится в этом файле. Поэтому современные файловые системы, а также усовершенствованные варианты уже существовавших файловых систем, как правило, поддерживают длинные простые символьные имена файлов. Например, в файловых системах NTFS и FAT32, входящих в состав операционной системы Windows, имя файла может содержать до 255 символов.

В иерархических файловых системах разным файлам разрешено иметь одинаковые простые символьные имена при условии, что они принадлежат разным каталогам. То есть здесь работает схема «много файлов – одно простое имя». Для однозначной идентификации файла в таких системах используется так называемое полное имя.

Полное имя представляет собой цепочку простых символьных имен всех каталогов, через которые проходит путь от корня до данного файла. Таким образом, полное имя является составным, в котором простые имена отделены друг от друга принятым в ОС разделителем. Часто в качестве разделителя используется прямой или обратный слеш, при этом принято не указывать имя корневого каталога. На рис. 4.13, *б* два файла имеют простое имя main.exe, однако их составные имена /depart/main.exe и /user/anna/main.exe различаются. В древовидной файловой системе между файлом и его полным именем имеется взаимно однозначное соответствие «один файл – одно полное имя». В файловых системах, имеющих сетевую структуру, файл может входить в несколько каталогов, а значит, иметь несколько полных имен; здесь справедливо соответствие «один файл – много полных имен». В обоих случаях файл однозначно идентифицируется полным именем.

Файл может быть идентифицирован также относительным именем. *Относительное имя файла* определяется через понятие «текущий каталог». Для каждого пользователя в каждый момент времени один из каталогов файловой системы является текущим, причем этот каталог выбирается самим пользователем по команде ОС. Файловая система фиксирует имя текущего каталога, чтобы затем использовать его как дополнение к относительным именам для образования полного имени файла. При использовании относительных имен пользователь идентифицирует файл цепочкой имен каталогов, через которые проходит маршрут от текущего каталога до данного файла. Например, если текущим каталогом является каталог /user, то относительное имя файла /user/anna/main.exe выглядит следующим образом: anna/ main.exe.

Уникальное имя представляет собой числовой идентификатор и предназначено только для операционной системы. Примером такого уникального имени файла является номер индексного дескриптора в системе UNIX.

Монтирование. В общем случае вычислительная система может иметь несколько дисковых устройств. Даже типичный персональный компьютер обычно имеет один накопитель на жестком диске и накопитель для компакт–дисков. Мощные же компьютеры, как правило, оснащены большим количеством дисковых накопителей, на которые устанавливаются пакеты дисков. Более того, даже одно физическое устройство с помощью средств операционной системы может быть представлено в виде нескольких логических устройств, в частности путем разбиения дискового пространства на разделы. Возникает вопрос, каким образом организовать хранение файлов в системе, имеющей несколько устройств внешней памяти? Первое решение состоит в том, что на каждом из устройств размещается автономная файловая система, то есть файлы, находящиеся на этом устройстве, описываются деревом каталогов, никак не связанным с деревьями каталогов на других устройствах. В таком случае для однозначной идентификации файла пользователь наряду с составным символьным именем файла должен указывать идентификатор логического устройства.

Примером такого автономного существования файловых систем является операционная система MS-DOS, в которой полное имя файла включает буквенный идентификатор логического диска. Так, при обращении к файлу, расположенному на диске A, пользователь должен указать имя этого диска, например A:\privat\letter\uni\let1.doc.

Другим вариантом является такая организация хранения файлов, при которой пользователю предоставляется возможность объединять файловые системы, находящиеся на разных устройствах, в единую

файловую систему, описываемую единым деревом каталогов. Такая операция называется монтированием. Рассмотрим, как осуществляется эта операция на примере ОС UNIX. Среди всех имеющихся в системе логических дисковых устройств операционная система выделяет одно устройство, называемое системным. Пусть имеются две файловые системы, расположенные на разных логических дисках (рис. 4.14), причем один из дисков является системным.

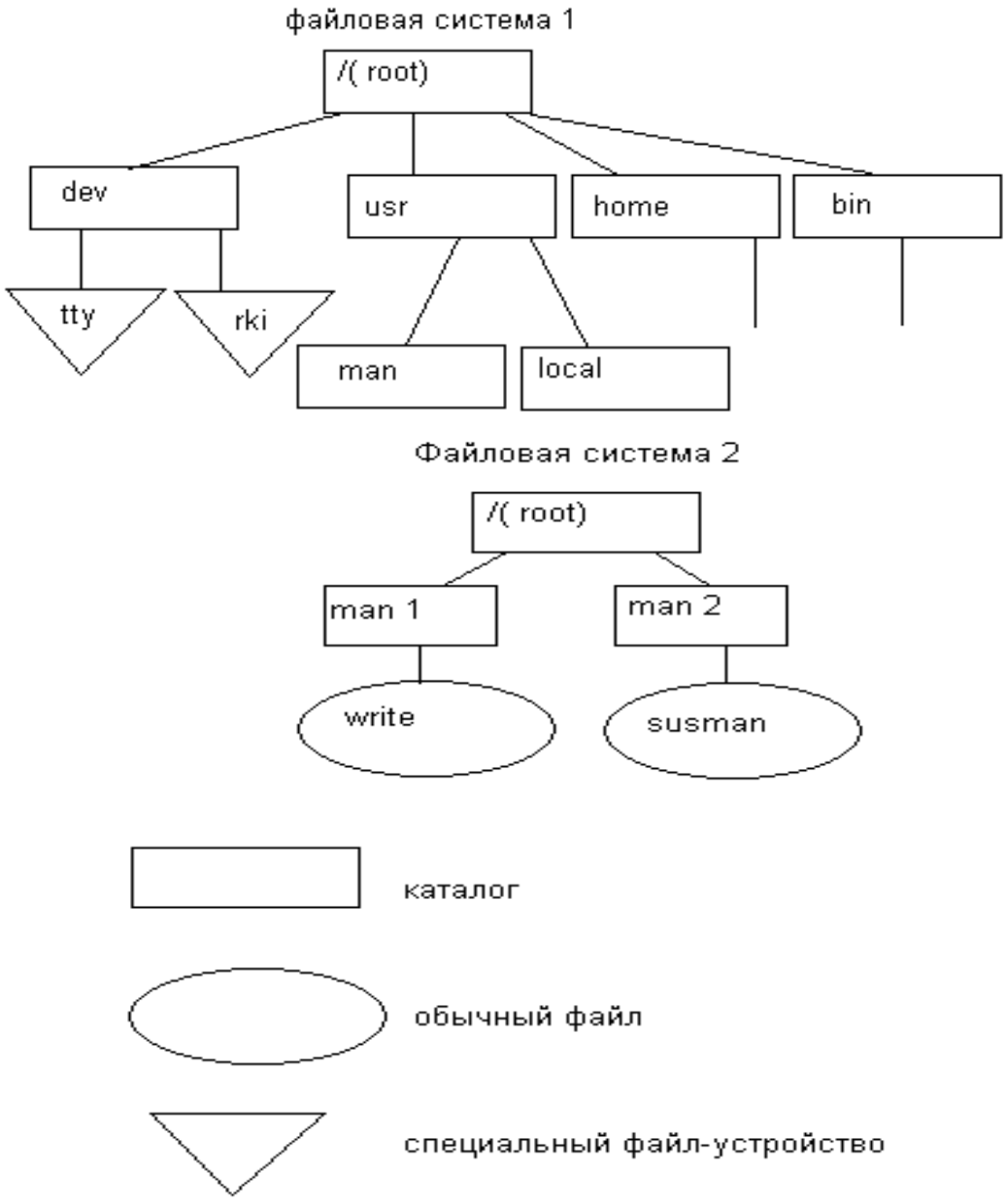


Рис. 4.14. Две файловые системы до монтирования

Файловая система, расположенная на системном диске, назначается *корневой.* Для связи иерархий файлов в корневой файловой системе выбирается некоторый существующий каталог, в данном примере –

каталог man. После выполнения монтирования выбранный каталог man становится корневым каталогом второй файловой системы. Через этот каталог монтируемая файловая система подсоединяется как поддерево к общему дереву (рис. 4.15).

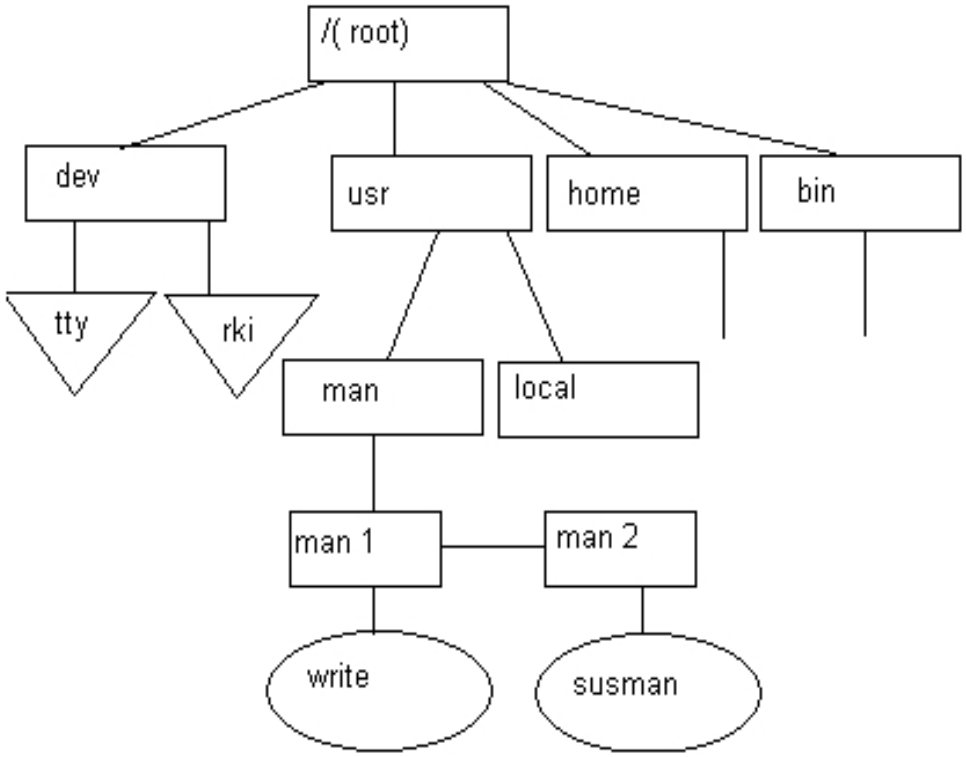


Рис. 4.15. Общая файловая система после монтирования

После монтирования общей файловой системы для пользователя нет логической разницы между корневой и смонтированной файловыми системами, в частности, именование файлов производится так же, как если бы она с самого начала была единой.

Атрибуты файлов. Понятие «файл» включает не только хранимые им данные и имя, но и атрибуты. *Атрибуты* – это информация, описывающая свойства файла.

Примеры возможных атрибутов файла:

- тип файла (обычный файл, каталог, специальный файл и т. п.);
- владелец файла;
- создатель файла;
- пароль для доступа к файлу;
- информация о разрешенных операциях доступа к файлу;
- время создания, последнего доступа и последнего изменения;
- текущий размер файла;
- максимальный размер файла;

- признак «только для чтения»;
- признак «скрытый файл»;
- признак «системный файл»;
- признак «архивный файл»;
- признак «двоичный/символьный»;
- признак «временный» (удалить после завершения процесса);
- признак блокировки;
- длина записи в файле;
- указатель на ключевое поле в записи;
- длина ключа.

Набор атрибутов файла определяется спецификой файловой системы: в файловых системах разного типа для характеристики файлов могут использоваться разные наборы атрибутов. Например, в однопользовательской ОС в наборе атрибутов будут отсутствовать характеристики, имеющие отношение к пользователям и защите, такие, как владелец файла, создатель файла, пароль для доступа к файлу, информация о разрешенном доступе к файлу.

Пользователь может получать доступ к атрибутам, используя средства, предоставленные для этих целей файловой системой. Обычно разрешается читать значения любых атрибутов, а изменять – только некоторые. Например, пользователь может изменить права доступа к файлу (при условии, что он обладает необходимыми для этого полномочиями), но изменять дату создания или текущий размер файла ему не разрешается.

Логическая организация файла. В общем случае данные, содержащиеся в файле, имеют некую логическую структуру. Эта структура является базой при разработке программы, предназначенной для обработки этих данных. Например, чтобы текст мог быть правильно выведен на экран, программа должна иметь возможность выделить отдельные слова, строки, абзацы и т. д.

Признаками, отделяющими один структурный элемент от другого, могут служить определенные кодовые последовательности или просто известные программе значения смещений этих структурных элементов относительно начала файла. Поддержание структуры данных может быть либо целиком возложено на приложение, либо в той или иной степени эту работу может взять на себя файловая система.

В первом случае, когда все действия, связанные со структуризацией и интерпретацией содержимого файла целиком относятся к ведению приложения, файл представляется ФС неструктурированной последовательностью данных. Приложение формулирует запросы к

файловой системе на ввод-вывод, используя общие для всех приложений системные средства, например, указывая смещение от начала файла и количество байт, которые необходимо считать или записать.

Поступивший к приложению поток байт интерпретируется в соответствии с заложенной в программе логикой. Например, компилятор генерирует, а редактор связей воспринимает вполне определенный формат объектного модуля программы. При этом формат файла, в котором хранится объектный модуль, известен только этим программам.

Подчеркнем, что интерпретация данных никак не связана с действительным способом их хранения в файловой системе.

Модель файла, в соответствии с которой содержимое файла представляется неструктурированной последовательностью (потоком) байт, стала популярной вместе с ОС UNIX, а теперь она широко используется в большинстве современных ОС. Неструктурированная модель файла позволяет легко организовать разделение файла между несколькими приложениями: разные приложения могут по-своему структурировать и интерпретировать данные, содержащиеся в файле.

Другая модель файла, которая применялась в ОС OS/360, DEC RSX и VMS, а в настоящее время используется достаточно редко, – это структурированный файл. В этом случае поддержание структуры файла поручается файловой системе. Файловая система видит файл как упорядоченную последовательность логических записей. Приложение может обращаться к ФС с запросами на ввод–вывод на уровне записей, например «считать запись 25 из файла FILE.DOC». ФС должна обладать информацией о структуре файла, достаточной для того, чтобы выделить любую запись. ФС предоставляет приложению доступ к записи, а вся дальнейшая обработка данных, содержащихся в этой записи, выполняется приложением. Развитием этого подхода стали системы управления базами данных (СУБД), которые поддерживают не только сложную структуру данных, но и взаимосвязи между ними.

Логическая запись является наименьшим элементом данных, которым может оперировать программист при организации обмена с внешним устройством. Даже если физический обмен с устройством осуществляется большими единицами, операционная система должна обеспечивать программисту доступ к отдельной логической записи.

Файловая система может использовать два способа доступа к логическим записям: читать или записывать логические записи последовательно (последовательный доступ) или позиционировать файл на запись с указанным номером (прямой доступ).

Очевидно, что ОС не может поддерживать все возможные способы структурирования данных в файле, поэтому в тех ОС, в которых вообще существует поддержка логической структуризации файлов, она существует для небольшого числа широко распространенных схем логической организации файла.

К числу таких способов структуризации относится представление данных в виде записей, длина которых фиксирована в пределах файла (рис. 4.16, *а*). В таком случае доступ к n-й записи осуществляется либо путем последовательного чтения (n–1) предшествующих записей, либо прямо по адресу, вычисленному по ее порядковому номеру. Например, если L–длина записи, то начальный адрес n-й записи равен Lxn. Заметим, что при такой логической организации размер записи фиксирован в пределах файла, а записи в различных файлах, принадлежащих одной и той же файловой системе, могут иметь различный размер.

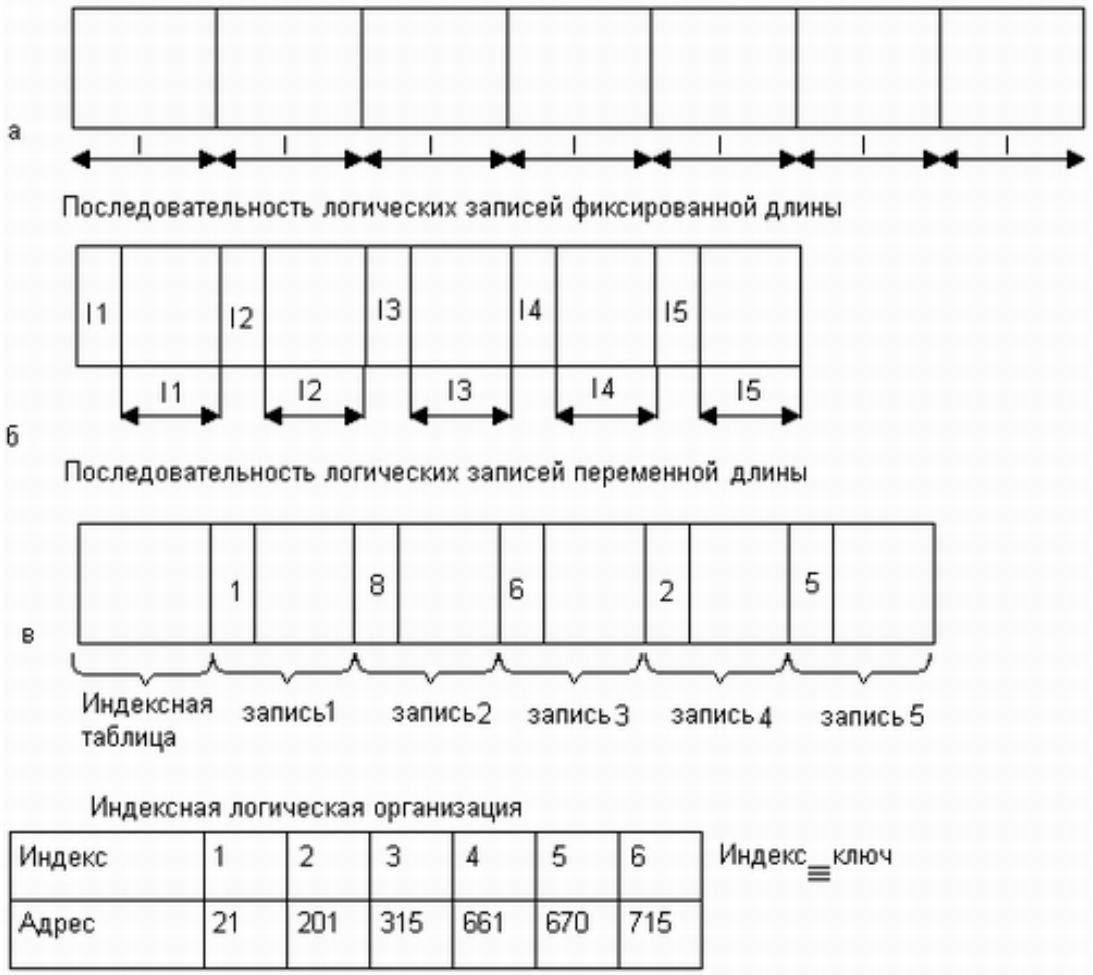


Рис. 4.16. Способы логической организации файлов

Другой способ структуризации состоит в представлении данных в виде последовательности записей, размер которых изменяется в пределах одного файла. Если расположить значения длин записей так, как это показано на рис. 4.16, *б*, то для поиска нужной записи система должна последовательно считать все предшествующие записи. Вычислить адрес нужной записи по ее номеру при такой логической организации файла невозможно, а следовательно, не может быть применен более эффективный метод прямого доступа.

Файлы, доступ к записям которых осуществляется последовательно, по номерам позиций, называются *неиндексированными*, или *последовательными.*

Другим типом файлов являются *индексированные файлы*, они допускают более быстрый прямой доступ к отдельной логической записи. В индексированном файле (рис. 4.16, *в*) записи имеют одно или более ключевых (индексных) полей и могут адресоваться путем указания значений этих полей. Для быстрого поиска данных в индексированном файле предусматривается специальная индексная таблица, в которой значениям ключевых полей ставится в соответствие адрес внешней памяти. Этот адрес может указывать либо непосредственно на искомую запись, либо на некоторую область внешней памяти, занимаемую несколькими записями, в число которых входит искомая запись. В последнем случае говорят, что файл имеет индексно-последовательную организацию, так как поиск включает два этапа: прямой доступ по индексу к указанной области диска, а затем последовательный просмотр записей в указанной области. Ведение индексных таблиц берет на себя файловая система.

Все вышесказанное в большей степени относится к обычным файлам, которые могут быть как структурированными, так и неструктурированными. Что же касается других типов файлов, то они обладают определенной структурой, известной файловой системе. Например, файловая система должна понимать структуру данных, хранящихся в файле-каталоге или файле типа «символьная связь».

Физическая организация файловой системы

Представление пользователя о файловой системе как об иерархически организованном множестве информационных объектов имеет мало общего с порядком хранения файлов на диске. Файл, имеющий образ цельного, непрерывающегося набора байт, на самом деле очень часто разбросан «кусочками» по всему диску, причем это разбиение никак не связано с логической структурой файла, например, его отдельная логическая запись может быть расположена в несмежных

секторах диска. Логически объединенные файлы из одного каталога совсем не обязаны соседствовать на диске. Принципы размещения файлов, каталогов и системной информации на реальном устройстве описываются физической организацией файловой системы. Очевидно, что разные файловые системы имеют разную физическую организацию.

Диски, разделы, секторы, кластеры. Основным типом устройства, которое используется в современных вычислительных системах для хранения файлов, являются дисковые накопители. Эти устройства предназначены для считывания и записи данных на жесткие диски. Жесткий диск состоит из одной или нескольких стеклянных или металлических пластин, каждая из которых покрыта с одной или двух сторон магнитным материалом. Таким образом, диск в общем случае состоит из пакета пластин (рис. 4.17).

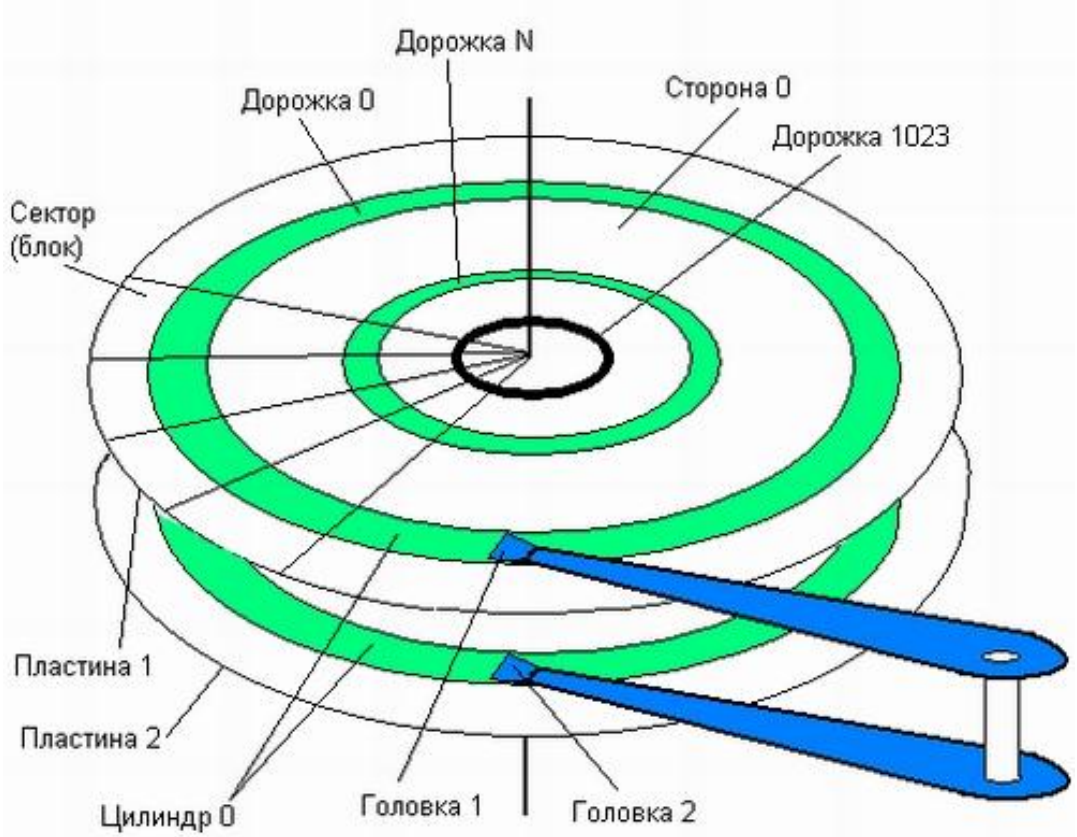


Рис. 4.17. Схема устройства жесткого диска

На каждой стороне каждой пластины размечены тонкие концентрические кольца – *дорожки* (traks), на которых хранятся данные. Количество дорожек зависит от типа диска. Нумерация дорожек начинается с 0 от внешнего края к центру диска. Когда диск вращается, элемент, называемый головкой, считывает двоичные данные

с магнитной дорожки или записывает их на магнитную дорожку. Головка может позиционироваться над заданной дорожкой. Головки перемещаются над поверхностью диска дискретными шагами, каждый шаг соответствует сдвигу на одну дорожку. Запись на диск осуществляется благодаря способности головки изменять магнитные свойства дорожки. В некоторых дисках вдоль каждой поверхности перемещается одна головка, а в других − имеется по головке на каждую дорожку.

В первом случае для поиска информации головка должна перемещаться по радиусу диска. Обычно все головки закреплены на едином перемещающем механизме и двигаются синхронно. Поэтому, когда головка фиксируется на заданной дорожке одной поверхности, все остальные головки останавливаются над дорожками с такими же номерами.

В тех же случаях, когда на каждой дорожке имеется отдельная головка, никакого перемещения головок с одной дорожки на другую не требуется, за счет этого экономится время, затрачиваемое на поиск данных.

Совокупность дорожек одного радиуса на всех поверхностях всех пластин пакета называется *цилиндром* (cylinder). Каждая дорожка разбивается на фрагменты, называемые *секторами* (sectors), или блоками (blocks), так что все дорожки имеют равное число секторов, в которые можно максимально записать одно и то же число байт (Иногда внешняя дорожка имеет несколько дополнительных секторов, используемых для замены поврежденных секторов в режиме горячего резервирования). Сектор имеет фиксированный для конкретной системы размер, выражающийся степенью двойки. Чаще всего размер сектора составляет 512 байт. Учитывая, что дорожки разного радиуса имеют одинаковое число секторов, плотность записи становится тем выше, чем ближе дорожка к центру.

Сектор − наименьшая адресуемая единица обмена данными дискового устройства с оперативной памятью. Для того чтобы контроллер мог найти на диске нужный сектор, необходимо задать ему все составляющие адреса сектора: номер цилиндра, номер поверхности и номер сектора. Так как прикладной программе в общем случае нужен не сектор, а некоторое количество байт, не обязательно кратное размеру сектора, то типичный запрос включает чтение нескольких секторов, содержащих требуемую информацию, и одного или двух секторов, содержащих наряду с требуемыми избыточные данные (рис. 4.18).

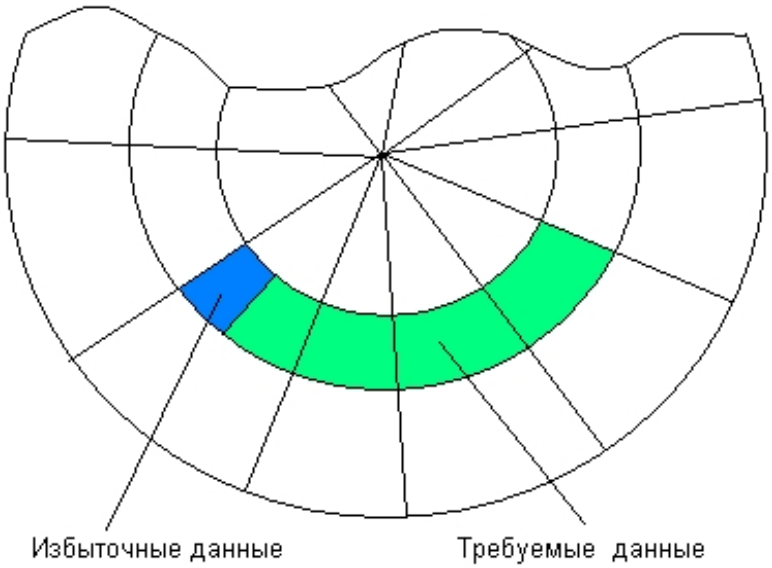


Рис. 4.18. Считывание избыточных данных при обмене с диском

Операционная система при работе с диском использует, как правило, собственную единицу дискового пространства, называемую *кластером* (cluster). Иногда кластер называют блоком (например, в ОС Unix), что может привести к терминологической путанице. Вообще, терминология, используемая при описании форматов дисков и файловых систем, зависит от аппаратной платформы (RISC, Wintel и т. п.) и операционной системы. Это нужно учитывать и трактовать термины в зависимости от контекста.

При создании файла место на диске ему выделяется кластерами. Например, если файл имеет размер 2560 байт, а размер кластера в файловой системе определен в 1024 байта, то файлу будет выделено на диске 3 кластера.

Дорожки и секторы создаются в результате выполнения процедуры физического, или низкоуровневого форматирования диска, предшествующей использованию диска. Для определения границ блоков на диск записывается идентификационная информация. Низкоуровневый формат диска не зависит от типа операционной системы, которая этот диск будет использовать.

Разметку диска под конкретный тип файловой системы выполняют процедуры высокоуровневого, или логического, форматирования. При высокоуровневом форматировании определяется размер кластера и на диск записывается информация, необходимая для работы файловой системы, в том числе информация о доступном и неиспользуемом пространстве, о границах областей, отведенных под файлы и каталоги, информация о поврежденных областях. Кроме того, на диск

записывается *загрузчик операционной системы* – небольшая программа, которая начинает процесс инициализации операционной системы после включения питания или рестарта компьютера.

Прежде чем форматировать диск под определенную файловую систему, он может быть разбит на разделы. *Раздел* – это непрерывная часть физического диска, которую операционная система представляет пользователю как логическое устройство (используются также названия логический диск и логический раздел). Во многих операционных системах используется термин «том» (volume). В разных ОС толкование этого термина имеет свои нюансы, но чаще всего он обозначает логическое устройство, отформатированное под конкретную файловую систему.

Логическое устройство функционирует так, как если бы это был отдельный физический диск. Именно с логическими устройствами работает пользователь, обращаясь к ним по символьным именам, используя, например, обозначения A, B, C, SYS и т. п. Операционные системы разного типа используют единое для всех них представление о разделах, но создают на его основе логические устройства, специфические для каждого типа ОС. Так же как файловая система, с которой работает одна ОС, в общем случае не может интерпретироваться ОС другого типа, логические устройства не могут быть использованы операционными системами разного типа. На каждом логическом устройстве может создаваться только одна файловая система.

В частном случае, когда все дисковое пространство охватывается одним разделом, логическое устройство представляет физическое устройство в целом. Если диск разбит на несколько разделов, то для каждого из этих разделов может быть создано отдельное логическое устройство. Логическое устройство может быть создано и на базе нескольких разделов, причем эти разделы не обязательно должны принадлежать одному физическому устройству. Объединение нескольких разделов в единое логическое устройство может выполняться разными способами и преследовать разные цели, основные из которых: увеличение общего объема логического раздела, повышение производительности и отказоустойчивости. Примерами организации совместной работы нескольких дисковых разделов являются так называемые RAID-массивы.

На разных логических устройствах одного и того же физического диска могут располагаться файловые системы разного типа. На рис. 4.19 показан пример диска, разбитого на три раздела, в которых установлены

две файловых системы NTFS (разделы C и E) и одна файловая система FAT (раздел D).

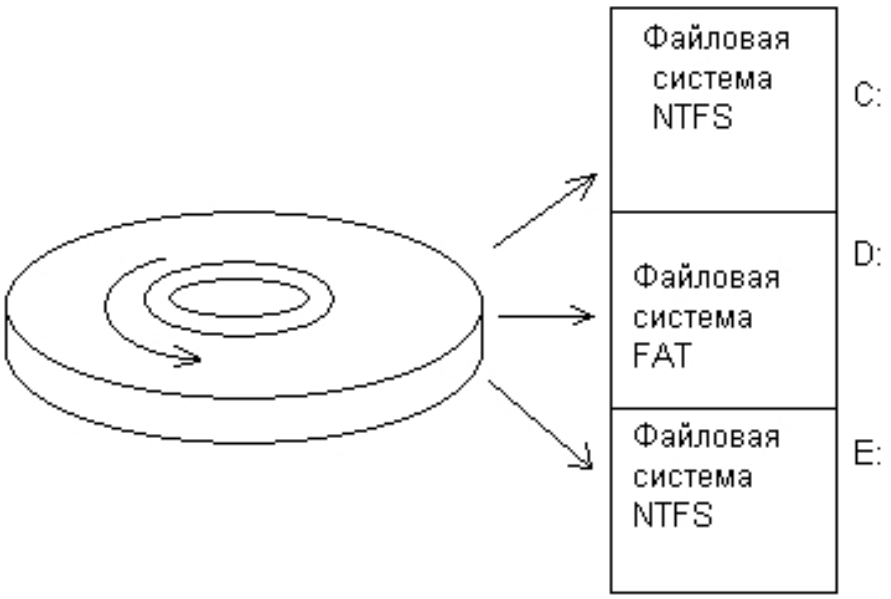


Рис. 4.19. Разбивки диска на разделы

Все разделы одного диска имеют одинаковый размер блока, определенный для данного диска в результате низкоуровневого форматирования. Однако в результате высокоуровневого форматирования в разных разделах одного и того же диска, представленных разными логическими устройствами, могут быть установлены файловые системы, в которых определены кластеры отличающихся размеров. Операционная система может поддерживать разные статусы разделов, особым образом отмечая разделы, которые могут быть использованы для загрузки модулей операционной системы, и разделы, в которых можно устанавливать только приложения и хранить файлы данных. Один из разделов диска помечается как загружаемый (или активный). Именно из этого раздела считывается загрузчик операционной системы.

Физическая организация и адресация файла. Важным компонентом физической организации файловой системы является физическая организация файла, то есть способ размещения файла на диске (рис. 4.20).

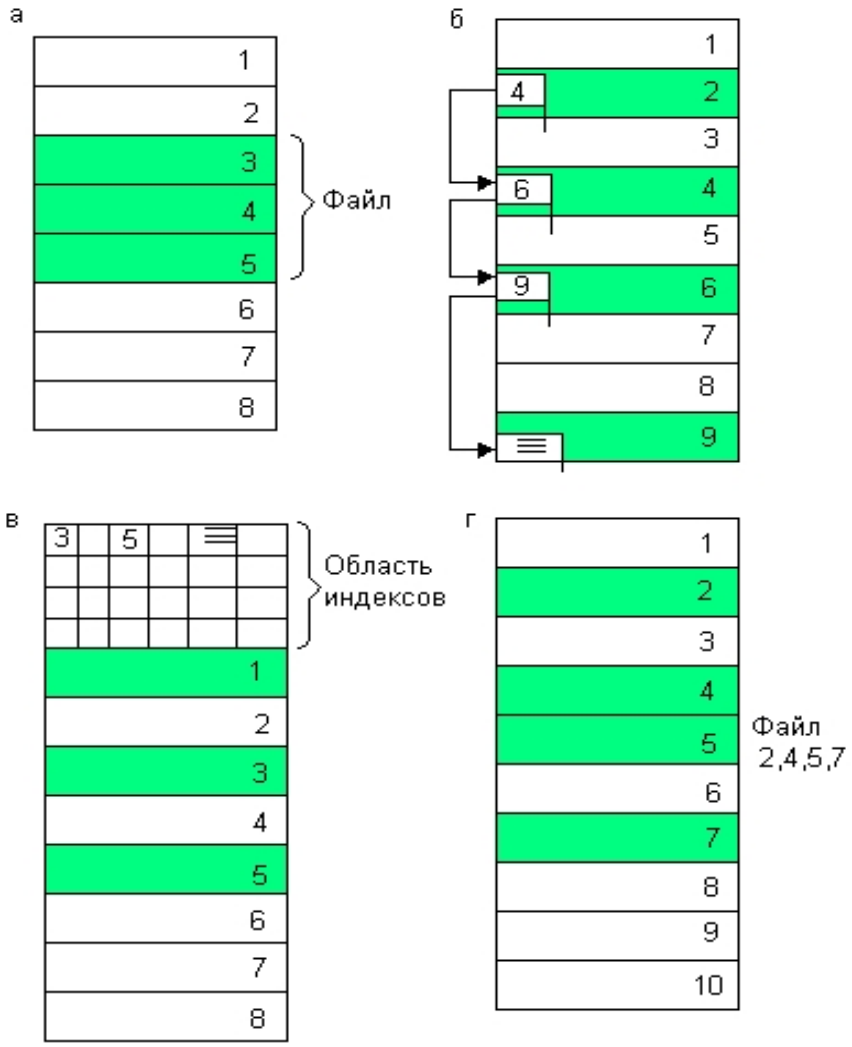


Рис. 4.20. Физическая организация файла:
непрерывное размещение (а); связанный список кластеров (б);
связанный список индексов (в); перечень номеров кластеров (г).

Основными критериями эффективности физической организации файлов являются:

- скорость доступа к данным;
- объем адресной информации файла;
- степень фрагментированности дискового пространства;
- максимально возможный размер файла.

Непрерывное размещение – простейший вариант физической организации (рис. 4.20, *а*), при котором файлу предоставляется последовательность кластеров диска, образующих непрерывный участок дисковой памяти. Основным достоинством этого метода является высокая скорость доступа, так как затраты на поиск и считывание кластеров файла минимальны. Также минимален объем

адресной информации – достаточно хранить только номер первого кластера и объем файла. Данная физическая организация максимально возможный размер файла не ограничивает.

Однако этот вариант имеет существенные недостатки, которые затрудняют его применимость на практике, несмотря на всю его логическую простоту. При более пристальном рассмотрении оказывается, что реализовать эту схему не так уж просто. Действительно, какого размера должна быть непрерывная область, выделяемая файлу, если файл при каждой модификации может увеличить свой размер? Еще более серьезной проблемой является фрагментация. Спустя некоторое время после создания файловой системы в результате выполнения многочисленных операций создания и удаления файлов пространство диска неминуемо превращается в «лоскутное одеяло», включающее большое число свободных областей небольшого размера. Как всегда бывает при фрагментации, суммарный объем свободной памяти может быть очень большим, а выбрать место для размещения файла целиком невозможно. Поэтому на практике используются методы, в которых файл размещается в нескольких, в общем случае несмежных областях диска.

Следующий способ физической организации – размещение файла в виде связанного списка кластеров дисковой памяти (рис. 4.20, *б*). При таком способе в начале каждого кластера содержится указатель на следующий кластер. В этом случае адресная информация минимальна: расположение файла может быть задано одним числом – номером первого кластера. В отличие от предыдущего способа каждый кластер может быть присоединен к цепочке кластеров какого-либо файла. Файл может изменять свой размер во время своего существования, наращивая число кластеров.

Недостатком является сложность реализации доступа к произвольно заданному месту файла. Чтобы прочитать пятый по порядку кластер файла, необходимо последовательно прочитать четыре первых кластера, прослеживая цепочку номеров кластеров. Кроме того, при этом способе количество данных файла, содержащихся в одном кластере, не равно степени двойки (одно слово израсходовано на номер следующего кластера), а многие программы читают данные кластерами, размер которых равен степени двойки.

Популярным способом, применяемым, например, в файловой системе FAT, является использование связанного списка индексов (рис. 4.20, *в*). Этот способ является некоторой модификацией предыдущего. Файлу также выделяется память в виде связанного списка кластеров.

Номер первого кластера запоминается в записи каталога, где хранятся характеристики этого файла. Остальная адресная информация отделена от кластеров файла.

С каждым кластером диска связывается некоторый элемент – индекс. Индексы располагаются в отдельной области диска – это таблица FAT (File Allocation Table), занимающая один кластер. Когда память свободна, все индексы имеют нулевое значение. Если некоторый кластер N назначен некоторому файлу, то индекс этого кластера становится равным либо номеру M следующего кластера данного файла, либо принимает специальное значение, являющееся признаком того, что этот кластер является для файла последним. Индекс же предыдущего кластера файла принимает значение N, указывая на вновь назначенный кластер.

При такой физической организации сохраняются все достоинства предыдущего способа: минимальность адресной информации, отсутствие фрагментации, отсутствие проблем при изменении размера. Кроме того, данный способ обладает дополнительными преимуществами. Во-первых, для доступа к произвольному кластеру файла не требуется последовательно считывать его кластеры, достаточно прочитать только секторы диска, содержащие таблицу индексов, отсчитать нужное количество кластеров файла по цепочке и определить номер нужного кластера. Во-вторых, данные файла заполняют кластер целиком, а значит, имеют объем, равный степени двойки.

Необходимо отметить, что при отсутствии фрагментации на уровне кластеров на диске все равно имеется определенное количество областей памяти небольшого размера, которые невозможно использовать, то есть фрагментация все же существует. Эти фрагменты представляют собой неиспользуемые части последних кластеров, назначенных файлам, поскольку объем файла в общем случае не кратен размеру кластера. На каждом файле в среднем теряется половина кластера. Это потери особенно велики, когда на диске имеется большое количество маленьких файлов, а кластер имеет большой размер. Размеры кластеров зависят от размера раздела и типа файловой системы. Примерный диапазон, в котором может меняться размер кластера, составляет от 512 байт до десятков килобайт.

Еще один способ задания физического расположения файла заключается в простом перечислении номеров кластеров, занимаемых этим файлом (рис. 4.20, *г*). Этот перечень и служит адресом файла. Недостаток данного способа очевиден: длина адреса зависит от размера файла и для большого файла может составить значительную величину.

Достоинством же является высокая скорость доступа к произвольному кластеру файла, так как здесь применяется прямая адресация, которая исключает просмотр цепочки указателей при поиске адреса произвольного кластера файла. Фрагментация на уровне кластеров в этом способе также отсутствует.

Последний подход с некоторыми модификациями используется в традиционных файловых системах ОС UNIX s5 и ufs. Современные версии UNIX поддерживают и другие типы файловых систем, в том числе и пришедшие из других ОС, как, например, FAT. Для сокращения объема адресной информации прямой способ адресации сочетается с косвенным.

Метод перечисления адресов кластеров файла задействован и в файловой системе NTFS, используемой в ОС Windows NT/2000. Здесь он дополнен достаточно естественным приемом, сокращающим объем адресной информации: адресуются не кластеры файла, а непрерывные области, состоящие из смежных кластеров диска. Каждая такая область, называемая отрезком (run), или экстентом (extent), описывается с помощью двух чисел: начального номера кластера и количества кластеров в отрезке. Так как для сокращения времени операции обмена ОС старается разместить файл в последовательных кластерах диска, то в большинстве случаев количество последовательных областей файла будет меньше количества кластеров файла, и объем служебной адресной информации в NTFS сокращается по сравнению со схемой адресации файловых систем ufs/s5.

Для того чтобы корректно принимать решение о выделении файлу набора кластеров, файловая система должна отслеживать информацию о состоянии всех кластеров диска: свободен/занят. Эта информация может храниться как отдельно от адресной информации файлов, так и вместе с ней.

Примеры файловых систем

Физическая организация FAT

Логический раздел, отформатированный под файловую систему FAT, состоит из следующих областей (рис. 4.21).

1. Загрузочный сектор содержит программу начальной загрузки операционной системы. Вид этой программы зависит от типа операционной системы, которая будет загружаться из этого раздела.
2. Основная копия FAT содержит информацию о размещении файлов и каталогов на диске.
3. Резервная копия FAT.

4. Корневой каталог занимает фиксированную область размером в 32 сектора (16 Кбайт), что позволяет хранить 512 записей о файлах и каталогах, так как каждая запись каталога состоит из 32 байт.

5. Область данных предназначена для размещения всех файлов и всех каталогов, кроме корневого каталога.

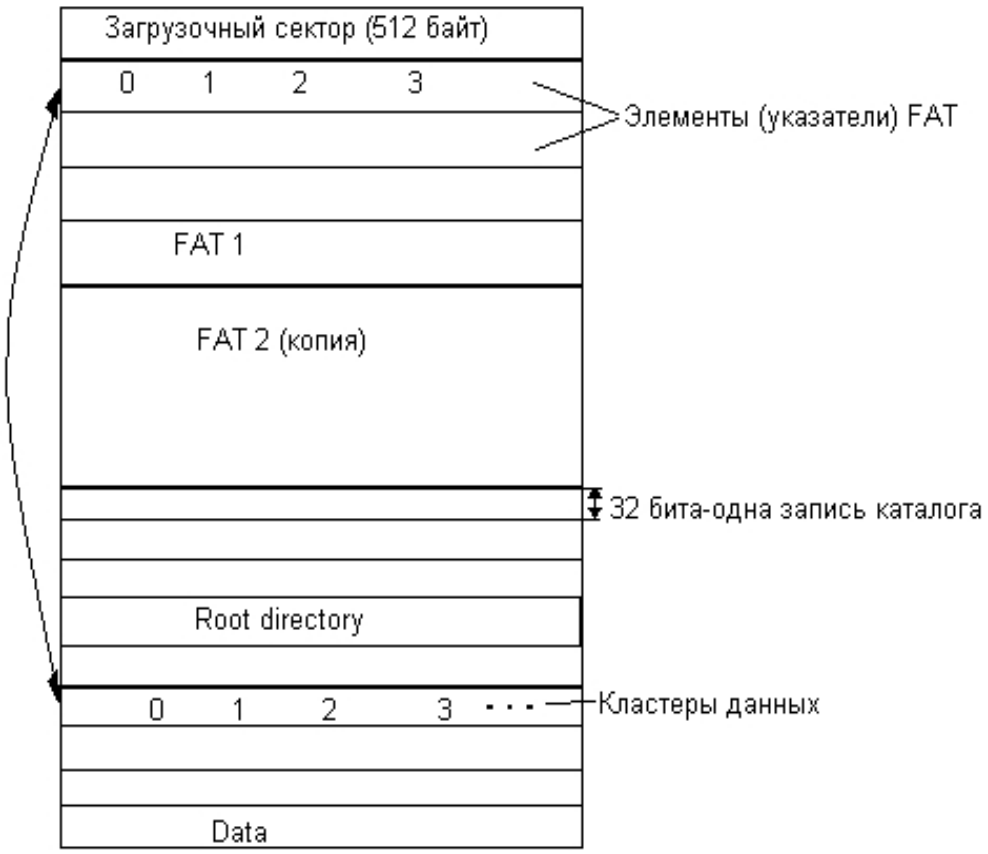


Рис. 4.21. Физическая структура файловой системы FAT

Файловая система FAT поддерживает всего два типа файлов: обычный файл и каталог. Файловая система распределяет память только из области данных, причем использует в качестве минимальной единицы дискового пространства кластер. Таблица FAT (как основная копия, так и резервная) состоит из массива индексных указателей, количество которых равно количеству кластеров области данных. Между кластерами и индексными указателями имеется взаимно однозначное соответствие − нулевой указатель соответствует нулевому кластеру и т. д.

Индексный указатель может принимать следующие значения, характеризующие состояние связанного с ним кластера:

- кластер свободен (не используется);

- кластер используется файлом и не является последним кластером файла; в этом случае индексный указатель содержит номер следующего кластера файла;
- последний кластер файла; О дефектный кластер; а резервный кластер.

Таблица FAT является общей для всех файлов раздела. В исходном состоянии (после форматирования) все кластеры раздела свободны и все индексные указатели (кроме тех, которые соответствуют резервным и дефектным блокам) принимают значение «кластер свободен». При размещении файла ОС просматривает FAT, начиная с начала, и ищет первый свободный индексный указатель. После его обнаружения в поле записи каталога «номер первого кластера» фиксируется номер этого указателя. В кластер с этим номером записываются данные файла, он становится первым кластером файла. Если файл умещается в одном кластере, то в указатель, соответствующий данному кластеру, заносится специальное значение «последний кластер файла». Если же размер файла больше одного кластера, то ОС продолжает просмотр FAT и ищет следующий указатель на свободный кластер. После его обнаружения в предыдущий указатель заносится номер этого кластера, который теперь становится следующим кластером файла. Процесс повторяется до тех пор, пока не будут размещены все данные файла. Таким образом создается связный список всех кластеров файла.

В начальный период после форматирования файлы будут размещаться в последовательных кластерах области данных, однако после определенного количества удалений файлов кластеры одного файла окажутся в произвольных местах области данных, чередуясь с кластерами других файлов (рис. 4.22). Размер таблицы FAT и разрядность используемых в ней индексных указателей определяется количеством кластеров в области данных. Для уменьшения потерь из-за фрагментации желательно кластеры делать небольшими, а для сокращения объема адресной информации и повышения скорости обмена наоборот – чем больше, тем лучше. При форматировании диска под файловую систему FAT обычно выбирается компромиссное решение, и размеры кластеров выбираются из диапазона от 1 до 128 секторов, или от 512 байт до 64 Кбайт.

Очевидно, что разрядность индексного указателя должна быть такой, чтобы в нем можно было задать максимальный номер кластера для диска определенного объема.

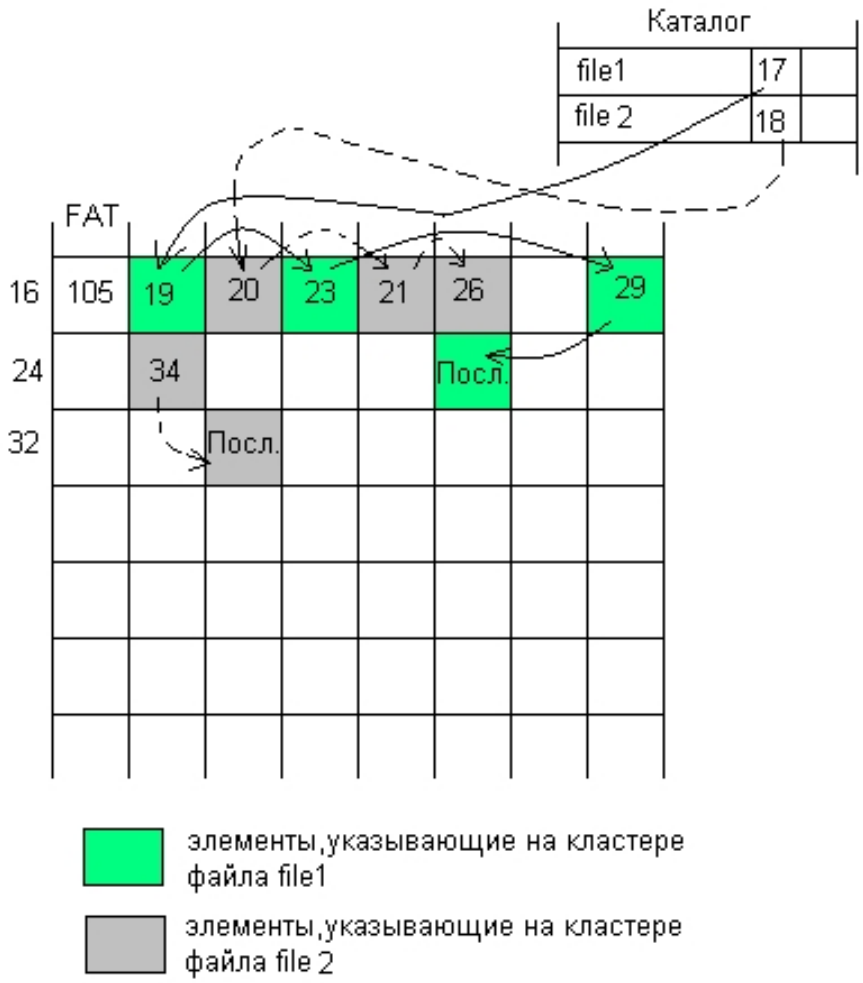


Рис. 4.22. Списки указателей файлов в FAT

Существует несколько разновидностей FAT, отличающихся разрядностью индексных указателей, которая и используется в качестве условного обозначения, – FAT16 и FAT32. В файловой системе FAT16 используются 16-разрядные указатели, что позволяет поддерживать до 65536 кластеров в области данных диска, в FAT32 – 32-разрядные для более чем 4 миллиардов кластеров. Реально это число немного меньше, так как несколько значений индексного указателя расходуется для идентификации специальных ситуаций, таких, как «Последний кластер», «Неиспользуемый кластер», «Дефектный кластер» и «Резервный кластер».

FAT 16 целесообразнее для дисков с объемом не более 512 Мбайт, а для больших дисков лучше подходит FAT32, которая способна использовать кластеры 4 Кбайт при работе с дисками объемом до 8 Гбайт и только для дисков большего объема начинает использовать 8, 16 и 32 Кбайт. Максимальный размер раздела FAT 16 ограничен 4 Гбайт, такой объем дает 65 536 кластеров по 64 Кбайт каждый, а

максимальный размер раздела FAT32 практически не ограничен – 2^{32} кластеров по 32 Кбайт. Таблица FAT при фиксированной разрядности индексных указателей имеет переменный размер, зависящий от объема области данных диска. При удалении файла из файловой системы FAT в первый байт соответствующей записи каталога заносится специальный признак, свидетельствующий о том, что эта запись свободна, а во все индексные указатели файла заносится признак «кластер свободен». Остальные данные в записи каталога, в том числе номер первого кластера файла, остаются нетронутыми, что оставляет шансы для восстановления ошибочно удаленного файла. Существует большое количество утилит для восстановления удаленных файлов FAT, выводящих пользователю список имен удаленных файлов с отсутствующим первым символом имени, затертым после освобождения записи. Очевидно, что надежно можно восстановить только файлы, которые были расположены в последовательных кластерах диска, так как при отсутствии связного списка выявить принадлежность произвольно расположенного кластера удаленному файлу невозможно (без анализа содержимого кластеров, выполняемого пользователем «вручную»). Используемый в FAT метод хранения адресной информации о файлах не отличается большой надежностью – при разрыве списка индексных указателей в одном месте, например из-за сбоя в работе программного кода ОС по причине внешних электромагнитных помех, теряется информация обо всех последующих кластерах файла. Из-за постоянно растущих объемов жестких дисков, а также возрастающих требований к надежности малоразрядные файловые системы быстро вытесняются как системой FAT32, впервые появившейся в Windows 95, OS/2, так и NTFS.

Физическая организация NTFS

Файловая система NTFS была разработана в качестве основной файловой системы для ОС Windows NT в начале 90-х годов с учетом опыта разработки файловых систем FAT и HPFS (основная файловая система для OS/2), а также других существовавших в то время файловых систем. Основными отличительными свойствами NTFS являются:

- поддержка больших файлов и больших дисков объемом до 2 байт;
- восстанавливаемость после сбоев и отказов программ и аппаратуры управления дисками;
- высокая скорость операций, в том числе и для больших дисков;

- низкий уровень фрагментации, в том числе и для больших дисков;
- гибкая структура, допускающая развитие за счет добавления новых типов записей и атрибутов файлов с сохранением совместимости с предыдущими версиями ФС;
- устойчивость к отказам дисковых накопителей;
- поддержка длинных символьных имен;
- контроль доступа к каталогам и отдельным файлам.

Структура тома NTFS. В отличие от разделов FAT все пространство тома NTFS представляет собой либо файл, либо часть файла. Основой структуры тома NTFS является главная таблица файлов (Master File Table, MFT), которая содержит, по крайней мере, одну запись для каждого файла тома, включая одну запись для самой себя. Каждая запись MFT имеет фиксированную длину, зависящую от объема диска, – 1,2 или 4 Кбайт. Для большинства дисков, используемых сегодня, размер записи MFT равен 2 Кбайт, который мы далее будет считать размером записи по умолчанию. Все файлы на томе NTFS идентифицируются номером файла, который определяется позицией файла в MFT. Весь том NTFS состоит из последовательности кластеров, что отличает эту файловую систему от других, где на кластеры делилась только область данных. Порядковый номер кластера в томе NTFS называется логическим номером кластера (Logical Cluster Number, LCN). Файл NTFS также состоит из последовательности кластеров, при этом порядковый номер кластера внутри файла называется виртуальным номером кластера (Virtual Cluster Number, VCN). Базовая единица распределения дискового пространства для файловой системы NTFS – непрерывная область кластеров, называемая отрезком. В качестве адреса отрезка NTFS использует логический номер его первого кластера, а также количество кластеров в отрезке k, то есть пара (LCN, k). Таким образом, часть файла, помещенная в отрезок и начинающаяся с виртуального кластера VCN, характеризуется адресом, состоящим из трех чисел: (VCN, LCN, k). Для хранения номера кластера в NTFS используются 64–разрядные указатели, что дает возможность поддерживать тома и файлы размером до 2^{64} кластеров. При размере кластера в 4 Кбайт это позволяет использовать тома и файлы, состоящие из 64 миллиардов килобайт. Структура тома NTFS показана на рис. 4.23.

Загрузочный блок тома NTFS располагается в начале тома, а его копия – в середине тома. Загрузочный блок содержит стандартный блок параметров BIOS, количество блоков в томе, а также начальный логический номер кластера основной копии MFT и зеркальную копию

MFT. Далее располагается первый отрезок MFT, содержащий 16 стандартных, создаваемых при форматировании записей о системных файлах NTFS. Назначение этих файлов описано в показанной ниже таблице MFT.

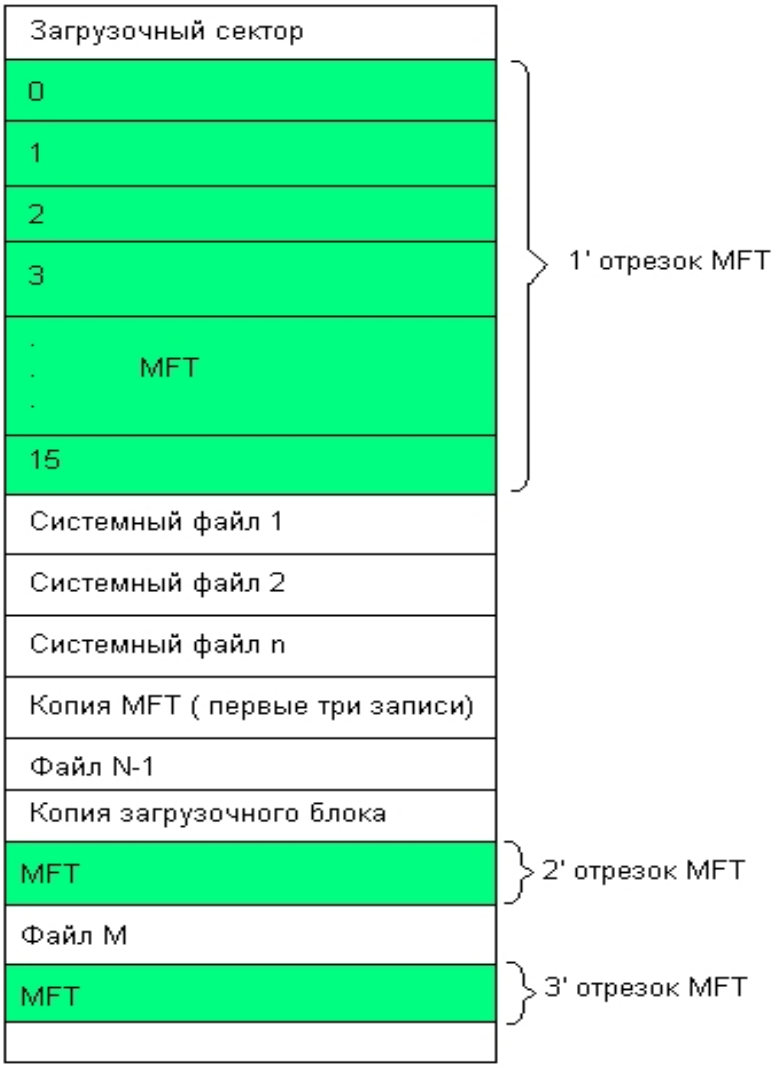


Рис. 4.23. Структура тома NTFS

В NTFS файл целиком размещается в записи таблицы MFT, если это позволяет сделать его размер. В том же случае, когда размер файла больше размера записи MFT, в запись помещаются только некоторые атрибуты файла, а остальная часть файла размещается в отдельном отрезке тома (или нескольких отрезках). Часть файла, размещаемая в записи MFT, называется *резидентной частью*, а остальные части – *нерезидентными*. Адресная информация об отрезках, содержащих нерезидентные части файла, размещается в атрибутах резидентной части. Некоторые системные файлы являются полностью

резидентными, а некоторые имеют и нерезидентные части, которые располагаются после первого отрезка MFT. Нулевая запись MFT содержит описание самой MFT, в том числе и такой ее важный атрибут, как адреса всех ее отрезков. После форматирования MFT состоит из одного отрезка, но после создания первого же несистемного файла для хранения его атрибутов требуется еще один отрезок, так как изначально непрерывная последовательность кластеров MFT уже завершена системными файлами. Из приведенного описания видно, что сама таблица MFT рассматривается как файл, к которому применим метод размещения в томе в виде набора произвольно расположенных нескольких отрезков.

В таблице 4.1 приведено сравнение нескольких файловых систем.

Таблица 4.1

Сравнение файловых систем

Файловая система	FAT32	HPFS	NTFS
Создатель	Microsoft	IBM & Microsoft	Microsoft, Gary Kimura, Tom Miller
Дата представления	1996	1988	1993
ОС или платформа	Windows	OS/2	Windows
Максимальная длина имён файлов	255 байт	255 байт	255 символов
Допустимые символы в названиях	Любые символы Юникода, кроме NUL	Любые символы, кроме NUL	Любые символы Юникода, кроме NUL, " / \ * ? < > | :
Максимальная длина пути файла	Нет установленых ограничений	Нет установленых ограничений	32 767 символов Юникода; каждая компонента пути (каталог или имя файла) – до 255 символов
Максимальный размер файла	4GiB	4GiB	16EiB
Максимальный размер тома	512MiB – 8TiB	2TiB	16EiB

4.4.3. Сетевые операционные системы

Операционная система компьютерной сети во многом аналогична ОС автономного компьютера – она также представляет собой комплекс взаимосвязанных программ, который обеспечивает удобство работы пользователям и программистам путем предоставления им некоторой виртуальной вычислительной системы, и реализует эффективный способ разделения ресурсов между множеством выполняемых в сети процессов.

Компьютерная сеть – это набор компьютеров, связанных коммуникационной системой и снабженных соответствующим программным обеспечением, позволяющим пользователям сети получать доступ к ресурсам этого набора компьютеров. Сеть могут образовывать компьютеры разных типов, которыми могут быть небольшие микропроцессоры, рабочие станции, мини-компьютеры, персональные компьютеры или суперкомпьютеры. Коммуникационная система может включать кабели, повторители, коммутаторы, маршрутизаторы и другие устройства, обеспечивающие передачу сообщений между любой парой компьютеров сети. Компьютерная сеть позволяет пользователю работать со своим компьютером, как с автономным, и добавляет к этому возможность доступа к информационным и аппаратным ресурсам других компьютеров сети.

При организации сетевой работы операционная система играет роль интерфейса, экранирующего от пользователя все детали низкоуровневых программно-аппаратных средств сети. Например, вместо числовых адресов компьютеров сети, таких как MAC-адрес и IP-адрес, операционная система компьютерной сети позволяет оперировать удобными для запоминания символьными именами. В результате в представлении пользователя сеть с ее множеством сложных и запутанных реальных деталей превращается в достаточно понятный набор разделяемых ресурсов.

Сетевые и распределенные ОС. В зависимости от того, какой виртуальный образ создает операционная система для того, чтобы подменить им реальную аппаратуру компьютерной сети, различают сетевые ОС и распределенные ОС.

Сетевая ОС предоставляет пользователю некую виртуальную вычислительную систему, работать с которой гораздо проще, чем с реальной сетевой аппаратурой. В то же время эта виртуальная система не полностью скрывает распределенную природу своего реального прототипа, то есть является виртуальной сетью.

При использовании ресурсов компьютеров сети пользователь сетевой ОС всегда помнит, что он имеет дело с сетевыми ресурсами и что для доступа к ним нужно выполнить некоторые особые операции, например отобразить удаленный разделяемый каталог на вымышленную локальную букву дисковода или поставить перед именем каталога еще и имя компьютера, на котором тот расположен. Пользователи сетевой ОС обычно должны быть в курсе того, где хранятся их файлы, и должны использовать явные команды передачи файлов для перемещения файлов с одной машины на другую. Работая в среде сетевой ОС, пользователь всегда знает, на какой машине выполняется его задание.

Магистральным направлением развития сетевых операционных систем является достижение как можно более высокой степени прозрачности сетевых ресурсов. В идеальном случае сетевая ОС должна представить пользователю сетевые ресурсы в виде ресурсов единой централизованной виртуальной машины. Для такой операционной системы используют специальное название – *распределенная ОС*, или истинно распределенная ОС.

Распределенная ОС, динамически и автоматически распределяя работы по различным машинам системы для обработки, заставляет набор сетевых машин работать как виртуальный унипроцессор. Пользователь распределенной ОС, вообще говоря, не имеет сведений о том, на какой машине выполняется его работа.

Распределенная ОС существует как единая операционная система в масштабах вычислительной системы. Каждый компьютер сети, работающей под управлением распределенной ОС, выполняет часть функций этой глобальной ОС. Распределенная ОС объединяет все компьютеры сети в том смысле, что они работают в тесной кооперации друг с другом для эффективного использования всех ресурсов компьютерной сети.

Два значения термина «сетевая ОС». В настоящее время практически все сетевые операционные системы еще очень далеки от идеала истинной распределенности. Степень автономности каждого компьютера в сети, работающей под управлением сетевой операционной системы, значительно выше по сравнению с компьютерами, работающими под управлением распределенной ОС.

В результате сетевая ОС может рассматриваться как набор операционных систем отдельных компьютеров, составляющих сеть. На разных компьютерах сети могут выполняться одинаковые или разные ОС. Например, на всех компьютерах сети может работать одна и та же ОС UNIX. Более реалистичным вариантом является сеть, в которой

работают разные ОС, например, часть компьютеров работает под управлением UNIX, часть – под управлением NetWare, а остальные – под управлением Windows NT и Windows 2000. Все эти операционные системы функционируют независимо друг от друга в том смысле, что каждая из них принимает независимые решения о создании и завершении своих собственных процессов и управлении локальными ресурсами. Но в любом случае операционные системы компьютеров, работающих в сети, должны включать взаимно согласованный набор коммуникационных протоколов для организации взаимодействия процессов, выполняющихся на разных компьютерах сети, и разделения ресурсов этих компьютеров между пользователями сети. Если операционная система отдельного компьютера позволяет ему работать в сети, то есть предоставлять свои ресурсы в общее пользование и/или потреблять ресурсы других компьютеров сети, то такая операционная система отдельного компьютера также называется *сетевой ОС.*

Таким образом, термин «сетевая операционная система» используется в двух значениях: во-первых, как совокупность ОС всех компьютеров сети и, во-вторых, как операционная система отдельного компьютера, способного работать в сети. Исходя из этого определения следует, что такие операционные системы, как, например Windows NT, NetWare, Solaris, HP-UX, являются сетевыми, поскольку все они обладают средствами, которые позволяют их пользователям работать в сети.

Функциональные компоненты сетевой ОС. На рис. 4.25 показаны основные функциональные компоненты сетевой ОС:

- средства управления локальными ресурсами компьютера реализуют все функции ОС автономного компьютера (распределение оперативной памяти между процессами, планирование и диспетчеризацию процессов, управление процессорами в мультипроцессорных машинах, управление внешней памятью, интерфейс с пользователем и т. д.);
- сетевые средства, в свою очередь, можно разделить на три компонента:

a) средства предоставления локальных ресурсов и услуг в общее пользование – серверная часть ОС;

b) средства запроса доступа к удаленным ресурсам и услугам – клиентская часть ОС;

c) транспортные средства ОС, которые совместно с коммуникационной системой обеспечивают передачу сообщений между компьютерами сети.

Рис. 4.25. Функциональные компоненты сетевой ОС

Упрощенно работа сетевой ОС происходит следующим образом. Предположим, что пользователь компьютера А решил разместить свой файл на диске другого компьютера сети – компьютера В. Для этого он набирает на клавиатуре соответствующую команду и нажимает клавишу Enter. Программный модуль ОС, отвечающий за интерфейс с пользователем, принимает эту команду и передает ее клиентской части ОС компьютера А.

Клиентская часть ОС не может получить непосредственный доступ к ресурсам другого компьютера – в данном случае к дискам и файлам компьютера В. Она может только «попросить» об этом серверную часть ОС, работающую на том компьютере, которому принадлежат эти ресурсы. Эти «просьбы» выражаются в виде сообщений, передаваемых по сети. Сообщения могут содержать не только команды на выполнение некоторых действий, но и собственно данные, например содержимое некоторого файла.

Управляют передачей сообщений между клиентской и серверными частями по коммуникационной системе сети транспортные средства ОС. Эти средства выполняют такие функции, как формирование сообщений, разбиение сообщения на части (пакеты, кадры), преобразование имен компьютеров в числовые адреса, организацию надежной доставки сообщений, определение маршрута в сложной сети и т. д. и т. п.

Правила взаимодействия компьютеров при передаче сообщений по сети фиксируются в коммуникационных протоколах, таких, как

Ethernet, Token Ring, IP, IPX и пр. Чтобы два компьютера смогли обмениваться сообщениями по сети, транспортные средства их ОС должны поддерживать некоторый общий набор коммуникационных протоколов. Коммуникационные протоколы переносят сообщения клиентских и серверных частей ОС по сети, не вникая в их содержание.

На стороне компьютера В, на диске которого пользователь хочет разместить свой файл, должна работать серверная часть ОС, постоянно ожидающая прихода запросов из сети на удаленный доступ к ресурсам этого компьютера. Серверная часть, приняв запрос из сети, обращается к локальному диску и записывает в один из его каталогов указанный файл. Конечно, для выполнения этих действий требуется не одно, а целая серия сообщений, переносящих между компьютерами команды ОС и части передаваемого файла.

Очень удобной и полезной функцией клиентской части ОС является способность отличить запрос к удаленному файлу от запроса к локальному файлу. Если клиентская часть ОС умеет это делать, то приложения не должны заботиться о том, с локальным или удаленным файлом они работают, – клиентская программа сама распознает и перенаправляет (redirect) запрос к удаленной машине. Отсюда и название, часто используемое для клиентской части сетевой ОС, – редиректор. Иногда функции распознавания выделяются в отдельный программный модуль, в этом случае редиректором называют не всю клиентскую часть, а только этот модуль.

Клиентские части сетевых ОС выполняют также преобразование форматов запросов к ресурсам. Они принимают запросы от приложений на доступ к сетевым ресурсам в локальной форме, то есть в форме, принятой в локальной части ОС. В сеть же запрос передается клиентской частью в другой форме, соответствующей требованиям серверной части ОС, работающей на компьютере, где расположен требуемый ресурс. Клиентская часть также осуществляет прием ответов от серверной части и преобразование их в локальный формат, так что для приложения выполнение локальных и удаленных запросов неразличимо.

Сетевые службы и сетевые сервисы. Совокупность серверной и клиентской частей ОС, предоставляющих доступ к конкретному типу ресурса компьютера через сеть, называется сетевой службой. В приведенном выше примере клиентская и серверная части ОС, которые совместно обеспечивают доступ через сеть к файловой системе компьютера, образуют файловую службу.

Говорят, что сетевая служба предоставляет пользователям сети некоторый набор услуг. Эти услуги иногда называют также *сетевым*

сервисом (от англоязычного термина «service»). Необходимо отметить, что этот термин в технической литературе переводится и как «сервис», и как «услуга», и как «служба». Хотя указанные термины иногда используются как синонимы, следует иметь в виду, что в некоторых случаях различие в значениях этих терминов носит принципиальный характер. Далее в тексте под «службой» мы будем понимать *сетевой компонент*, который реализует некоторый набор услуг, а под «сервисом» – описание того набора услуг, который предоставляется данной службой.

Таким образом, сервис – это интерфейс между потребителем услуг и поставщиком услуг (службой).

Сервисы Интернет – сервисы, предоставляемые в сети пользователям, программам, системам, уровням, функциональным блокам. В сети Интернет сервисы предоставляют сетевые службы. Наиболее распространенными Интернет-сервисами являются: хранение данных; передача сообщений и блоков данных; электронная и речевая почта; организация и управление диалогом партнеров; предоставление соединений; проведение сеансов; видео-сервис.

Каждая служба связана с определенным типом сетевых ресурсов и/или определенным способом доступа к этим ресурсам. Например, служба печати обеспечивает доступ пользователей сети к разделяемым принтерам сети и предоставляет сервис печати, а почтовая служба предоставляет доступ к информационному ресурсу сети – электронным письмам.

Способом доступа к ресурсам отличается, например, служба удаленного доступа – она предоставляет пользователям компьютерной сети доступ ко всем ее ресурсам через коммутируемые телефонные каналы. Для получения удаленного доступа к конкретному ресурсу, например к принтеру, служба удаленного доступа взаимодействует со службой печати. Наиболее важными для пользователей сетевых ОС являются файловая служба и служба печати.

Среди сетевых служб можно выделить такие, которые ориентированы не на простого пользователя, а на администратора. Сетевая служба может быть представлена в операционной системе либо обеими (клиентской и серверной) частями, либо только одной из них.

Сетевые службы по своей природе являются *клиент-серверными* системами. Поскольку при реализации любого сетевого сервиса естественно возникает источник запросов (клиент) и исполнитель запросов (сервер), то и любая сетевая служба содержит в своем составе две несимметричные части – клиентскую и серверную (рис. 4.26).

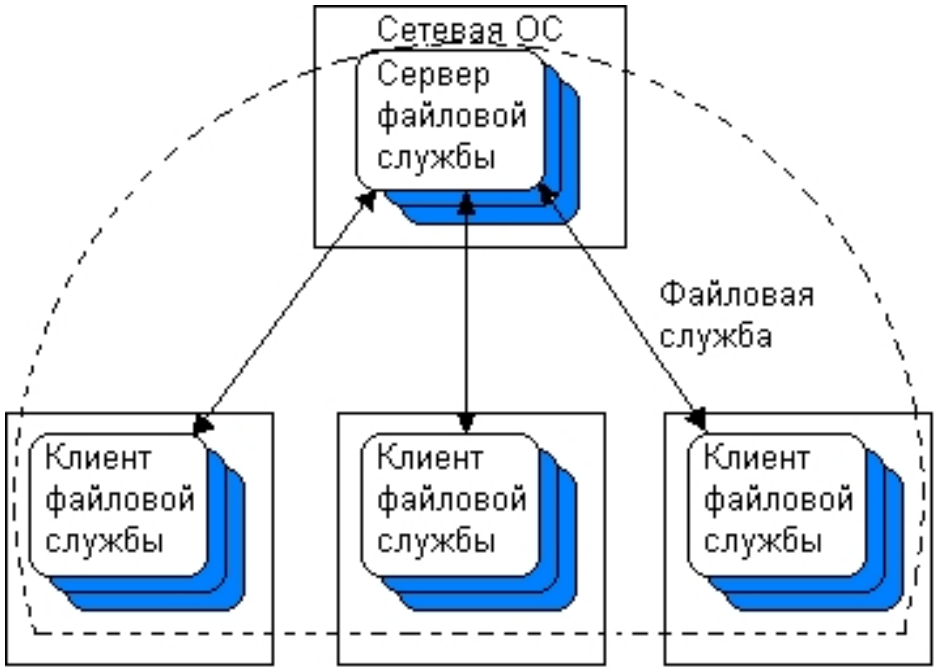


Рис. 4.26. Клиент-серверная природа сетевых служб

Обычно говорят, что сервер предоставляет свои ресурсы клиенту, а клиент ими пользуется. Принципиальной же разницей между клиентом и сервером является то, что инициатором выполнения работы сетевой службой всегда выступает клиент, а сервер всегда находится в режиме пассивного ожидания запросов. Например, почтовый сервер осуществляет доставку почты на компьютер пользователя только при поступлении запроса от почтового клиента.

Контрольные вопросы и задания

1. Расскажите об ОС как о виртуальной машине.
2. Опишите ОС как систему управления ресурсами.
3. Дайте понятие «процесса» и «потока».
4. Начертите и объясните граф состояний переходов процесса из одной фазы в другую.
5. Дайте понятие виртуального адресного пространства.
6. Понятие вытесняющей и невытесняющей многозадачности.
7. Опишите принцип мультипрограммирования на основе прерываний.
8. В чём назначение прерываний? Перечислите типы прерываний.
9. Опишите механизм прерываний.
10. Опишите механизм диспетчеризации и приоритезации прерываний в ОС.
11. Назначение и функции подсистемы управления ресурсами.
12. Опишите задачи ОС по управлению файлами и устройствами.

13. Дайте понятие логической организации файловой системы и логической организации файла.
14. Дайте понятие физической организации файловой системы.
15. Приведите примеры и характеристики файловых систем. Сделайте сравнительный анализ ФС.
16. Опишите физическую организацию FAT.
17. Опишите физическую организацию NTFS.
18. Основные понятия сетевых операционных систем.

5. ПРИКЛАДНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Прикладные программы предназначены для того, чтобы обеспечить применение вычислительной техники в различных сферах деятельности человека.

ППП определяется и как совокупность программ для решения определенного класса задач (рис. 5.1).

Под классом задач понимается множество прикладных проблем, обладающих общностью применяемых алгоритмов и информационных массивов, а также определение пакета как комплекса взаимосвязанных программ, обладающих специальной организацией, которая обеспечивает значительное повышение производительности труда программистов и пользователей пакета. Одной из главных особенностей является ориентация ППП не на отдельную задачу, а на некоторый класс задач, включающий и специфические задачи предметной области. Рассмотрим более подробно ППО общего назначения и проблемно-ориентированные, которые в свою очередь можно разделить на ПС специального и профессионального назначения.

Интегрированной программной системой назовем комплекс программ, элементами которого являются различные пакеты и библиотеки программ. Примером служат системы автоматизированного проектирования, имеющие в своем составе несколько ППП различного назначения. Часто в подобной системе решаются задачи, относящиеся к различным классам или даже к различным предметным областям.

Следует указать на отсутствие четких и однозначных границ между перечисленными на рис. 5.1 формами прикладного программного обеспечения. Как и почти всякая классификация, приведенная на рисунке, не является единственно возможной.

В ней представлены даже не все виды прикладных программ. Тем не менее, использование классификации полезно для создания общего представления о ППО.

5.1. Инструментальные программные средства общего назначения

5.1.1. Средства представления, хранения и обработки текстовой и числовой информации

Текстовые редакторы и издательские системы

Программы для работы с текстами – текстовые редакторы и издательские системы. Несмотря на широкие возможности использования компьютеров для обработки различной информации,

самыми популярными являются программы, предназначенные для работы с текстами – *текстовые редакторы и издательские системы.*

Рис. 5.1. Состав ППП общего назначения и проблемно-ориентированных ПС

Например, *MS Word, Лексикон, Слово и Дело* и издательские системы *Corel Ventura*, *Page Maker*. Это ПО для набора, редактирования и подготовки к печати любых документов от маленьких заметок или договора на одну страничку до многотомной энциклопедии или цветного иллюстрированного журнала. Эксперты оценивают использование компьютера в качестве печатающей машинки в 80 %.

На рис. 5.2 изображено окно Microsoft Word с документом, содержащим текст, таблицу и элементы графики.

Элементы издательского дела. Для того, чтобы уверенно работать с текстовыми редакторами и настольными издательскими системами, необходимо освоить и уяснить некоторые сведения из издательского дела.

Особую значимость при подготовке и формировании текста для издания имеют шрифты. *Шрифты* – основное изобразительное средство издательских систем, с их помощью можно добиться большой художественной выразительности текста.

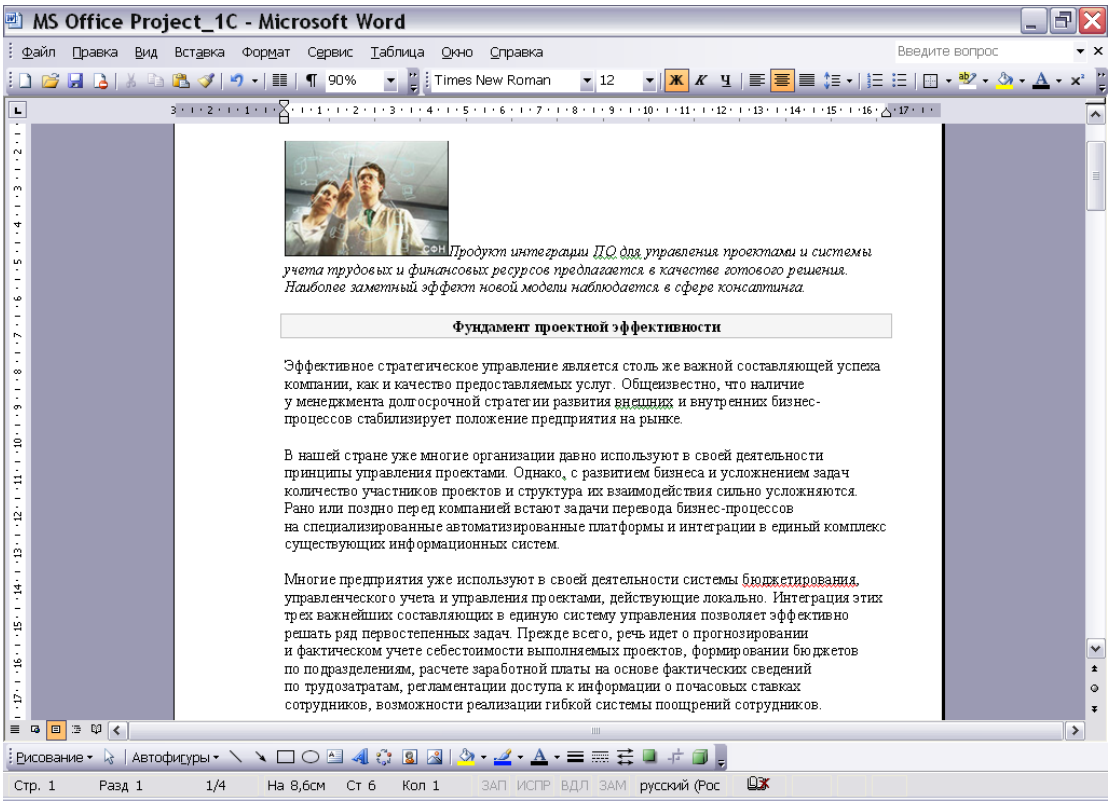


Рис. 5.2. Окно Microsoft Word

Шрифты различают по *гарнитуре* (рисунку), *начертанию, размеру* и *назначению.* Гарнитурой называется совокупность шрифтов одного рисунка во всех начертаниях и кеглях. Кегль – размер шрифта, определяемый размером литеры по вертикали, исчисляемый в *пунктах* (1 пункт равен 0,367 мм). Полный комплект гарнитуры содержит шрифты всех начертаний и кеглей, а в каждом кегле – русский и латинский (и, если нужно, другие) алфавиты прописных и строчных букв, а также относящиеся к ним знаки.

Различия между буквами разных шрифтов объясняется их различным построением. Среди элементов, из которых строятся буквы, выделяют (рис.5.3):

- основные штрихи (задают структуру буквы);
- дополнительные штрихи (играют вспомогательную и соединительную роль);
- засечки;

- верхние и нижние выносные элементы;
- овалы и полуовалы (с наплывами или без них);
- концевые элементы.

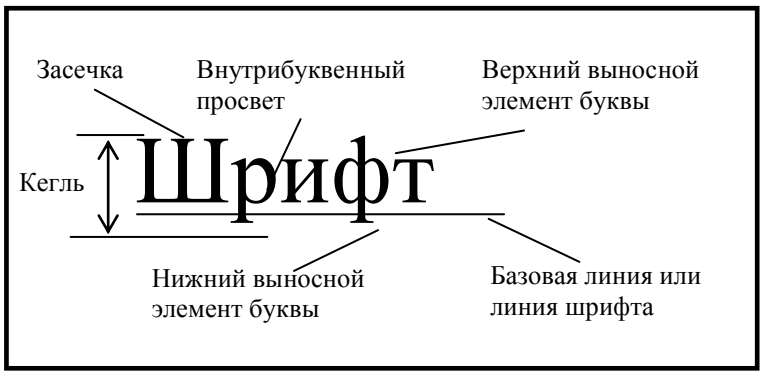


Рис.5.3.Состав шрифта

Буквы располагаются по *базовой линии.* Расстояние между строками называются *интерлиньяжем.* Отношение толщины основных и дополнительных элементов определяет контрастность шрифта. Различают неконтрастные, малоконтрастные, контрастные и очень контрастные шрифты. Форму букв шрифта определяют *цветность* и *ритм* (соотношение черного и белого, просветы, наплывы и пр.). Отношение высоты буквы к ее ширине называют шириной шрифта. Бывают сверхузкие, узкие, нормальные, широкие и сверхширокие шрифты.

Насыщенность шрифта определяется *светлотой.* Бывают сверхсветлые, светлые, нормальные, полужирные, жирные и сверхжирные шрифты. Шрифты могут быть прямыми и наклонными. Наклонный вариант шрифтов часто называют *курсивом.* Шрифты одного типа, но отличающиеся по ширине, насыщенности и наклону являются различными *начертаниями* одного и того шрифта. Начертания одного шрифта составляют *гарнитуру.*

Шрифт на компьютере – это файл или группа файлов, обеспечивающих вывод текста на печать со стилевыми особенностями шрифта. Существуют программы, позволяющие создавать собственные варианты шрифтов. Как правило, такие программы входят в состав текстовых редакторов и издательских систем. Однако существует довольно большой спектр стандартных (в компьютерном смысле) шрифтов, разработанных полиграфистами. Приведем некоторые из них: Times New Roman, Arial, Courier New, **Arial Black, Impact** и др.

Наиболее часто используются следующие гарнитуры:

1. *Литературная* – для набора всех видов изданий, кроме букварей, энциклопедий, а также малоформатных, нормативных, периодических и литературно-художественных изданий.

2. *Обыкновенная новая* – для набора всех видов книжно–журнальных изданий, кроме учебников для школы, карманных и нормативных изданий.

3. *Банниковская* и *Академическая* – для набора литературно–художественных, научных, учебных и научно-популярных изданий по гуманитарным областям знаний.

4. *Школьная* – для набора учебников начальной и средней школы, изданий для детей, художественной и научно-популярной литературы, нормативных изданий и массовых иллюстрированных журналов.

5. *Балтика* и *Таймс* – для набора художественной и научно-популярной литературы, вузовских учебников.

Каждый текст, подготовленный к изданию в качестве брошюры или книги, должен пройти техническое редактирование, которое предполагает подготовку *макета* готового издания. Издательским текстовым оригинал-макетом является набранный, сверстанный на компьютере и отпечатанный на лазерном или струйном принтере текстовый оригинал, представляющий собой точный прообраз будущего издания (по числу страниц, абзацев, рисунку шрифта), предназначенный для изготовления печатной формы.

Исходным материалом для подготовки оригинала является отредактированный, вычитанный автором и литературным редактором размеченный текстовый оригинал. Техническое редактирование – процесс достаточно сложный, особенно если в тексте есть таблицы, схемы, формулы и иллюстрации.

Поскольку текстовые материалы бывают различной сложности по набору и вёрстке, существуют и развиваются различные программы обработки текстов. Их можно классифицировать по уровням требований к обработке текстов. Перечислим наиболее популярные из них.

Программы для набора и обработки простых текстов:

- MultiEdit;
- Блокнот;
- WordPad.

Программы для набора сложных текстов:

- ChiWriter;
- TechWord;
- Word;

- Word for Windows;
- WordPerfect;
- TeX;
- LaTex и др.

Разработчики программ текстовых редакторов стараются предусмотреть в них предоставление пользователю всех необходимых операций и сервисных возможностей для эффективной обработки текстов. Выделим главные из них:

- набор текста в интерактивном режиме;
- редактирование текста;
- работа с фрагментами текста (копирование, перемещение, удаление);
- форматирование текста (установка абзаца, перенос, выравнивание границ строки и т.п);
- работа с несколькими текстами одновременно посредством многооконного принципа;
- файловая организация работы с текстами и взаимодействие с операционной системой;
- импорт/экспорт текстов из одного формата в другой, в другие прикладные темы;
- работа с разными шрифтами;
- работа со спецсимволами (математические знаки, индексы и т.п.);
- работа с иллюстративным материалом (таблицы, схемы);
- проверка правописания;
- поиск и замена фрагментов текста.

Издательские системы. Настольные компьютерные издательские системы приобрели широкую популярность в различных сферах производства, бизнеса, науки, культуры и образования. Издательское дело становится актуальным практически для любой организации. Выпуск информационных бюллетеней, рекламных проспектов, собственных малотиражных газет и даже книг теперь становится необходимым атрибутом информационного обеспечения современных учреждений. Пожалуй, из всех информационных технологий компьютерное издательство является наиболее массовой и практически легко внедряемой технологией.

Настольные издательские системы (desktor publishing) представляют собой комплекс аппаратных и программных средств, предназначенных для компьютерного набора, верстки и издания текстовых и иллюстративных материалов.

Существуют различные программные системы, среди них наиболее распространены следующие: Express Publisher, Illustrator for Windows, Ventura Publisher, PageMarker, TeX. Перечисленные программные системы предназначены для компьютерной верстки. Среди программ подготовки иллюстраций можно выделить следующие: CorelDraw, CorelSystem, Designer, DrawPerfect, GalleryEffect, PC Paintbrush, PhotoStuler, Adobe Photoshop и др.

Для издательских систем существуют различные сервисные программы обработки текстовых материалов. К этому классу ПО можно отнести и *программы проверки правописания*, *программы-переводчики*, например, *Stylus* (дают не только письменный перевод введенных слов, но и устный, что облегчает понимание и усваивание слов написанных на иностранном языке), *программы-словари* (Lingvo), *программы распознавания образов* (Fine Reader) и т.д.

Среди сервисных программ можно выделить 7 основных групп:

- преобразования растровой графики в векторную;
- обработки сканированных изображений;
- обработки шрифтов;
- проверки правописания;
- чтения текстов с помощью сканера;
- русификации программ;
- программы-переводчики.

Работа с издательскими системами предусматривает

- работу с меню и с диалоговой панелью;
- работу с текстом и иллюстрациями (набор и формирование);
- редактирование текста и иллюстраций;
- оформление текста (форматирование; выбор шрифтов, гарнитуры, стиля, размеров и т. п.);
- настройку экрана.

Издательские системы имеют стандартные правила работы с меню, командами, сервисными утилитами.

Приведём примеры программ для издательских систем.

QuarkXPress 7.31 Passport

QuarkXPress – это мощная издательская система, обладающая интуитивным интерфейсом и расширенным набором инструментов для обработки текста, управления цветом и графическими элементами и проектирования web-страниц. QuarkXPress широко используется в книжных, газетных и журнальных издательствах, рекламных и маркетинговых агентствах, дизайнерских фирмах и типографиях.

Adobe PageMaker 7.0.2 GB

Приложение Adobe PageMaker 7.0 предлагает высококачественные инструменты для профессиональных дизайнеров и других специалистов, в чьи обязанности входит вёрстка и предпечатная подготовка различных публикаций, например брошюр или официальных бланков. Коллекция шаблонов и готовых изображений, а также интуитивно понятные дизайнерские инструменты помогут вам приступить к работе немедленно. Интеграция с другими приложениями Adobe сделает работу более продуктивной.

Microsoft Office Publisher 2007 Academic (Russian) Edition

Office Publisher 2007, русская версия, помогает самостоятельно создавать, настраивать широкий диапазон маркетинговых материалов, а также обмениваться ими. Новые и усовершенствованные возможности обеспечивают пошаговое руководство в процедурах создания маркетинговых материалов и их распространения в печати, через Интернет и по электронной почте, что помогает создавать корпоративный стиль, управлять списками клиентов и отслеживать ход маркетинговых кампаний.

Corel Ventura 10 Лицензия

Одно из самых мощных решений для создания бизнес–публикаций. Это уникальное средство позволяет преобразовывать большие сложные документы, включая документы XML, в наглядные публикации с богатыми возможностями форматирования.

TeX

T_EX (обычным текстом – TeX) – система компьютерной вёрстки, разработанная американским профессором информатики Дональдом Кнутом в целях создания компьютерной типографии. В неё входят средства для секционирования документов, для работы с перекрёстными ссылками и для набора сложных математических формул.

Название произносится как «тех» (от греч. τέχνη – «искусство», «мастерство»). В написании буква E опущена ниже T и X.

Документы набираются на собственном языке разметки в виде обычных ASCII–файлов, содержащих информацию о форматировании текста или выводе изображений. Эти файлы (обычно имеющие расширение «.tex») транслируются специальной программой в файлы «.dvi» (device independent – «независимые от устройства»), которые могут быть отображены на экране или напечатаны. DVI-файлы можно специальными программами преобразовать в PostScript, PDF или другой электронный формат.

Ядро TeX’a представляет собой язык низкоуровневой разметки, содержащий команды отступа и смены шрифта. Огромные возможности в TeX’е предоставляют готовые наборы макросов и расширений. Наиболее распространённые расширения стандартного TeX’a (наборы шаблонов, стилей и т.д): *LaTeX* (произносится «латéх») и *AMS–TeX*. При использовании пакета расширения LaTeX можно превратить разросшуюся статью в книгу изменением одного слова в исходнике, вставлять оглавление одной командой, не задумываться о нумерации разделов, теорем, рисунков. Есть много пакетов для оформления химических формул (например, пакет XymTeX), диаграмм (xypic), создания презентаций и визитных карточек и тому подобного.

Распространённые комплекты вёрстки на основе TeX’a: для Windows – TeX Live или MikTeX, для UNIX-подобных систем – TeX Live или teTeX.

Для создания шрифтов совместно с TeX’ом используется специально разработанная система METAFONT, в которой шрифты описываются программами на специализированном языке Meta. Могут также использоваться векторные шрифты в формате PostScript Type 1, TrueType и OpenType.

BusinessCards MX. При помощи этой программы вы легко и быстро запроектируете и распечатаете профессиональные визитки.

Suitcase Fusion Windows 11 ESD VLA. Менеджер шрифтов. Содержит созможности по установке, упорядочиванию, поиску, просмотру шрифтов. Suitcase позволит быстро найти и установить нужный шрифт, создать страницу-образец.

Azalea Bar Code Software. Семейство системных шрифтов и утилит для печати штрих-кодов и этикеток.

FinePrint WorkStation version. Виртуозная печать из любых Windows-приложений экономит ваше время, силы и расходные материалы.

Программы для работы с графикой

Большую популярность приобрели программы обработки графической информации. Компьютерная графика в настоящее время является одной из самых динамично развивающихся областей программного обеспечения. Она включает в себя обработку и вывод графической информации – чертежей, рисунков, картин, текстов и т.д. – средствами компьютерной техники.

Различные типы *графических систем* (Adobe Photoshop, Adobe Illustrator, Corel Draw, Paintbrush, Alias Power Animator, 3d Studio Max, программа для обработки видео Premier) позволяют быстро строить

изображения, вводить иллюстрации с помощью сканера или видеокамеры, создавать анимационные ролики. Графические редакторы позволяют пользоваться инструментарием художника, стандартными библиотеками изображений, наборами стандартных шрифтов, редактированием изображений, копированием и перемещением фрагментов по страницам экрана и др. Для профессиональных дизайнеров и всех тех, чей род занятий связан с обработкой и созданием изображений компания *Adobe* (http://www.adobe.com) предлагает несколько программных продуктов (рис.5.4).

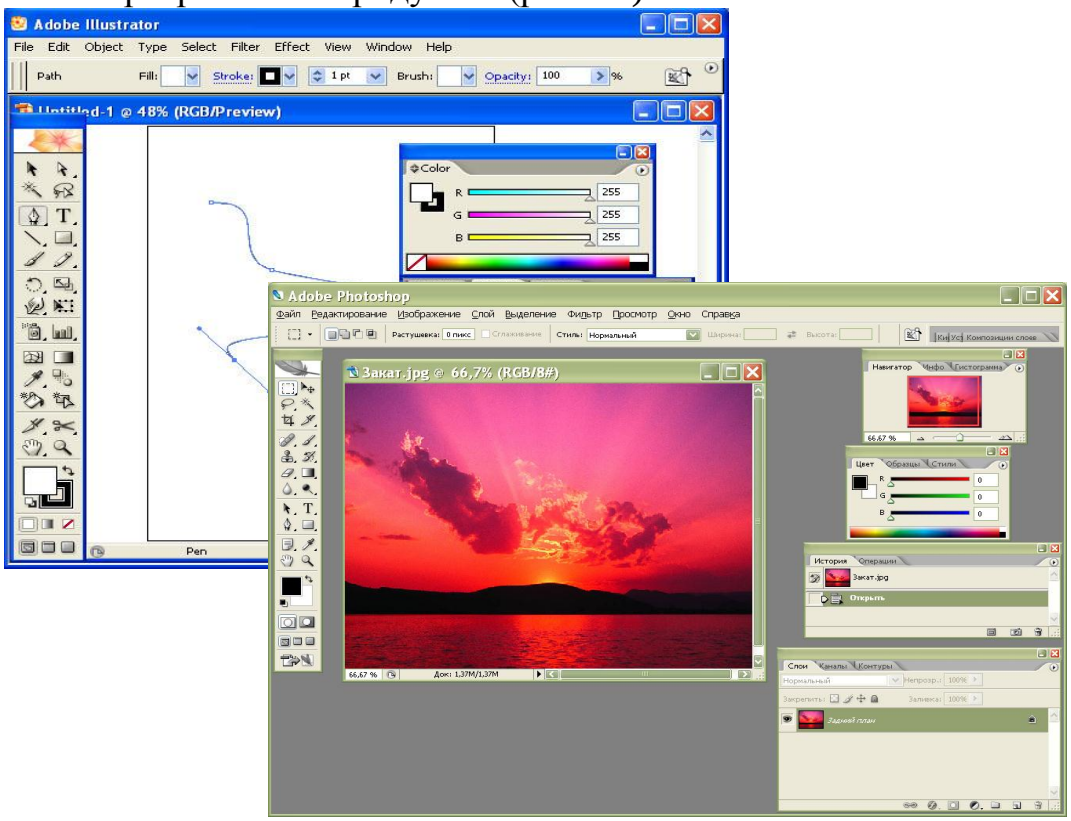


Рис. 5.4. Графические редакторы Adobe Illustrator и Photoshop

Adobe Photoshop – своего рода эталон растровых редакторов. Не так давно была представлена очередная версия этого редактора под названием *Adobe Photoshop CS.* Здесь аббревиатура расшифровывается как *Creative Suite (набор для творчества).*

Основные рабочие качества Photoshop:

1. Возможность создания многослойного изображения. При этом каждый элемент может быть сохранён в собственном, отдельном слое, который редактируется отдельно, перемещается относительно других слоёв и так далее. Конечное изображение можно сохранить как в оригинальном, «многослойном» виде (формат PSD), так и слить все

слои в один, переводя готовую картинку в один из стандартных форматов – JPG, GIF, TIF и так далее.

2. Улучшенные инструменты для работы с текстом. Начиная с шестой версии программы, вы можете добавлять текстовые вставки в любой участок изображения, «набивая» текст прямо поверх картинки. В дальнейшем текст можно редактировать, указав на него мышкой.

3. Около 100 разнообразных фильтров и спецэффектов, возможность подключения дополнительных plug-ins.

4. Несколько десятков инструментов для рисования, вырезания контуров изображения.

5. Богатейшие возможности совмещения изображений, работа с текстурами.

6. Возможность работы с десятками популярных графических форматов.

7. Профессиональные инструменты для выделения и редактирования отдельных участков изображения.

8. Формат файлов, общий для платформ PC и Mac.

9. Возможность многоступенчатой отмены внесённых изменений.

Электронные таблицы

Для выполнения расчетов и обработки числовой информации существуют специальные программы – электронные таблицы (рис. 5.5).

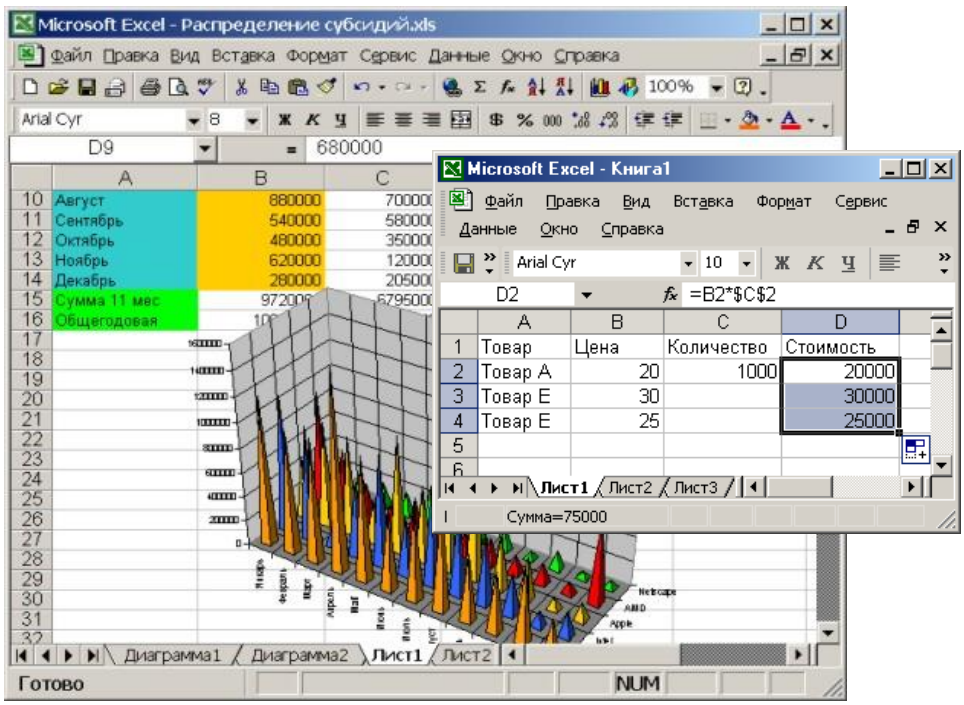


Рис. 5.5. Окно Microsoft Excel с таблицей данных и трехмерной диаграммой

В процессе деятельности любого специалиста часто требуется представить результаты работы в виде таблиц, где одна часть полей занята исходными данными, а другая–результатами вычислений и графического анализа. Характерными для них является большой объем перерабатываемой информации, необходимость многократных расчетов при изменении исходных данных. Автоматизацией подобной рутинной работы и занимаются электронные таблицы.

Табличный процессор MS Excel является одним из наиболее популярных пакетов программ, предназначенных для создания табличных документов. Он обладает мощными вычислительными возможностями, средствами деловой графики, обработки текстов, ведения баз данных.

Работа с табличным процессором Excel позволяет:

– использовать для хранения взаимосвязанных таблиц рабочую книгу, состоящую из отдельных листов, которые можно в процессе работы удалять, переименовывать, переставлять местами, копировать и скрывать;

- применять средства корректировки данных в таблице, используя широкий спектр возможностей работы с фрагментами;
- для расчетов, помимо написания формул, использовать большой набор встроенных функций, для задания которых может применяться Мастер функций;
- применять имена для ссылки на часто используемые диапазоны ячеек;
- оформлять таблицы с применением разнообразных шрифтов, выравнивания текста и чисел, изменения ширины столбцов и высоты строк, затенения и обрамления ячеек; Excel позволяет также применять встроенные форматы оформления таблиц (автоформатирование);
- применять разнообразные форматы отображения числовых данных;
- для графического представления данных рабочего листа применять различные диаграммы; должным образом оформлять их и печатать;
- дополнять рабочие книги рисунками и другими графическими объектами для демонстрации данных, диаграмм и графических объектов;
- выполнять свод данных из нескольких таблиц путем их консолидации;
- сортировать данные в таблице, выполнять отбор данных из таблицы по определенному критерию;

- вводить и корректировать информацию в таблице с использованием формы данных;
- автоматически рассчитывать промежуточные итоги;
- решать задачи по оптимизации данных, проводить статистический анализ данных;
- осуществлять обмен данными с другими приложениями;
- выполнять запросы к базам данных с применением средств MS-Query;
- создавать собственные пользовательские функции, макро-команды и программы с использованием языка Visual Basic.

Область электронной таблицы, содержащую данные определенной структуры, можно рассматривать как *базу данных*, при этом столбцы называются полями, а строки – записями. Столбцам присваиваются имена, которые будут использоваться как имена полей записей. Работа с любой базой данных заключается в поиске информации по определенным критериям, перегруппировке и обработке информации.

Системы управления базами данных

Одним из наиболее перспективных направлений развития вычислительной техники является создание специальных аппаратных средств для хранения гигантских массивов информационных данных и последующей нечисловой обработки их поиска и сортировки. Для компьютерной обработки подобных *баз данных* используют *системы управления базами данных (СУБД). База данных (БД)* – это интегрированная совокупность взаимосвязанных данных. Обычно база данных создается для одной конкретной предметной области, организации или прикладной задачи.

СУБД – это набор средств программного обеспечения, необходимых для создания, обработки и вывода записей баз данных. Различают несколько моделей данных: *иерархические, сетевые, реляционные.* При работе с СУБД выделяют несколько последовательных этапов:

- проектирование базы данных;
- создание структуры базы данных;
- заполнение базы данных;
- просмотр и редактирование базы данных;
- сортировку базы данных;
- поиск необходимой записи;
- выборку информации;
- создание отчетов.

Как правило, большинство популярных систем управления базами данных поддерживают эти этапы и предоставляют удобный инструментарий для их реализации.

Система управления базами данных MS Access. MS Access – наиболее популярная на сегодняшний день СУБД для персональных компьютеров. Она представляет собой систему обслуживания реляционных баз данных с графической оболочкой. Данные в таких базах оформляются в виде одной или нескольких таблиц, состоящих из однотипных записей. Система обслуживания включает в себя ввод данных в ЭВМ, отбор данных по каким-либо признакам (критериям или параметрам), преобразование структуры данных, вывод данных, являющихся результатом решения задач в табличном или каком-либо ином удобном для пользователя виде. MS Access (рис. 6.6) позволяет создавать связанные объекты и устанавливать ссылочную целостность данных. MS Access поддерживает встраивание OLE-объектов (Object Linking and Embedding) в рамках среды Windows. В состав пакета MS Access входит также ряд специализированных программ, решающих отдельные задачи (так называемых «мастеров»).

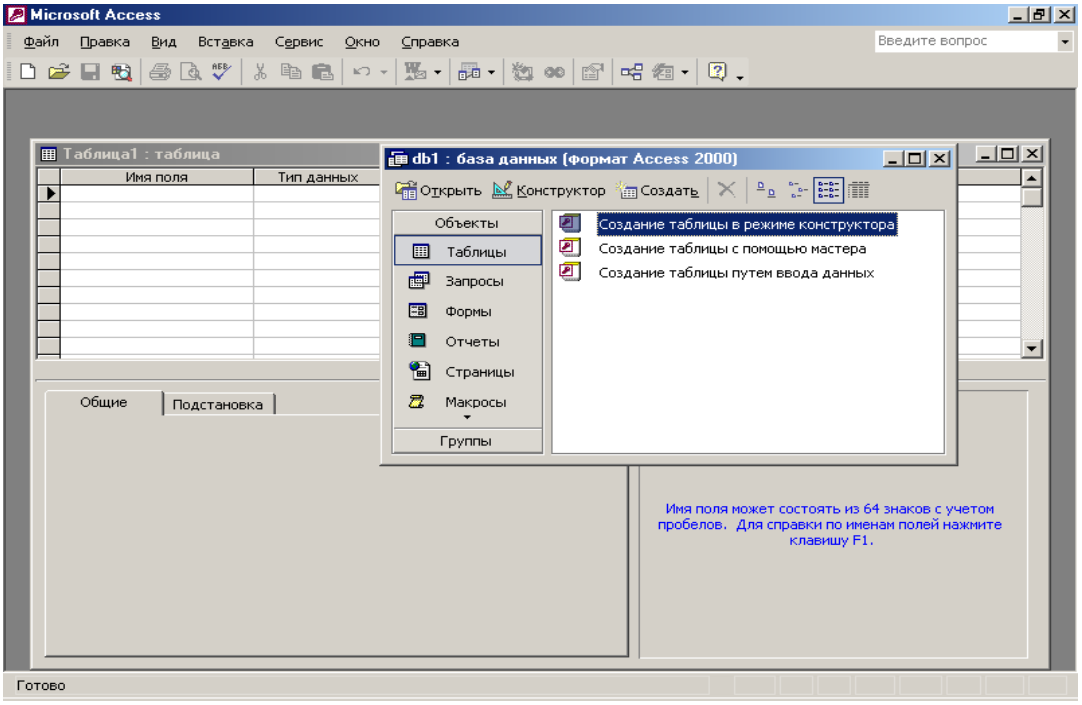


Рис. 5.6. Окно СУБД MS Access

Окно базы данных состоит из шести вкладок. В этом окне осуществляются все операции обработки входящих в базу объектов. Их перечень соответствует ярлыкам вкладок в верхней части окна базы данных. При создании новой базы данных список объектов в каждой

вкладке пуст. В Access-базе данных различают следующие *типы объектов.*

Таблица – набор данных по конкретной теме. Данные таблицы хранятся в записях (строках), состоящих из отдельных *полей* (столбцов). В БД Microsoft Access все данные хранятся в виде таблиц.

Запрос позволяет выбрать из БД только необходимую информацию, т.е. ту, которая соответствует определенному условию и нужна для решения определенной задачи.

Форма представляет собой бланк, подлежащий заполнению, или маску-формуляр, позволяющую ограничить объем информации, доступной пользователю.

Отчет предназначен для печати любого набора данных, оформленного соответствующим образом.

Макрос автоматизирует выполнение конкретной операции БД без программирования.

Модуль содержит программы на языке Visual Basic, применяемые для настройки, оформления и расширения БД.

Таблицы, запросы, формы, отчеты, макросы и модули – это самостоятельные объекты, сохраняющиеся в *общем файле базы.*

Обзор существующих СУБД, предлагаемых ведущими фирмами-разработчиками ПО вы найдете ниже.

Интегрированные пакеты программ

Желание объединить функции различных прикладных программ в единую систему привело к созданию интегрированных систем. Универсальные *интегрированные системы* разрабатывались по принципу единой системы, содержащей в качестве элементов текстовые и графические редакторы, электронные таблицы и систему управления базами данных, например Framework, Works, Мастер. Современная концепция интеграции программных средств – кооперация отдельных прикладных программных систем по типу широко известного пакета MicroSoft Office. Сами системы, входящие в пакет, являются независимыми, более того, они сами представляют локально интегрированный пакет, поскольку помимо основной своей задачи поддерживают функции других систем. Например, текстовый редактор Word обладает возможностью манипулировать с электронными таблицами и базами данных, а в электронной таблице Excel встроен мощный текстовый редактор. Для сопряжения информационных данных из различных программных систем в них предусматривают импорт-экспортную систему обмена с перекодировкой форматов представления данных.

Интегрированные пакеты программ – это комплекс полностью совместимых между собой программ «на все случаи жизни», призванный составить для пользователя единую в своей основе комфортную деловую среду.

Microsoft ПО MS Works 9.0 Win32 Russian – популярнейший в мире интегрированный программный продукт для Windows – Microsoft Works. Он обеспечивает ведущий набор основных средств повышения продуктивности для облегчения повседневных домашних вычислительных задач. Works включает текстовый процессор, средства работы с электронными таблицами, с базами данных, рисования и коммуникации – все в одном интегрированном продукте вместе с Microsoft Internet Explorer.

Банк Freeware: Офис – 350 программ. Коллекция свободно распространяемых приложений Офис. 350 программ из серии Банк freeware – уникальный сборник программ для оптимизации офисной работы.

Microsoft ПО MS Office Pro 2007 Win32 Russian CD Microsoft Office – профессиональный 2007, исчерпывающий комплект рабочих приложений и программ управления базами данных, помогающих экономить время и эффективно организовать работу. Широкий спектр возможностей управления контактами.

Программное обеспечение APPLE iWork'08 – очень удобный офисный пакет, с помощью которого можно готовить текстовые документы и красочные презентации. В состав Apple iWork входят два приложения: текстовый процессор Pages 2 и программа создания презентаций Keynote 3.

5.1.2. Телекоммуникационные и сетевые программы

Телекоммуникационная сеть – это сложный комплекс взаимосвязанных и согласованно функционирующих программных и аппаратных компонентов. Программную платформу сети образуют ОС. От того, какие концепции управления локальными и распределенными ресурсами положены в основу сетевой ОС, зависит эффективность работы всей сети. При проектировании сети важно учитывать, насколько просто данная операционная система может взаимодействовать с другими ОС сети, насколько она обеспечивает безопасность и защищенность данных, до какой степени она позволяет наращивать число пользователей, можно ли перенести ее на компьютер другого типа и многие другие соображения.

Самым верхним слоем сетевых средств являются различные сетевые приложения, такие, как сетевые базы данных, почтовые системы, средства архивирования данных, системы автоматизации коллективной работы и др. Очень важно представлять диапазон возможностей, предоставляемых приложениями для различных областей применения, а также знать, насколько они совместимы с другими сетевыми приложениями и операционными системами.

Некоторые примеры данного ПО:

- интернет-броузеры (Netscape Navigator);
- терминалы (TeleMax, Hyper Terminal);
- почтовые редакторы (GoldED).

5.2. Инструментальные программные средства специального назначения

5.2.1. Гиперсреды

В последнее время широкую популярность получили программы обработки гипертекстовой информации. *Гипертекст* – это текст, представленный в виде ассоциативно связанных автономных блоков. Это форма организации текстового материала не в линейной последовательности, а в форме указаний возможных переходов (ссылок), связей между отдельными его фрагментами.

Название «гипертекст» впервые было предложено Т.Нельсоном. Модель гипертекста определяется графом, в вершинах которого располагаются части (блоки) текста, которые содержат какие-либо сведения, предложения, мысли, обобщения. Эти блоки имеют смысловую связь, фиксируемую ребрами графа.

Благодаря этому гипертекст серьезно отличается от обычного текста. Обычные, линейные тексты имеют последовательную структуру и предусматривают (в христианском мире) их чтение слева направо и сверху вниз. Между тем, использование гипертекста позволяет фиксировать отдельные мысли, факты, а затем связывать их друг с другом, двигаясь в любых направлениях, определяемых ассоциативными связями. В результате образуется нелинейный текст.

В гипертекстовых системах информация напоминает текст энциклопедии, и доступ к любому выделенному фрагменту текста осуществляете произвольно по ссылке. Организация информации в гипертекстовой форме используется при создании справочных пособий, словарей, контекстной помощи (He1p) в прикладных программах.

Расширение концепции гипертекста на графическую и звуковую информацию приводит к понятию *гипермедиа*. *Гипермедиа* – технология представления любых видов информации в виде

относительно небольших блоков, ассоциативно связанных друг с другом.

Гиперсреда, именуемая также гипермедиа, является моделью взаимодействия блоков данных по ассоциации – совокупности различных свойств, характеристик, параметров. Этими блоками являются тексты, изображения, видеофильмы, файлы, программы, фрагменты звука.

В гиперсреде информация разбита на относительно небольшие блоки, представляемые вершинами графа. При работе с гиперсредой абонентская система содержимое каждой вершины высвечивает на экран монитора. Вершины соединены ребрами (связями), активизируются с помощью клавиатуры, светового пера либо мыши. Пользователь при работе с гиперсредой осуществляет навигацию и переходит от одной вершины к другой, двигаясь по сети области знаний. Гиперсреда может просматриваться и без обращения к содержимому блоков информации. Ее ПО управляет переходами по ребрам и формирует необходимые документы.

Важной характеристикой гиперсреды является представляемое ею информационное пространство. Перечень блоков информации, установление связей между ними зависят от разработчиков, определяющих гиперсреду. В результате образуются специальные Базы Данных (БД). Некоторые из них сопровождаются схемой маршрутов, в соответствии с которыми осуществляются переходы от одного блока к другому. В ряде баз пользователям разрешается добавлять новые блоки информации и связи. Расширяется рынок прикладных программ, созданных для работы в гиперсреде, он, в первую очередь, предоставляет:

- энциклопедии;
- учебники;
- каталоги товаров и изделий;
- справочники и справочные пособия;
- средства коллективной работы в локальных сетях;
- системы искусственного интеллекта.

Особое значение гиперсреда получает при использовании аудиовидеосистем, осуществляющих обработку данных всех возможных видов. Гиперсреда широко используется в обучающих системах и дистанционном обучении. Если в блоках информации гиперсреды располагаются в основном, тексты, то ее называют гипертекстом. Между тем следует иметь ввиду, что в гипертексте используются текстовые файлы, а в гиперсреде – графические файлы

или звуковые файлы. Благодаря этому в гиперсреде можно ссылаться не только на изображения, звук, но и на их детали.

Идеи гипермедиа получили распространение в сетевых технологиях, в частности в Интернет-технологиях. Технология WWW (World Wide Web) позволила структурировать громадные мировые информационные ресурсы посредством гипертекстовых ссылок. Появились программные средства, позволяющие создавать Web-странички. Здесь с помощью *языка разметки* в текстах страниц, в местах ассоциированных связей, делаются гипертекстовые *ссылки.* Они не видны на экране, но благодаря им компьютер находит следующую страницу, требуемую пользователем.

Создание технологии гипертекста и развитие службы глобального соединения привели к появлению нового вида языков. Задачей каждого из них является выделение в тексте объектов, составляющих гипертекст, и указание на их ассоциативные (тематические) связи. Кроме этого, язык разметки должен обеспечить стандартное форматирование каждого из указанных объектов.

Приведём несколько примеров.

Язык HTML представляет собой совокупность команд, которые вставляются в текст, подготовленный с помощью текстового редактора. Использование HTML дает возможность:

- описывать документы и их составляющие;
- указывать ассоциативные связи между документами;
- просматривать документы и находить в них необходимые сведения;
- вести обработку документов.

Язык HTML предоставляет разнообразные возможности создания гиперсреды. Вместе с этим он не удовлетворяет принципу «что вы увидели, то и получили»–WYSIWYG. Пользователь же часто хочет, чтобы все вспомогательные символы, не входящие в создаваемый им документ глобального соединения, были от него спрятаны. Это привело к появлению нового поколения редакторов просмотра, учитывающих принципы WYSIWYG. Развитие идей HTML привело к созданию языка моделирования виртуальной реальности – VRML (Virtual Realty Modeling Language). Как следует из названия, язык предназначен для создания виртуальной реальности, использующей гиперсреду, в том числе и документы HTML. Язык VRML (Virtual Realty Modeling Language) предназначен для описания трехмерных изображений и оперирует объектами, описывающими геометрические фигуры и их

расположение в пространстве. Его назначение – создание виртуальных миров во всемирной паутине.

Стандартный обобщенный язык разметки Standard Generalized Markup Language (SGML) – язык разметки логического описания структуры документов. SGML оперирует с разнообразными объектами: текстами, изображениями, фрагментами звука-и предоставляет пользователям общие языковые средства описания этих компонентов документов. Благодаря этому стандартизируются форма и логическая структура представления документов, их редактирование, распространение и размещение в Базах Данных (БД).

SGML, являясь общим языковым средством, используется не только для обработки документов, но и для записи музыкальных произведений. «Стандартный язык описания музыки» SMDL позволяет производить разметку музыкальных произведений, вводить необходимые обозначения, музыкальные характеристики и сопутствующую текстовую информацию. Кроме этого, SGML является эффективным средством синхронизации во времени музыкальной информации. С развитием гиперсред стали развиваться механизмы поиска нужной информации в лабиринте информационных потоков. *Поисковые системы* (синонимы – поисковые сайты, поисковые машины) предназначены для поиска информации в Интернете.

Специальная программа – «поисковый робот» или, как её иногда называют, «*паук*» – непрерывно ищет в Интернете новые сайты и проверяет старые. Новые адреса заносятся в базу данных поисковой системы, а устаревшие удаляются из базы. Поисковый робот не просто ищет новые сайты – он анализирует содержание их страниц. Особым образом он описывает просмотренные страницы и заносит в базу данных поисковой системы информацию о словах, содержащихся на странице, а также информацию о ключевых словах и краткое описание (если они присутствуют в коде html страницы).

Когда вы что-либо ищите с помощью поисковой системы, поиск по вашему запросу ведётся не непосредственно в Интернете, а в базе данных поисковой системы. В силу особенностей работы поисковых роботов у разных поисковых систем базы данных не совпадают. Поэтому, если вам не удастся найти информацию с помощью одной поисковой системы, имеет смысл воспользоваться другой.

Результат поиска отображается в виде списка адресов страниц сайта, отсортированных по релевантности, т. е. по степени соответствия найденной страницы поисковому запросу. Страницы, в большей степени отвечающие запросу, находятся в начале списка, в меньшей степени – в конце. Поиск этих страниц осуществляется по

«унифицированным указателям ресурсов» URL. Страницы оформляются таким образом, что в их текстах даются ссылки на любые другие документы, входящие в тот же либо в другой граф, содержащиеся как в том же, так и в других серверах WWW. При этом безразлично, где находится предлагаемый далее для просмотра документ, в той же комнате либо на другом континенте.

Русскоязычные поисковые системы. Они, в целом, не содержат информацию об иностранных сайтах, поэтому для поиска информации на иностранных языках лучше использовать англоязычные поисковые системы.

Наиболее популярные:

- Яндекс – www.yandex.ru;
- Рамблер – www.rambler.ru;
- Mail – www.mail.ru.

Англоязычные поисковые системы

Применение англоязычных поисковых систем существенно расширяет возможности поиска. Естественно, если вы знаете английский язык. К сожалению, эти системы некорректно отображают результаты поиска на русскоязычных сайтах (рис. 5.7).

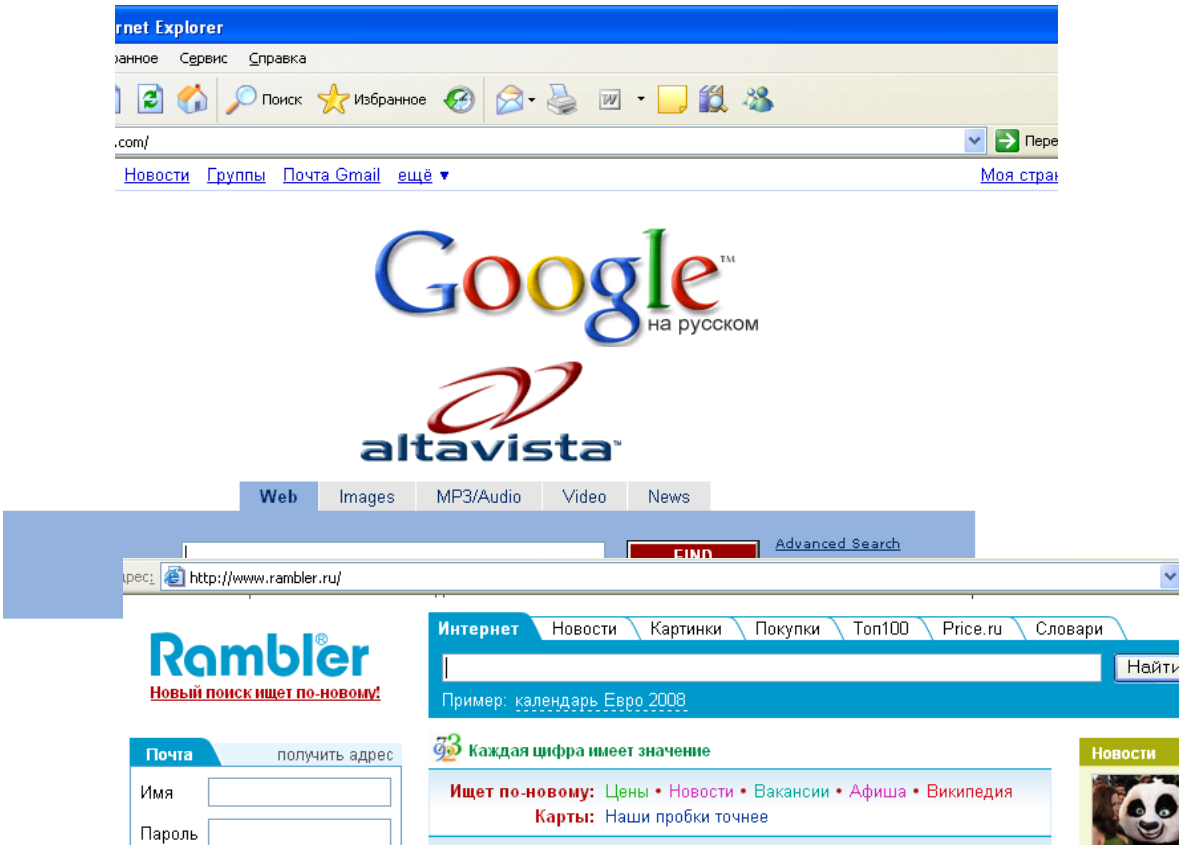


Рис. 5.7. Примеры окон поисковых систем

Наиболее популярные из них:

- Google – www.google.com;
- Yahoo! – www.yahoo.com;
- AltaVista – www.altavista.com.

Если вас интересуют другие поисковые системы, вы можете скачать коллекцию избранных страниц «Поисковые сайты: наиболее популярные и известные поисковые сайты русского интернета» на сайте www.shirs.org по адресу http://www.chirs.org/collect/. Сегодня созданы программы перевода текстов страниц WWW с одного языка на другой. Используя эти программы, можно нажатием клавиши перевести содержимое страницы с английского языка на французский, испанский, немецкий или наоборот – с указанных языков на английский.

5.2.2. Мультисреды

Мультисреда (multimedia – многосредовость) – технология комплексного представления любых типов данных. Мультисреда, именуемая также *мультимедиа*, обеспечивает совместную обработку изображений, обработку речи и обработку документов. Это позволяет выдавать на экран изображение с текстом и звуковым сопровождением. Мультисреда является подмножеством гиперсреды, объединяющим элементы первой с гипертекстом. Оно представляет собой объединение нескольких видов информации под единым интерактивным программным управлением.

Основными направлениями применения мультимедиа являются:

1. Образование, обучение.
2. Реклама, презентация.
3. Энциклопедии, справочники.
4. Развлечения.

Средства мультимедиа – это комплекс аппаратных и программных средств, позволяющих человеку общаться с компьютером, используя самые разные, естественные для себя среды: звук, видео, графику, тексты, анимацию и др. Таким образом, *мультимедиа* – это взаимодействие визуальных и аудиоэффектов под управлением интерактивного программного обеспечения.

Появление и широкое распространение внешних носителей (CD, DVD и др.) сделало эффективным использование мультимедиа в рекламной и информационной службе, сетевых телекоммуникационных технологиях, обучении. *Мультимедиа электронные издания,* мультимедийные игровые и обучающие системы начинают вытеснять традиционные «бумажные библиотеки». Сегодня в библиотеках CD и

DVD-ROM можно «гулять» с помощью «электронного путеводителя» по музеям, городам, континентам, планетам и т.д. Важным направлением мультисреды является создание обучающих систем. Это связано с тем, что при активной работе в мультисреде пользователь запоминает до 75 % используемой информации, между тем как из услышанной информации он запоминает лишь 25 %. В последнее время мультимедиа-компоненты все чаще используются в Web-технологиях при разработке различных сайтов. Бурно развивается направление *мультимедиа в телекоммуникациях*. Многие крупные фирмы активно ведут разработки коммуникаций на основе волоконнооптических сетей стандарта ISDN. Канал в такой сети имеет пропускную способность 64 Кбит/сек, и пользователю может быть предоставлено одновременно несколько каналов. Спектр предлагаемых применений мультимедиа весьма широк – от просмотра заказной телепередачи до выбора нужной книги и участия в мультимедиа-конференции. Такие разработки уже имеют общее *название Information Highway*, и предполагается их объединение в рамках государственных программ.

Музыкальные редакторы

Существуют программы, позволяющие самому писать музыку, редактировать уже написанные мелодии. Программы-микшеры позволяют по ходу дела регулировать громкость и стерео баланс по каждому звуковому каналу, несколько дорожек позволяют производить наложение одной мелодии на другую. Например, редактор Scream Tracker, плейеры (Jet Audio).

Среды подготовки презентаций

Вершиной эволюции коммуникационных инструментов являются мультимедийные презентации. Это интегрированный продукт, который состоит из текста, графики, анимации, видео и аудио (голоса или музыки). Несмотря на то, что мультимедийные презентации намного мощнее видео, их изготовление и использование стоят в несколько раз дешевле.

Ко всем изобразительным возможностям видео у мультимедийных презентаций прибавляются два мощных инструмента – интерактивность и возможность простого обновления информации. Не нужно переснимать фильм или привлекать специалистов, чтобы обновить материал – это можно сделать и самому. Конечно, если об этом заранее позаботился изготовитель презентации. Можно добавить, что обычно мультимедийная презентация, создаваясь для решения одной задачи, решает сразу несколько. Например, виртуальный тур не просто передает много разной информации, но и обязательно производит впечатление и

привлекает внимание. Учитывая современные мировые тенденции к дискредитированию и падению эффективности массовой рекламы, информационной перенасыщенности и ужесточению конкуренции за ресурсы и потребителей, такие интересные и точно направленные инструменты, как мультимедийные презентации, по нашему мнению и отзывам клиентов, являются наилучшим способом передачи информации и эмоций и установления контакта с целевой аудиторией. Ведь чтобы быть эффективной, презентация должна быть настоящим шедевром дизайна, маркетинга, PR и менеджмента. Мультимедийная презентация может быть адаптирована для использования в качестве Интернет-сайта или его части. Оптимизация мультимедиа-презентации производится также для рассылки ее по электронной почте потенциальным клиентам, партнерам и контрагентам. Для сравнения возможностей различных мультимедийных редакторов возьмем два различных редактора: *PowerPoint (рис.5.8) и Anark Studio*.

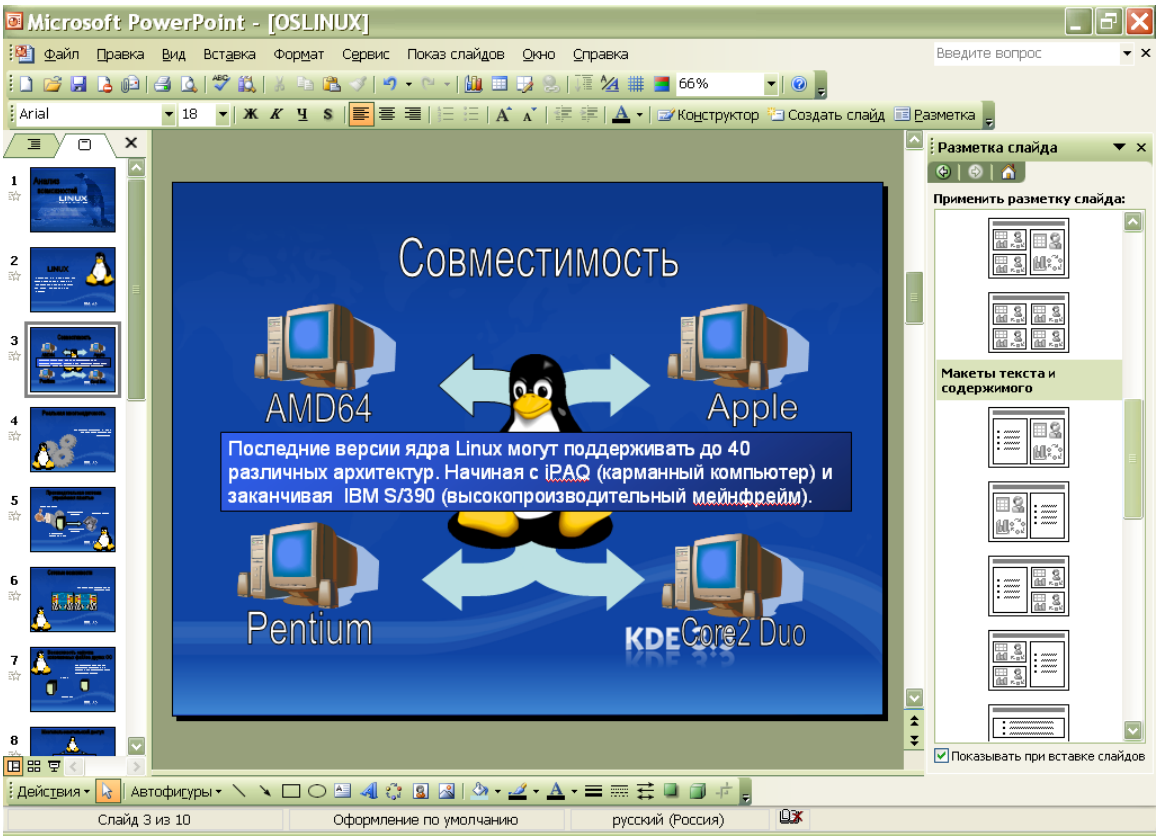


Рис. 5.8. Окно PowerPoint 2000

Создавая *PowerPoint 2000*, корпорация Microsoft стремилась дать пользователям самые простые инструменты для создания презентаций и их проведения при личной встрече, в удаленных аудиториях или по Интернету. Новые возможности PowerPoint 2000 представляют собой

прямой результат обширных исследований и анализа пожеланий пользователей. Как обнаружили разработчики, в большинстве случаев пользователи этого приложения сталкиваются с необходимостью срочно выполнить задание. Они работают с программой около шести часов в неделю, т.е. гораздо меньше, чем со многими другими приложениями комплекта Office. Когда приходит время готовить презентацию, пользователи должны быть в состоянии легко и быстро изучить продукт и добиться желаемых результатов.

Anark Studio – среда разработки интерактивных мультимедийных приложений. Anark Studio – это мультимедийный пакет, предназначенный для интеграции 3D-графиков, видео, аудио и текстов в единые проекты и дальнейшего их переноса на CD, DVD или другие носители. Из возможностей решения отметим лишь создание кросс–платформенных приложений (при использование дополнительных Plug-in'ов), поддержку JavaScript, создание композиции в режиме реального времени и многое другое.

Power Point получил общее признание за свою простоту в освоении и использовании. Как было сказано ранее, данное приложение позволяет быстро создать довольно привлекательную презентацию, но данная продукция будет обладать малой интерактивностью.

Интерактивность – главная особенность Anark Studio. С помощью этого приложения можно создавать реально интерактивные презентации (рис. 5.9).

Anark Studio – это уже профессиональная среда для создания презентации, требующая больше, чем знания обычного пользователя. Например для создания интересных физических явлений (гравитация, невесомость и прочее) требуется знание языков программирования (JavaScript), а для создания сложных 3D-моделей необходимо знание и умение использования различных 3D-редакторов, данные возможности реализованы с помощью аппаратной поддержки OpenGL (архитектура приложения позволяет взаимодействовать с приложениями 3D-моделирования, такими, как Maya, LightWave, 3DS Max и Cinema 4D).

Как и Point, Anark Studio имеет сходное строение в интерфейсе, но за счет использования элементов 3D-графики ее внутреннее устройство намного сложнее. При помощи встроенных элементов можно самостоятельно, без применения каких-либо других редакторов, создать яркую и интересную презентацию. Для подключения дополнительных возможностей (таких, как импорт или экспорт, дополнительные элементы рендеринга или же новые элементы управления) используются различные Plug-in'ны.

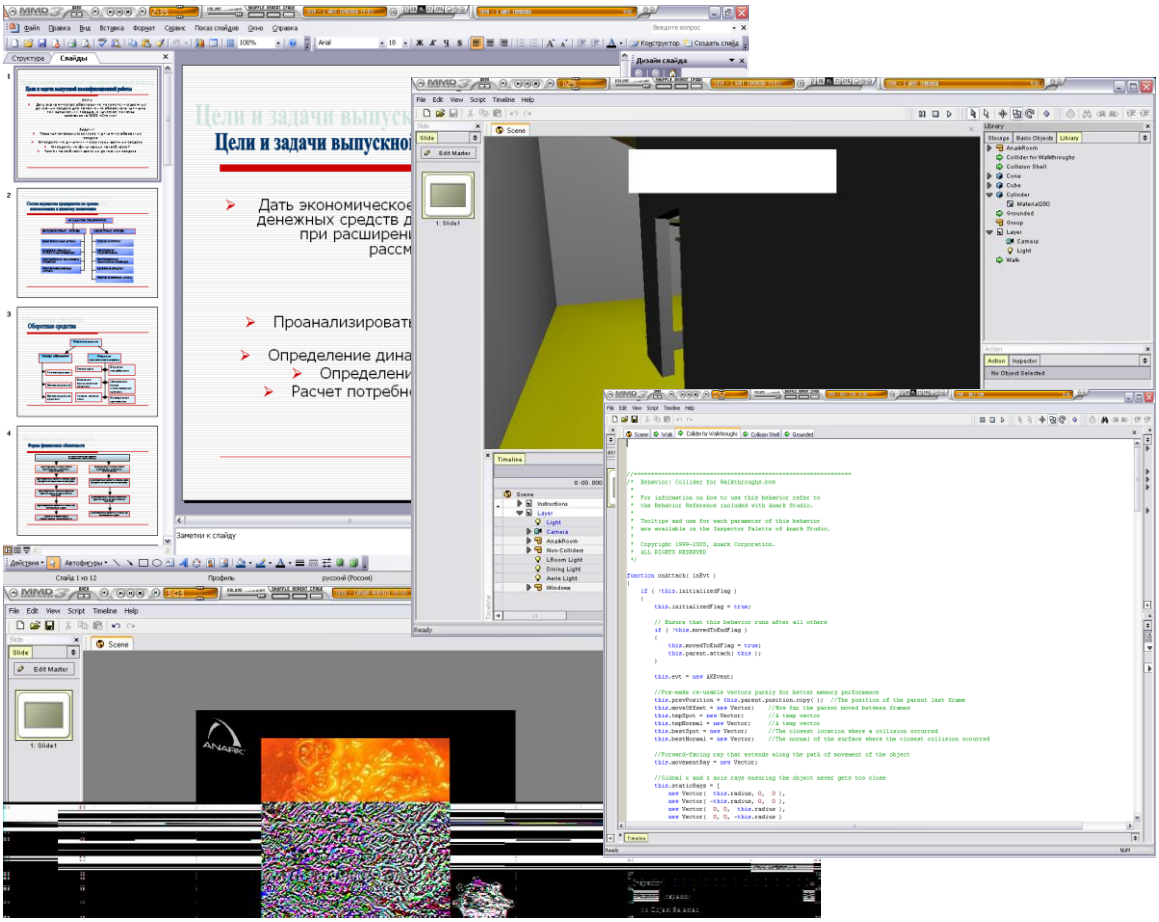


Рис. 5.9. *Интерфейс среды разработки интерактивных мультимедийных приложений Anark Studio*

Также в Anark Studio используется система скриптов основанных на семантике Паскль и C++. С помощью данной системы можно самостоятельно создавать новые эффекты или элементы управления (в стандартных примерах, приложенных вместе с программой, можно найти множество стандартных скриптов (скрипт на упрогое столкновение тел, на создание реалистичной гравитации и прочее).

Скрипты могут быть использованы для описания работы некоторых рабочих элементов, таких, например, как кнопки для переключения слайдов. Применяя к ним написанные скрипты, можно создать интерактивное управление.

Как пример, можно создать скрипт, который будет не просто анимировать слайд (как правило, это скрипты рендеринга сцены), а производить вращение плоскостей (на которых расположены слайды) при нажатии кнопки переключения слайдов. Но для полноценного использования данной программы знание основных элементов управления и языков программирования недостаточно. Для полного раскрытия возможностей программы необходимо знание других

программ, работающих со звуком и графикой.

Существует множество организаций, занимающихся разработкой интерактивных презентаций, открыток и рекламы, которые используют Anark и ей подобное программное обеспечение. Спрос на мультимедийные презентации на рынке обуславливается тем, что это не только очень удобный и информативный способ представления информации докладчика, а новый метод «оживления» преподносимой информации, способной привлечь внимание как простых людей, так и инвесторов, и новых партнеров.

Мультимедийная презентация – универсальный маркетинговый инструмент. Разработанная однажды мультимедийная презентация может иметь более десятка различных применений. Это делает мультимедийную презентацию экономически привлекательным средством для продвижения и информирования о товарах, услугах и брэндах

Еще больший эффект от использования мультимедийной презентации на выставке достигается путем ее одновременного использования в качестве раздаточного материала на одном из носителей. Электронная визитная карточка (CD-визитка), мини-диск или USB-flash drive с мультимедийной презентацией будут ярко выделяться на фоне однообразия полиграфических материалов. Для менеджеров по продажам, которые проводят переговоры с потенциальными клиентами и партнерами, мультимедийная презентация является незаменимым инструментом продаж. Деловые переговоры с использованием мультимедийной презентации в качестве наглядного презентационного материала проходят намного живее и динамичнее, внимание собеседников фокусируется на основной теме разговора. Мультимедийные презентации – еще один шаг в развитии сферы применения информационных технологий в экономике.

Виртуальная реальность

Термин «виртуальная реальность» можно определить как новую информационную технологию, позволяющую пользователю в реальном времени находиться и перемещаться в иллюзорном трехмерном пространстве (вверх, вниз, вправо, влево, вглубь, наружу).

Можно считать виртуальную реальность дальнейшим развитием технологии мультимедиа. Впервые технология виртуальной реальности (ВР) была использована для обучения военных летчиков.

Виртуальная реальность позволяет создать для медицинских работников иллюзию реально проводимой хирургической операции. Архитектор может рассмотреть интерьер и внешний вид

спроектированного им здания. Конструктор (или инженер) может создавать трехмерное изображение объекта и испытывать созданную модель, находясь внутри нее.

Для демонстрации мод используют виртуальные модели, которые заимствуют лучшие черты известных манекенщиц.

Виртуальный музей позволяет посетителям увидеть любой экспонат коллекции в трехмерном виде и с разных сторон. Имитация танкового боя дает экономию средств военным ведомствам в результате отказа от проведения реальных маневров.

ВР применяется при тренировке летчиков, космонавтов и спортсменов. Разработан специальный тренажер, имитирующий спуск на спортивных санях. Спортсмены могут, не выходя из здания, опробовать любые трассы и пройти подготовку к соревнованиям.

Виртуальные банки и виртуальные магазины входят в обыденную жизнь людей. При этом действия в виртуальном универсаме напоминают обычную покупку в традиционном магазине. Можно рассматривать и выбирать товары, переходя от одной полки к другой. Затем нужно расплатиться в виртуальной кассе с помощью кредитной карточки, а товар будет доставлен на дом с помощью реальных транспортных средств.

ВР является одним из захватывающих средств развлечений.

Например, во время моделирования полета в космическом корабле происходят столкновения с астероидами и нападения инопланетян. Ускорение и замедление корабля имитируются перемещениями кресла. Полет сопровождается звуковыми и световыми эффектами.

Имитация пикирования к земле или мертвая петля в маленьком спортивном самолете заставляют весь организм напрячься. Человек теряет связь с действительностью и чувствует себя парящим в кабине самолета над простирающимися до горизонта холмами и равнинами. Принцип имитации ускорений (перегрузок и силы тяжести) основывается на использовании управляемой центрифуги с независимым вращением по трем осям.

С помощью виртуального учителя есть возможность изучать боевые искусства и танцы.

Можно ожидать, что виртуальная реальность найдет широкое применение при изучении стереометрии, черчения, при решении конструкторских задач.

Технология ВР формирует трехмерное изображение, стереофонический и квадрофонический звук, тактильные ощущения, воздействует на чувства равновесия. Напомним, что *тактильные ощущения* – это ощущения прикосновения, осязания.

Программные продукты и аппаратные средства виртуальной реальности имитируют реальную действительность с такой степенью достоверности, что можно «потрогать» объекты, находящиеся в этом призрачном мире, и они соответствующим образом будут реагировать на прикосновение. Процедуру вхождения в виртуальный мир часто называют погружением. Конечно, это не следует понимать буквально, как это показано на рисунке слева.

Для взаимодействия с виртуальной реальностью применяются специальные устройства ввода-вывода: шлемы-дисплеи, манипуляторы, информационные перчатки и информационный костюм (рис. 6.10).

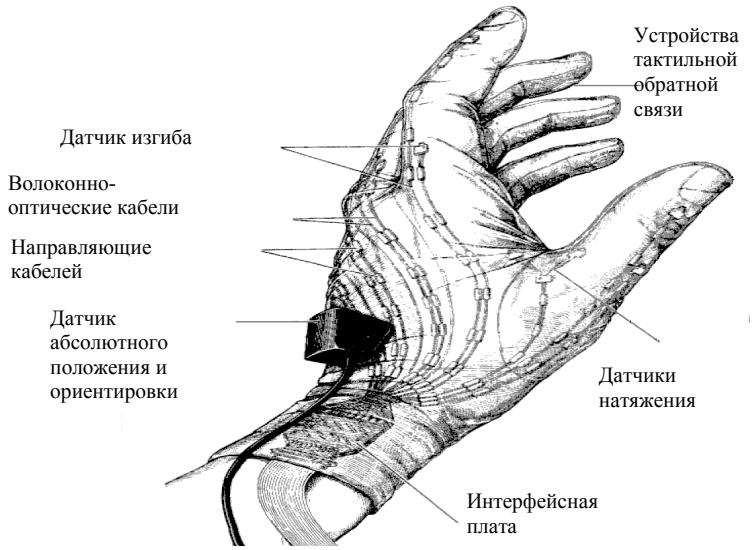


Рис. 5.10. Информационная перчатка

Рука пользователя, одетая в информационную перчатку, может быть спроецирована (перенесена, погружена) в трехмерную компьютерную среду. Манипулируя информационной перчаткой, пользователь в состоянии взаимодействовать с виртуальным миром, управляя объектами. Для описания виртуальной реальности создан специальный язык – VRML (**V**irtual **R**eality **M**odelling **L**anguage), на базе которого различные фирмы создают программные средства.

Игры и тренажеры

Игровые программы позволяют не только развлекаться, но и получать некоторые новые полезные знания.

Компьютерные деловые игры (КДИ). Применение различных моделей для экономического анализа, конечно, не является новым явлением, однако только в последние годы широко распространились технические возможности и возник массовый спрос на построение сложных, многосторонних моделей и применение этих моделей для проведения учебных занятий, демонстрации экономических явлений, организации соревнований между слушателями курсов и многого другого. Уже сегодня учебные курсы могут быть дополнены работой с компьютерными тренажерами.

К одним из наиболее эффективных инструментов обучения дисциплинам экономического профиля следует отнести компьютерные деловые игры (КДИ), которые ориентированы на формирование практических навыков управления крупной производственной компанией в рыночной ситуации. Основная цель КДИ – дать студентам возможность приобрести опыт управления фирмой как целостным субъектом рынка и развить их ролевые навыки принятия комплексных управленческих решений.

Вместе с развитием возможностей вычислительной техники КДИ прошли несколько этапов, различающихся технологией применения от первых автоматизированных КДИ, которые назывались машинными, до сегодняшних КДИ, в которых существует возможность их проведения в сетевых классах и через глобальную сеть Интернет.

КДИ можно разделить на два типа: коллективные и индивидуальные.

В коллективных КДИ участвуют несколько игроков или групп, выполняющих роли лиц, принимающих решения.

Коллективные КДИ более приближены к реальности. Большое преимущество индивидуальной КДИ – невысокие требования к квалификации преподавателя. Игра может проводиться вовсе без преподавателя, что весьма важно для дистанционного обучения и для самостоятельной работы обучаемого.

Преимущества компьютерных деловых игр перед обычными деловыми играми заключаются в следующем:

- снижении организационных затрат, т. к. снижается количество персонала, занимающегося непосредственно организацией игры;
- сокращении времени на проведение деловой игры;

- возможности быстрого и наглядного отображения результатов игры в любой момент времени;
- возможности получения справок, описаний и определений в ходе игры;
- проведение автоматизированного контроля знаний и многое другое.

В целом же применение КДИ повышает эффективность учебного процесса: материал изучается и усваивается за меньшее учебное время (как показано на рис. 5.11).

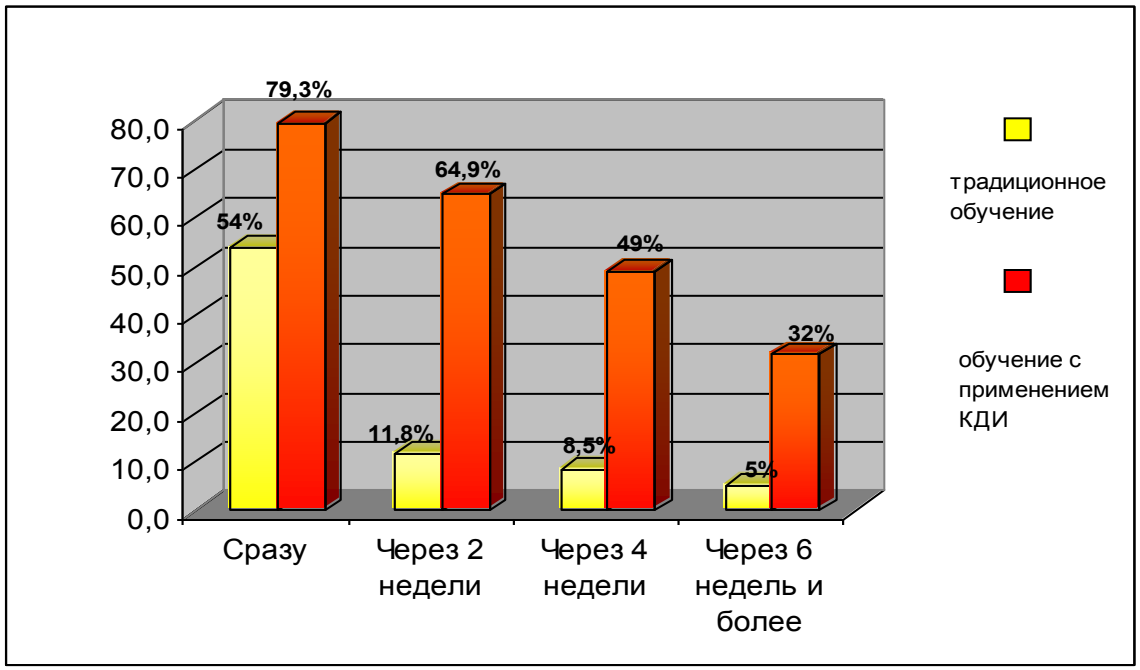


Рис. 5.11. Динамика роста интереса студентов к изучению экономических дисциплин с применением КДИ и полученных знаний, умений и навыков для дальнейшей профессиональной деятельности.

Для дистанционного обучения (ДО) удобнее всего применять КДИ индивидуального типа. В настоящее время таких разработок еще совсем немного. Один из коллективов, целенаправленно работающий над созданием комплекса индивидуальных КДИ, – компания КОББИ (Компьютерное обучение бизнесу)

Главная цель разработчиков комплекса КОББИ – создание эффективных средств обучения для проведения практических занятий по основным экономическим дисциплинам.

На базе КОББИ-игр созданы компьютерные экономические практикумы.

На сегодняшний момент также большое распространение получили следующие игры.

Компьютерная деловая игра «Моделирование экономики и менеджмента (МЭМ)». Она служит базой для перехода к более серьезным и продвинутым компьютерным деловым играм.

Компьютерная деловая игра «БИЗНЕС-КУРС». Суть игры – управление предприятием в условиях конкуренции. При этом компьютер «ведет» бухгалтерский учет на предприятии, составляет финансовую, налоговую и управленческую отчетность, рассчитывает разнообразные финансовые показатели. Игра представлена в двух вариантах. Коллективный вариант игры «БИЗНЕС-КУРС» предназначен для проведения групповых занятий под руководством преподавателя. Индивидуальный вариант игры можно использовать при очном и дистанционном обучении.

«Президент» – компьютерная деловая игра, предназначенная для изучения вопросов государственного регулирования рыночной экономики.

КДИ «*Капитализм»* предлагает участнику большие возможности. В данной игре можно создать собственную легенду, стать новым Генри Фордом автомобильной промышленности; стать лидером на компьютерном рынке за счет продажи компьютеров, имеющих разработанные вами научно-технические секреты, и т. д.

Компьютерная деловая игра *«Никсдорф Дельта»*. В основе лежит имитационная модель предприятия, функционирующего в условиях рыночной конкуренции. Условия деятельности виртуального предприятия в рамках компьютерной модели, лежащей в основе деловой игры «Никсдорф Дельта», определяются набором взаимосвязанных параметров, позволяющих имитировать изменение как внутренней, так и внешней среды предприятия.

Также последние несколько лет активизировались работы по проведению деловых Интернет-игр – соревнований по управлению проводимых через сеть Интернет с одновременным участием большого количества игроков.

Участие в Интернет-играх стимулирует общение студентов разных учебных заведений между собой, а зачастую и общение студентов с практикующими менеджерами, участвующими в Интернет-игре.

Наиболее известными деловыми Интернет-играми по управлению производственным предприятием, проходящими в России, стали игры «Business Battle» и «Global Management Challenge». Обе игры проводятся Академией народного хозяйства при Правительстве РФ и

являются российским этапом всемирного соревнования по стратегическому управлению.

Global Management Challenge – крупнейшее в мире соревнование по стратегическому управлению компанией для действующих менеджеров и лучших студентов бизнес-школ и университетов. Команды менеджеров со всего мира получают в управление виртуальные компании и конкурируют друг с другом за лидерство на глобальных рынках в условиях, максимально приближенных к реальным.

5.2.3. Авторские системы

Разработчики создают специальные программные системы целевого назначения для специалистов в некоторой предметной области. Такие программы называют авторскими инструментальными системами. *Авторская система* представляет интегрированную среду с заданной интерфейсной оболочкой, которую пользователь может наполнить информационным содержанием своей предметной области.

Авторские системы предназначены для создания программных продуктов с высокой степенью взаимодействия с пользователем. Часто для разработки пользовательского интерфейса авторские системы предлагают специальный язык сценариев. Они позволяют создать конечный продукт, объединяющий все мультимедиа-компоненты единой управляющей программой. Его отличительной чертой является наличие общего интерфейса, позволяющего выбрать любой из мультимедиа-компонентов, запустить его на выполнение (прослушать звуковой файл или просмотреть видео), организовать поиск требуемого объекта и т.п.

Часто авторские системы ориентированы на авторов учебного материала, преподавателей и не требуют от пользователей знаний языков программирования. Создание КСО осуществляется путем манипулирования визуальными представлениями образующих его компонентов – кадров. Составляется структурно-функциональная схема курса, формируются кадры различного назначения, определяются их свойства, указываются связи элементов.

Такой подход существенно упрощает разработку, снижает требования к компьютерной квалификации исполнителей, а также способствует повышению производительности за счет использования настраиваемых шаблонов типовых элементов. Другой важной особенностью «авторской» ИС является то, что она необязательно содержит внутри себя все редакторы, необходимые для представления

информации, например, графический, звуковой, видео, но является интегрирующей средой, позволяющей включать в учебный курс и рисунки, и анимацию, и речь, и видео, созданные в профессиональных редакторах. Обязательно наличие развитых систем анализа ответов, контроля и диагностики знаний, сбора и обработки статистики. Таким образом, универсальные авторские системы отличаются:

- ориентацией на пользователя – автора КСО;
- визуализацией представления элементов КСО;
- многофункциональностью – сбалансированным представлению всех подсистем;
- широким спектром анализируемых ответов;
- интеграцией с другими приложениями.

Среди авторских систем мы рассмотрим только те, которые присутствуют на рынке уже много лет и поддерживаются разработчиками и сегодня.

Formula Graphics (www.formulagraphics.com/). Авторская система Formula Graphics фирмы Formula Software применяется для разработки интерактивных приложений мультимедиа.

Formula Graphics имеет программируемую двух- и трехмерную графику и используется также для разработки приложений с анимацией и игровых программ. Разработанные мультимедиа-приложения могут быть проиграны с гибкого диска, CD-ROM, непосредственно через Интернет или внедрены в Web-страницу (рис. 5.12).

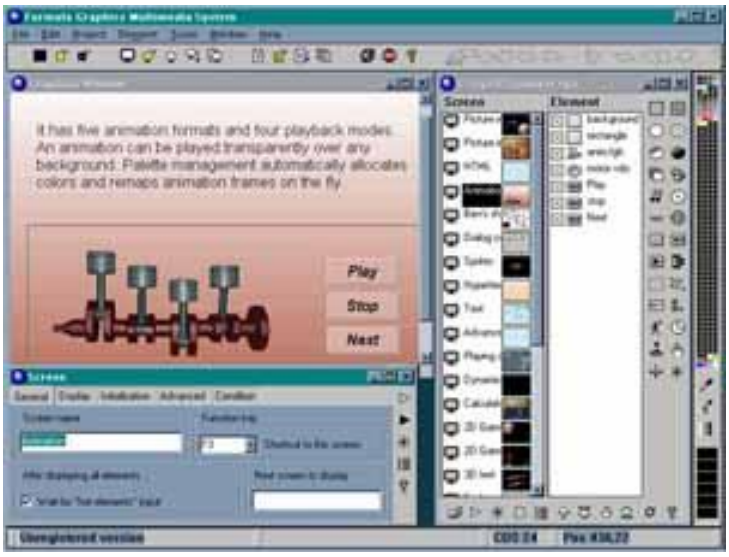


Рис. 5.12. Мультимедиа-проект в окне Formula Graphics

Она имеет простой и удобный в использовании графический интерфейс и не накладывает никаких ограничений на изображения,

звуки и анимации, которые могут быть объединены с ее помощью. Formula Graphics имеет мощный объектно-ориентированный язык, однако приложения можно разрабатывать и без применения программирования. Управляющие элементы на экране отображаются в виде гипертекста и графических гиперссылок.

GLpro (www.gmedia.com/glpro/). Авторская система Graphics Language for professionals (GLpro) фирмы IMS Communication. Это мощная и быстрая авторская система, использующая язык сценариев, для создания презентаций, демонстрационных дисков, руководств, компьютерных обучающих программ и других приложений. GLpro превосходит традиционные средства создания мультимедиа-приложений по быстродействию, гибкости и производительности приложений, созданных с его помощью. Однако она требует от разработчика знания программирования.

В ее состав входят различные инструменты, позволяющие сглаживать форму текстовых надписей в приложении, встраивать в приложение используемые шрифты, создать анимацию, оптимизировать палитру и обеспечивающих множество других возможностей.

HyperMethod (www.hypermethod.com/rus/index.html). Российская авторская система HyperMethod работает под Windows 95/98/NT. Она позволяет создавать самые разнообразные мультимедиа-приложения и по своим функциональным возможностям приближается к программе Macromedia Director. Поддерживает распространенные форматы звуковых и видеофайлов, а также возможность контролируемой покадровой анимации. Обеспечивает быстрое создание гипертекстовых приложений, а совместимость с HTML позволяет создавать приложения для Интернета. Имеет собственный язык сценариев. Новые возможности, добавленные в последней версии, делают ее привлекательной как для новичков, так и для профессионалов.

Authorware(www.macromedia.com/software/authorware/productinfo). Система фирмы Macromedia, позволяющая создавать интерактивные обучающие программы с элементами мультимедиа. Основана на изобразительном представлении потока данных и может быть использована профессиональными дизайнерами.

Структура приложения формируется на основе шаблонов, а также простым перемещением на линию потока данных значков различных файлов. Затем с помощью команд меню и различных мастеров формируется гипертекст. Допускает применение гиперссылок, полнотекстового поиска, имеет встроенные элементы управления для организации взаимодействия с приложением. Средства сжатия

позволяют оптимизировать приложение для доступа к нему через Интернет.

ToolBook (www.asymetrix.com/products/instructor.html). Система фирмы Asymetrix, состоящая из двух компонентов: ToolBook Assistant и ToolBook Instructor. Мультимедиа-приложение строится по принципу страниц книги с кнопками, полями данных и встроенными мультимедиа-элементами. Все управляющие элементы выбираются из каталога, включая возможность создания интерактивных вопросов и анимированных изображений.

Позволяет создавать эффективные обучающие и образовательные продукты, в том числе работающие дистанционно в среде Интернета. Легкий в использовании ToolBook Assistant содержит набор шаблонов, в которые добавляются тексты, рисунки, аудио- и видеофайлы, объединенные динамическим взаимодействием. Специальный мастер публикует курс в сети Интернет. Для профессиональных разработчиков и преподавателей предназначен ToolBook Instructor. Он позволяет создавать специализированные курсы со специфическими реакциями на действия пользователя. Этому способствует поддержка языка OpenScript, редактор Actions Editor и возможность применения DHTML. Разработанные мультимедиа-приложения могут распространяться на CD-ROM, непосредственно через Интернет или быть внедрены в Web-страницу.

IconAuthor (www.asymetrix.com/pr/new_iconauthor.html). Система фирмы AimTech. Позволяет создавать продукты для интерактивного обучения или изготавливать рекламные ролики. В качестве основы разрабатывается структурная схема из пиктограмм, каждая из которых обозначает определенное действие или функцию, выполняющихся в заданной последовательности. Требует знания принципов алгоритмизации. Приложения, созданные с помощью IconAuthor, могут взаимодействовать в Интернете с ToolBook Librarian.

Multimedia Builder (www.mediachance.com/). Это условно-бесплатная авторская система для создания мультимедиа-приложений, позволяющая построить полноценные Windows-приложения, содержащие графику, анимацию, музыкальное сопровождение (в том числе в формате MP3). Программа имеет объектно-ориентированный интерфейс и позволяет использовать анимированные GIF-файлы, управляя при этом запуском анимации, ее остановкой или запуская GIF-файлы на бесконечное проигрывание.

Первоначально в окне 400x300 точек строится главная форма, в которую можно вставить обычные кнопки с привязанными к ним командами, либо графические кнопки с тремя состояниями и

прозрачными участками любой формы. В качестве объектов можно вставить любые картинки, анимированные файлы GIF, AVI, WAV, MP3. Создав первую форму, к ней можно добавить последующие и вставить кнопки перехода между ними. В результате будет создано многоэкранное приложение. При этом Multimedia Builder создает либо exe-файл, либо файл в собственном формате, для выполнения которого используется небольшой проигрыватель. Конечный файл *autorun.exe* будет сжат и оптимизирован для запуска программ и файлов по относительным путям, например**,** .../audio/mysong.mp3**.** Применение сжатия данных позволяет создавать очень компактные программы.

*WebCompiler (*www.rey.ee/webcompiler/rus/index.html). WebCompiler позволяет Вам создать один исполняемый файл из набора HTML файлов и существующих картинок Web-узла. Встроенная поисковая система позволяет находить информацию в пределах скомпилированной презентации. Исходный текст HTML может быть закрыт от просмотра. Группы страниц можно защитить паролями, что позволяет делать информацию доступной только зарегистрированным пользователям. Встроенный обработчик форм позволяет создать презентации, в которых будут запрашиваться данные от пользователей, каталоги и т.д. Для работы требует установленный браузер Internet Explorer 4.0 или выше. Исполняемый файл, полученный в WebCompiler, содержит все элементы интерфейса браузера.

Hyper Maker HTML (http://bersoft.com/hmhtml/). Эта условно-бесплатная программа позволяет преобразовать Web-сайт в приложение, распространяемое на дискетах или CD-ROM, а также предназначена для быстрого создания гипертекстовых и мультимедиа-публикаций. Для разработки мультимедиа-приложения также потребуется HTML-редактор, в котором можно будет дополнить содержимое Web-сайта управляющими элементами. На страницах приложения возможно воспроизведение анимированных GIF-файлов, звука в форматах MIDI, WAV и MP3 и видео в формате AVI. Можно защитить приложение паролем от редактирования, запретить печать или копирование отдельных фрагментов в буфер обмена. Поддерживает полнотекстовый поиск на нескольких языках. После компиляции приложение не требует браузера для просмотра.

LERSUS easyContent. LERSUS – программный продукт, позволяющий создавать интерактивные учебные материалы, курсы компьютерной профессиональной подготовки (*CBT*), курсы дистанционного обучения (*WBT*) в соответствии с существующими *e-Learning* стандартами. С помощью LERSUS Вы легко создадите наглядные и интерактивные учебные материалы для дистанционного

обучения через интерент, в локальной сети или с помощью сервера дистанционного обучения (*LMS*). LERSUS позволяет создавать тесты для проверки знаний и включать их в интерактивные материалы.

Авторская среда «ДЕЛЬФИН» для проектирования учебных курсов на базе мультимедиа технологий. ИС «ДЕЛЬФИН» разработана в Центре новых информационных технологий Московского энергетического института (ЦНИТ МЭИ).

Авторская система «ДЕЛЬФИН» позволяет интегрировать видео-лингво-гипермедиа-компьютерные- и интернет-компоненты в единую обучающую среду. Является эффективным средством управления процессом обучения.

Особенностями «Дельфина» являются:

- детальное описание дидактической цели каждого элемента курса;
- большие возможности анализа разнообразных ответов обучаемого;
- возможность изменения хода обучения в зависимости от результатов.

Полная версия ИС включает модуль для создания ситуационных тренажеров.

«Универсальный Редактор Обучающих Курсов» («УРОК»). ИС разработана НПФ ДиСофт (г. Москва). Система «УРОК» представляет собой программный комплекс, обеспечивающий создание компьютерных обучающих и контролирующих курсов в различных предметных областях, проведение тестов и контрольных заданий. «УРОК» может быть также использован для создания презентационных, демонстрационных комплексов и проектов. В «УРОКе» существует возможность подключения и запуска обучающих программ, созданных в других инструментальных средах. Существует возможность формирования интегральной оценки знаний с учетом результатов, полученных обучаемым в программах, являющихся внешними по отношению к «УРОКу».

Macromedia Authorware. Разработано фирмой Macromedia, США. Macromedia Authorware – визуальная среда разработки, предназначенная для создания интерактивных мультимедийных обучающих программ. Пакет Authorware предусматривает совместное использование различных форм подачи материала: текста, рисунков, видео и звукового сопровождения. Конечный продукт, созданный с помощью Authorware, представляет собой независимое приложение, которое может быть либо записано на компакт-диск, либо опубликовано в Интернете. Среда разработки Macromedia Authorware хорошо интегрирована с другими продуктами фирмы Macromedia.

5.2.4. Экспертные системы

Экспертная система – это программа, которая ведет себя подобно эксперту в некоторой узкой прикладной области, использующая знания одного или нескольких экспертов, представленные в некотором формальном виде, а также логику принятия решения человеком-экспертом в трудно- или неформализуемых задачах. Экспертные системы призваны решать задачи с неопределенностью и неполными исходными данными, требующие для своего решения экспертных знаний. Экспертные системы способны в сложной ситуации (при недостатке времени, информации или опыта) дать квалифицированную консультацию (совет, подсказку), помогающую специалисту или менеджеру принять обоснованное решение. Основная идея этих систем состоит в использовании знаний и опыта специалистов высокой квалификации в данной предметной области специалистами менее высокой квалификации в той же предметной области при решении возникающих перед ними проблем.

Кроме того, эти системы должны уметь объяснять свое поведение и свое решение. Принципиальным отличием экспертных систем от других программ является их адаптивность, т.е. изменчивость в процессе самообучения.

Принято выделять в экспертных системах три основных модуля:

- модуль базы знаний;
- модуль логического вывода;
- интерфейс с пользователем.

База знаний содержит формальное описание знаний экспертов, представленное в виде набора фактов и правил. Механизм вывода или решатель – это блок, представляющий собой программу, реализующую прямую или обратную цепочку рассуждений в качестве общей стратегии построения вывода. С помощью интеллектуального интерфейса экспертная система задает вопросы пользователю и отображает сделанные выводы, представляя их обычно в символьном виде.

Обычно экспертные системы создаются в узких предметных областях. Первые модели были созданы в середине 70-х годов: система MYCIN использовалась в медицине для диагностики заболеваний, DENDRAL – в разведке месторождений полезных ископаемых для анализа химического состава почв.

Сегодня экспертные системы разработаны в самых разнообразных областях. Приведём несколько примеров.

Экспертная система выбора хостинга. В настоящее время на рынке существует достаточно большое количество хостинг-компаний, предлагающих свои услуги. Многообразие тарифных планов и различных вариантов хостинга может запутать даже достаточно опытного пользователя. Как не потеряться в этом объеме информации? Разработчики экспертной системы предлагают воспользоваться системой подбора хостинга, автоматизированный подбор которого доступен любому уровню пользователей и позволяет подобрать хостинг под любые идеи и начинания. Экспертная система подскажет, где удобней и быстрее вы сможете получить желанное место под домашнюю страничку с наименьшими затратами.

Экспертная сеть. Экспертные организации, построенные по принципу сети и, как правило, объединенные с помощью Internet. В США в последние годы сформировались сети экспертных организаций, гарантирующие, что обратившийся к ним клиент (разумеется, с запросом по профилю организации) получит возможность связаться с экспертом в требуемой области максимум в течение дня. Ассоциации объединяют сотни экспертов (на уровне, соотносимом с российскими федеральными округами), а ProfNet (называемый «дедушкой американских экспертных директорий») – даже тысячи, около 5000 экспертов и более 14000 информационных центров в университетах и корпорациях. Для ассоциаций, центров и сетей типичным является не только непосредственно решение проблемы клиента, но и тщательная проработка методологических и теоретических основ своей деятельности. Вот так, например, формулируется миссия Центра оценки (в нашем понимании – скорее экспертизы) штата Индиана (Indiana Center for Evaluation): «Обеспечить клиентов информацией, которая помогает им принимать эффективные программные и поведенческие (политические) решения».

Программа-консультант «Приемы журналистики & PR» (на основе более 12.500 текстов Мастеров журналистики, литературы и рекламы). Основные классы решаемых задач:

- избежать нежелательной автору ассоциации у читателя;
- подтвердить/ознакомить читателя с деталями ему ранее известного;
- передать читателю эмоции, страхи, ощущения...
- сделать неизвестное читателю – известным, своим;
- получить «эффект новизны» при хорошо известном читателю предмете описания;
- привлечь внимание читателя к фрагменту текста;

- побудить читателя осознать свои стереотипы, создать предпосылки к изменению мнения;
- спародировать известное читателю.

Программа-консультант «EXPO: 1001 Рекламоноситель»

Основные классы решаемых задач:

- синтез PR-акций и рекламных кампаний;
- поиск идей для создания выставочных стендов и проведения презентаций;
- поиск рекламоносителей (особенно новых, нестандартных) и их поставщиков;
- анализ полученной системы решений.

Программа-консультант «HeadLiner/Заголовщик» (на основе анализа более 6000 произведений)

Программа предназначена для создания:

- заголовков, кратких анонсов;
- запоминающихся образов, метафор и фраз;
- слоганов, девизов и эхо-фраз;
- текстов для наружной рекламы и баннеров;
- подписей под иллюстрациями;
- эпиграфов и афоризмов.

Учитывая, что слоганы и заголовки часто являются видоизменением известных выражений, программа также содержит 10 тематических баз данных:

- пословицы и поговорки русского языка;
- крылатые фразы из мульт- и кинофильмов;
- изречения из Ветхого и Нового завета;
- крылатые выражения российских политиков XIX–XX веков;
- современный жаргон;
- афоризмы К. Пруткова и других уважаемых авторов.

Программа-консультант «Приемы менеджмента» v 6.0 (разработка компании «Intelpart»). Программа предназначена для:

- работников кадровых служб;
- руководителей подразделений и фирм, где работает более 10–15 человек;
- консультантов по управлению;
- имиджмейкеров;
- преподавателей ВУЗов, читающих курс «Менеджмент» и т.п.

Решаемые задачи:

- оперативная (в течение 15–20 минут) оценка сотрудников, партнеров, клиентов, конкурентов и выявление их сильных и слабых сторон;
- построение эффективной системы управления в данной ситуации, с учетом «несовершенности» людей.

Оценка проводится заочно. В программе предусмотрены специальные процедуры, фильтрующие субъективность оператора, работающего с программой. Статья о программе: http://www.triz–chance.ru/manager.html. Таким образом, экспертная система – это компьютерная программа, способная заменить специалиста-эксперта в решении проблемной ситуации.

Преимущества экспертных систем перед человеком-экспертом:

- у них нет предубеждений, и они устойчивы к различным помехам;
- они не делают поспешных выводов;
- эти системы выдают не первое нашедшееся, а оптимальное (по определенным критериям) решение;
- база знаний может быть очень большой. Введенные в машину один раз знания сохраняются навсегда. Человек же имеет ограниченную базу знаний, и если данные долгое время не используются, то они забываются и навсегда теряются.

Экспертные системы не заменяют специалиста, а являются его безэмоциональным советчиком, интеллектуальным партнером.

Экспертные системы, являющиеся основой искусственного интеллекта, получили широкое распространение в следующих областях:

- науке (классификация животных и растений по видам);
- в медицине (постановка диагноза, определение методов лечения);
- в технике (поиск неисправностей в технических устройствах, слежение за полетом космических кораблей и спутников);
- в политологии и социологии;
- криминалистике;
- лингвистике и т.д.

5.2.5. Системы поддержки принятия решений (СППР)

Считается, что если экспертные системы предназначены для принятия решения в достаточно стандартных ситуациях и позволяют воспользоваться опытом и знаниями высококвалифицированных специалистов в области принятия решения, то СППР предназначены для поддержки принятия решения в менее стандартных ситуациях при

управлении слабоструктурированными объектами. Синонимы: система поддержки решения; Decision-Making Support System (DMSS).

Для задач, которые относятся к области применения СППР, характерна неопределенность, делающая практически невозможным отыскание единственного объективно наилучшего решения. Поэтому при принятии решений в таких ситуациях должен использоваться более тонкий инструментарий определения системы предпочтений, более глубокий сопоставительный анализ альтернативных вариантов, необходимое информационное обеспечение лиц, принимающих решение.

Экспертные системы и СППР относятся к системам искусственного интеллекта. Остановимся подробнее на назначении и основах использования систем искусственного интеллекта.

5.2.6. Назначение и основы использования систем искусственного интеллекта

Термин интеллект (intelligence) происходит от латинского intellectus – что означает ум, рассудок, разум; мыслительные способности человека. Соответственно искусственный интеллект (artificial intelligence) ИИ (AI) обычно толкуется как свойство автоматических систем брать на себя отдельные функции интеллекта человека, например, выбирать и принимать оптимальные решения на основе ранее полученного опыта и рационального анализа внешних воздействий.

Понятие интеллекта человека. *Интеллектом* называется способность мозга решать (интеллектуальные) задачи путем приобретения, запоминания и целенаправленного преобразования знаний в процессе обучения на опыте и адаптации к разнообразным обстоятельствам.

Мы употребили термин интеллектуальная задача. Для того, чтобы пояснить, чем отличается интеллектуальная задача от просто задачи, необходимо ввести термин «алгоритм» – один из краеугольных терминов кибернетики.

Под *алгоритмом* понимают точное предписание о выполнении в определенном порядке системы операций для решения любой задачи из некоторого данного класса (множества) задач. В математике и кибернетике класс задач определенного типа считается решенным, когда для ее решения установлен алгоритм. Нахождение алгоритмов является естественной целью человека при решении им разнообразных классов задач. Отыскание алгоритма для задач некоторого данного типа связано с тонкими и сложными рассуждениями, требующими большой изобретательности и высокой квалификации. Принято считать, что

подобного рода деятельность требует участия интеллекта человека. Задачи, связанные с отысканием алгоритма решения класса задач определенного типа, будем называть интеллектуальными.

Что же касается задач, алгоритмы решения которых уже установлены, то, как отмечает известный специалист в области ИИ М. Минский, «излишне приписывать им такое мистическое свойство, как «интеллектуальность». В самом деле, после того, как такой алгоритм уже найден, процесс решения соответствующих задач становится таким, что его могут в точности выполнить человек, вычислительная машина (должным образом запрограммированная) или робот, не имеющие ни малейшего представления о сущности самой задачи.

Поэтому представляется совершенно естественным исключить из класса интеллектуальных такие задачи, для которых существуют стандартные методы решения. Примерами таких задач могут служить чисто вычислительные задачи: решение системы линейных алгебраических уравнений, численное интегрирование дифференциальных уравнений и т. д. В противоположность этим примерам для широкого класса интеллектуальных задач, таких, как распознавание образов, игра в шахматы, доказательство теорем и т. п., напротив, это формальное разбиение процесса поиска решения на отдельные элементарные шаги часто оказывается весьма затруднительным, даже если само их решение несложно.

Таким образом, можно перефразировать определение интеллекта: это универсальный сверхалгоритм, который способен создавать алгоритмы решения конкретных задач.

Еще интересным замечанием здесь является то, что профессия программиста, исходя из наших определений, является одной из самых интеллектуальных, поскольку продуктом деятельности программиста являются программы – алгоритмы в чистом виде.

Деятельность мозга (обладающего интеллектом), направленную на решение интеллектуальных задач, мы будем называть мышлением, или интеллектуальной деятельностью. Характерными чертами интеллекта, проявляющимися в процессе решения задач, является способность к обучению, обобщению, накоплению опыта (знаний и навыков) и адаптации к изменяющимся условиям в процессе решения задач. Таким образом, мозг, наделенный интеллектом, является универсальным средством решения широкого круга задач (в том числе неформализованных), для которых нет стандартных, заранее известных методов решения.

Следует иметь в виду, что существуют и другие, чисто поведенческие (функциональные) определения, например определение

А. Тьюринга. Его смысл заключается в следующем. В разных комнатах находятся люди и машина. Они не могут видеть друг друга, но имеют возможность обмениваться информацией (например, с помощью электронной почты). Если в процессе диалога между участниками игры людям не удается установить, что один из участников – машина, то такую машину можно считать обладающей интеллектом.

Интересен план имитации мышления, предложенный А. Тьюрингом. «Пытаясь имитировать интеллект взрослого человека, – пишет Тьюринг, – мы вынуждены много размышлять о том процессе, в результате которого человеческий мозг достиг своего настоящего состояния… Почему бы нам вместо того, чтобы пытаться создать программу, имитирующую интеллект взрослого человека, не попытаться создать программу, которая имитировала бы интеллект ребенка? Ведь если интеллект ребенка получает соответствующее воспитание, он становится интеллектом взрослого человека… Наш расчет состоит в том, что устройство, ему подобное, может быть легко запрограммировано… Таким образом, мы расчленим нашу проблему на две части: на задачу построения «программы-ребенка» и задачу «воспитания» этой программы». Можно сказать, что именно этот путь используют практически все системы ИИ.

Понятие интеллекта в информатике. Интеллектом в информатике называют действия, выполняемые устройствами или ПО, предоставляющими большие возможности обработки данных, а также сложные виды сервиса для пользователей. В соответствии с этим объект, обладающий способностями к этим действиям, называют интеллектуальным.

В последнее время широко используются понятия:

- интеллектуальная карточка;
- интеллектуальная телефония;
- интеллектуальная сеть;
- интеллектуальный агент;
- интеллектуальный анализ данных;
- интеллектуальный модем;
- система интеллектуальных аккумуляторов;
- интеллектуальный дом, в котором осуществлена автоматизация отопления, освещения и охраны;
- интеллектуальная автомашина, которая объезжает заторы на дорогах, сообщает водителю о неисправности своих частей;
- интеллектуальная автомагистраль, способная следить за движением транспортного средства и сообщать о его местонахождении,

автоматически брать плату за проезд. Для этого автомагистраль оборудована датчиками и видеокамерами.

Интеллектуальная сеть (Intelligent Network) – коммуникационная сеть, которая осуществляет не только передачу данных, но и предоставляет разнообразные виды сложных информационных услуг. Архитектура интеллектуальной сети, часто называемой разумной сетью, является концепцией, определяющей комплекс, включающий коммуникационную сеть с интеллектуальной подсетью. IN, в первую очередь, предоставляет пользователям разнообразные базы данных, дополнительные услуги, гибкое распределение и возможность перемещения сетевых функций по различным информационным системам, персональную адресацию и др.

Доступ со стандартным интерфейсом в IN сочетается с созданием распределенных баз данных. В результате в сети образуется общая логическая функциональная платформа, на которой располагаются прикладные процессы.

Появилось понятие интеллектуальной телефонной сети, которое затем распространилось и на другие типы сетей. Характерным примером такой сети является сотовая связь. Важное значение в IN получает интеграция коммутации с преобразованием интерфейсов. Это позволяет расширить спектр абонентов, включаемых в сеть, и обеспечить сопряжение каналов и сетей различных типов.

Интеллектуальный анализ данных (data mining) – процесс обнаружения взаимозависимостей групп данных. Сегодня наука, искусство и предпринимательство связаны с большими потоками данных, выдаваемыми средствами массовой информации (СМИ), специальными изданиями и соответствующим персоналом различных обществ. Стремление ускорить и сделать объективным принятие решений по разнообразным вопросам, возникающим в различных сферах деятельности, привело к автоматизации этого процесса. Интеллектуальный анализ данных (ИАД) основывается на выявлении взаимосвязи данных с помощью статистических процедур. При этом используются нейронные сети, деревья решений, визуализация данных.

Новая технология появилась в результате:

- осознания того, что в потоках данных содержится ценная информация;
- создания и развития информационных хранилищ, образующих единое информационное пространство;
- снижения стоимости систем сбора, передачи, хранения и обработки данных;

- производства запоминающих устройств (ЗУ), обладающих очень большой памятью.

Выделяют две методологии интеллектуального анализа данных. Первый из них связан с автоматизированным поиском знаний. Он охватывает прогнозное моделирование, анализ взаимосвязей отдельных данных. Второй метод ИАД определяется высказыванием и проверкой разнообразных гипотез. Он охватывает аналитическую обработку в реальном времени OLAP, статистический анализ, генерацию отчетов с помощью команд языка структурированных запросов SQL. Структура систем ИАД опирается на информационное хранилище. Селекция заключается, здесь в отборе данных, которые будут подвергнуты ИАД. Предварительная обработка обеспечивает создание нужных структур и форматов данных. Все это позволяет осуществить ИАД и получить необходимые знания.

Системы ИАД просматривают большие потоки данных и по определенным алгоритмам выявляют закономерности, которые не являются очевидными для пользователей. Системы используются практически во всех сферах деятельности человечества: в науке (обработка результатов научных исследований, опросов общественного мнения, изучение поведения специальных групп населения), в информатике (распознавание образов речи и др.), в предпринимательстве (оптимизация производственных процессов, анализ корзин товаров, оценка рисков, управление портфелем ценных бумаг, оценка недвижимого имущества), в медицине (поиск зависимостей между симптомами и заболеваниями, диагностика заболеваний, классификация пациентов и т.д.).

ИАД проводится таким образом, что позволяет относительно легко анализировать данные и представлять информацию, позволяющую исследовать нужную проблему и принимать мотивированные решения. Для этой цели создаются комплексы программ, образующих так называемое программное обеспечение интеллектуального анализа и обнаружения данных.

Краткая история развития науки об искусственном интеллекте

Искусственный интеллект – одна из новейших наук, появившихся во второй половине 20–го века на базе вычислительной техники, математической логики, программирования, психологии, лингвистики, нейрофизиологии и других отраслей знаний. Искусственный интеллект – это образец междисциплинарных исследований, где соединяются профессиональные интересы специалистов разного профиля.

Название новой науки возникло в конце 60-х годах, а в 1969 г. в Вашингтоне (США) состоялась первая Всемирная конференция по

искусственному интеллекту. Известно, что совокупность научных исследований обретает права науки, если выполнены два необходимых условия:

- исследования должен иметь объект изучения, не совпадающий с теми, которые изучают другие науки;
- должны существовать специфические методы исследования этого объекта, отличные от методов других, уже сложившихся наук.

Когда в конце 40-х – начале 50-х годов появились ЭВМ, стало ясно, что инженеры и математики создали не просто быстро работающее устройство для вычислений, а нечто более значительное, способное автоматизировать такие виды человеческой деятельности, которые называются интеллектуальными и считаются доступными лишь человеку. Можно сказать, что первые компьютеры «не отличали» вычислительные программы от невычислительных. Они одинаковым образом находили корни квадратного уравнения или писали стихи. В памяти компьютера не было знаний о том, что он на самом деле делает. Об интеллекте компьютера можно было бы говорить, если бы он сам, на основании собственных знаний о том, как протекает игра в шахматы и как играют в эту игру люди, сумел составить шахматную программу или синтезировал программу для написания несложных вальсов и маршей. Не сами процедуры, с помощью которых выполняется та или иная интеллектуальная деятельность, а понимание того, как их создать, как научиться новому виду интеллектуальной деятельности, – вот где скрыто то, что можно назвать интеллектом. Специальные метапроцедуры обучения новым видам интеллектуальной деятельности отличают человека от компьютера.

Следовательно, в создании искусственного интеллекта основной задачей становится реализация машинными средствами тех метапроцедур, которые используются в интеллектуальной деятельности человека. Что же это за процедуры?

В психологии мышления есть несколько моделей творческой деятельности. Одна из них называется лабиринтной. Суть *лабиринтной гипотезы,* на которой основана лабиринтная модель, состоит в следующем: переход от исходных данных задачи к решению лежит через лабиринт возможных альтернативных путей. Не все пути ведут к желаемой цели, многие из них заводят в тупик, надо уметь возвращаться к тому месту, где потеряно правильное направление. Это напоминает попытки не слишком умелого школьника решить задачу об упрощении алгебраических выражений. Для этой цели на каждом шагу можно применять некоторые стандартные преобразования или придумывать искусственные приемы. Но весьма часто вместо

упрощения выражения происходит его усложнение, и возникают тупики, из которых нет выхода. По мнению сторонников лабиринтной модели мышления, решение всякой творческой задачи сводится к целенаправленному поиску в лабиринте альтернативных путей с оценкой успеха после каждого шага. С лабиринтной моделью связана первая из метапроцедур – целенаправленный поиск в лабиринте возможностей.

Программированию этой метапроцедуры соответствуют многочисленные процедуры поиска, основанные на соображениях здравого смысла (человеческого опыта решения аналогичных задач). В 60-х годах было создано немало программ на основе лабиринтной модели, в основном игровых и доказывающих теоремы «в лоб», без привлечения искусственных приемов. Соответствующее направление в программировании получило название *эвристического программирования.*

Высказывались даже предположения, что целенаправленный поиск в лабиринте возможностей – универсальная процедура, пригодная для решения любых интеллектуальных задач. Но исследователи отказались от этой идеи, когда столкнулись с задачами, в которых лабиринта возможностей либо не существовало, либо он был слишком велик для метапроцедуры поиска, как, например, при игре в шахматы. Конечно, в этой игре лабиринт возможностей – это все мыслимые партии игры. Но как в этом астрономически большом лабиринте найти те партии, которые ведут к выигрышу? Лабиринт столь велик, что никакие мыслимые скорости вычислений не позволят целенаправленно перебрать пути в нем. И все попытки использовать для этого человеческие эвристики (в данном случае профессиональный опыт шахматистов) не дают пути решения задачи. Поэтому созданные шахматные программы уже давно используют не только метапроцедуру целенаправленного поиска, но и другие метапроцедуры, связанные с другими моделями мышления.

Долгие годы в психологии изучалась *ассоциативная модель мышления.* Основной метапроцедурой модели является ассоциативный поиск и ассоциативное рассуждение. Предполагается, что решение неизвестной задачи так или иначе основывается на уже решенных задачах, чем–то похожих на ту, которую надо решить. Новая задача рассматривается как уже известная, хотя и несколько отличающаяся от известной. Поэтому способ ее решения должен быть близок к тому, который когда-то помог решить подобную задачу. Для этого надо обратиться к памяти и попытаться найти нечто похожее, что ранее уже встречалось. Это и есть *ассоциативный поиск*. Когда, увидев

незнакомого человека, вы стараетесь вспомнить, на кого он похож, реализуется метапроцедура ассоциативного поиска. Но понятие ассоциации в психологии шире, чем просто «похожесть». Ассоциативные связи могут возникнуть и по контрасту, как противопоставление одного другому, и по смежности, т. е. в силу того, что некоторые явления возникали в рамках одной и той же ситуации или происходили одновременно (или с небольшим сдвигом по времени).

Ассоциативное рассуждение позволяет переносить приемы, использованные ранее, на текущую ситуацию. К сожалению, несмотря на многолетнее изучение ассоциативной модели, не удалось создать стройную теорию ассоциативного поиска и ассоциативного рассуждения. Исключение составляет важный, но частный класс ассоциаций, называемых условными рефлексами. И все же метапроцедура ассоциативного поиска и рассуждения сыграла важную роль: она помогла создать эффективные программы в распознавании образов, в классификационных задачах и в обучении ЭВМ.

Но одновременно эта метапроцедура привела к мысли о том, что для ее эффективного использования надо привлечь результаты, полученные в другой модели мышления, опирающейся на идею внутреннего представления проблемной области, на знания об ее особенностях, закономерностях и процедурах действия в ней. Это представление о мыслительной деятельности человека обычно называют *модельной гипотезой*. Согласно ей, мозг человека содержит модель проблемной ситуации, в которой ему надо принять решение. Для решения используются метапроцедуры, оперирующие с совокупностью знаний из той проблемной области, к которой принадлежит данная проблемная ситуация.

Например, если проблемная ситуация – переход через улицу с интенсивным движением, то знания, которые могут помочь ее разрешить, касаются способов организации движения транспорта, сигналов светофоров, наличия дорожек для перехода и т. п. В модельной гипотезе основными метапроцедурами становятся представление знаний, рассуждения, поиск релевантной (связанной с данной проблемной ситуацией) информации в совокупности имеющихся знаний, их пополнение и корректировка.

Эти метапроцедуры составляют ядро интеллектуальных возможностей современных программ и программных систем, ориентированных на решение творческих задач. В совокупности с метапроцедурами целенаправленного поиска в лабиринте возможностей, ассоциативного поиска и рассуждения они образуют арсенал интеллектуальных средств, которым располагают современные

интеллектуальные системы, часто называемые системами, основанными на знаниях.

Известно, что в свое время ученый А. Тьюринг предложил специальный критерий, с помощью которого определяют, может ли машина мыслить. Согласно этому критерию машина может быть признана мыслящей, если эксперт, ведя с ней диалог по достаточно широкому кругу вопросов, не сможет отличить ее ответов от ответов разумного человека.

ИИ разделяется на два научных направления: нейрокибернетику (или искусственный разум) и кибернетику «черного ящика» (или машинный интеллект).

Кибернетика – это наука об управлении, связи и переработке информации. Кибернетика исследует объекты независимо от их материальной природы (живые и неживые системы).

Первое направление – нейрокибернетика – базируется на аппаратном моделировании работы головного мозга человека. Основой мозга является большое число (около 14 миллиардов) связанных и взаимодействующих нервных клеток – нейронов.

Системы искусственного интеллекта, которые моделируют работу головного мозга, называют *нейронными сетями* (или нейросетями). Первые нейросети были созданы в конце 50-х годов XX столетия американскими учеными Г. Розенблаттом и П. Мак-Каллоком.

Для второго направления ИИ – *кибернетики «черного ящика»* – не имеет значения, какова конструкция «мыслящего» устройства. Главное, чтобы на заданные входные воздействия оно реагировало так же, как человеческий мозг.

Искусственный интеллект и образование. Происходит смена ведущего субъекта образования: принцип преподавателя «следуй за мной» меняется на принцип учащегося «веди себя сам» с помощью средств обучения и консультаций преподавателей.

Взаимовлияние искусственного интеллекта и образования:

- типология знаний и анализ возможностей их формализации для автоматизации обучения и контроля;
- подготовка и переподготовка инженеров по знаниям в России: анализ гуманитарной (психологической и лингвистической), математической и программистской составляющих, логистика;
- использование инженерии знаний: теоретические аспекты получения знаний, практические методы извлечения знаний;
- программные реализации (языки, оболочки и прикладные экспертные системы (ЭС) и нейросистемы, САПР обучающих курсов);

- интеллектуальные процедуры, реализованные в зарубежных и в отечественных интеллектуальных обучающих (агентные курсы-роботы), справочных, моделирующих, тренирующих и (самогенерирующихся) контролирующих системах.

Мысль Монтеня «ум, хорошо устроенный, лучше, чем ум, хорошо наполненный» в широком смысле относится к принципам обучения, а в узком – к интеллектуализации любых информационных (особенно, распределенных) систем, т.е. к необходимости разработки систем ИИ для оптимальной реализации запросов к ИС и принятий эффективных решений.

Цели и задачи искусственного интеллекта. Имитация функций разума привела к развитию искусственного интеллекта – способности устройства или прикладного процесса обнаруживать свойства, ассоциируемые с разумным поведением человека.

Таким образом, можно сформулировать основные цели и задачи искусственного интеллекта. *Объектом изучения* ИИ являются метапроцедуры, используемые при решении человеком задач, традиционно называемых интеллектуальными, или творческими. Но если психология мышления изучает эти метапроцедуры применительно к человеку, то искусственный интеллект создает программные (а сейчас уже и программно-аппаратные) модели таких метапроцедур.

Цель исследований в области искусственного интеллекта – создание арсенала метапроцедур, достаточного для того, чтобы ВМ (или другие технические системы, например роботы) могли находить по постановкам задач их решения. Иными словами, стали автономными программистами, способными выполнять работу профессиональных программистов-прикладников (создающих программы для решения задач в определенной предметной области).

Разумеется, сформулированная цель не исчерпывает всех задач, которые ставит перед собой искусственный интеллект. Это цель ближайшая. Последующие цели связаны с попыткой проникнуть в области мышления человека, которые лежат вне сферы рационального и выразимого словесно (вербально) мышления, так как в поиске решения многих задач, особенно сильно отличающихся от ранее решенных, большую роль играет та сфера мышления, которую называют *подсознательной*, бессознательной, или интуитивной.

Задачей искусственного интеллекта является придание системам способности обучаться и «думать», решение которой осуществляется по нескольким направлениям:

- создание эффективно функционирующих обучающих систем;

- разработка экспертных систем, определяемых набором взаимосвязанных правил, формулирующих опыт специалистов в некоторой области, и механизмом решения, позволяющим распознать ситуацию, ставить диагноз, давать рекомендации к действию;
- использование диагностических систем в исследовании явлений и процессов, например, для анализа крови, управления доменным производством, изучения состояния нефтяных полей;

– решение многих задач распознавания образов:

1) распознавания речи, что позволяет компьютерам понимать естественные языки, правда, пока в ограниченной области применения;

2) внедрение в реальную жизнь технологии и идентификации отпечатков пальцев;

3) серьезные достижения в распознавании передних частей головы человека. Эти задачи уже решены для тех случаев, когда в этом заинтересованы сами люди, например, опознание для работы с банковских системах и т.д.

Важную роль в развитии искусственного интеллекта играют специально создаваемые для этой цели языки. Среди них следует выделить язык LISP и язык PROLOG.

Язык LISP (LISP language) – специализированный язык обработки списков. Все объекты языка (программы и данные) рассматриваются как списки. Обеспечивается работа пользователя с информационной системой в режиме диалога. В отличие от числовой обработки представление данных в виде списков позволяет выражать понятия. LISP может быть также использован для программирования прикладных процессов, не связанных с искусственным интеллектом, например, с обычными расчетами. На основе LISP разработаны средства программирования с удобными интерфейсами пользователя. LISP позволяет также работать со структурами данных, размеры и состав которых видоизменяются в ходе выполнения прикладных процессов.

Язык PROLOG (PROLOG language). «Программирование в понятиях логики» (PROLOG) представляет собой совокупность логических утверждений и правил. Утверждения состоят из условий (предикатов), связок, констант, образуя базу данных (БД). Правила имеют вид: «А, если В и D». Основным элементом языка является так называемый «атом», который выражает отношения между отдельными объектами.

PROLOG – это новый подход к использованию формальной логики. Пользуясь этим языком, программист имеет дело непосредственно с

логическими связями между понятиями. Для этого в языке определяются правила, которые могут выполняться для достижения целей, поставленных перед прикладным процессом.

Основными методами, используемыми в искусственном интеллекте, являются разного рода программные модели и средства, эксперимент на ВМ и теоретические модели. Однако современные ВМ уже мало удовлетворяют специалистов по искусственному интеллекту. Они не имеют ничего общего с тем, как устроен человеческий мозг. Поэтому идет интенсивный поиск новых технических структур, способных лучше решать задачи, связанные с интеллектуальными процессами. Сюда относятся исследования по нейроподобным искусственным сетям, попытки построить молекулярные машины, работы в области голографических систем и многое другое.

5.2.7. Базы знаний

*Данные–информация–знания***.** В обиходе эти термины часто отождествляют. В популярной или научной литературе нет универсальных определений этих понятий. Можно говорить лишь о некоторых подходах, разделяемых или не разделяемых различными авторами.

Данные – сведения, представленные в определенной знаковой системе и на определенном материальном носителе для обеспечения возможностей хранения, передачи, приема и обработки.

Информация – это данные, сопровождающиеся смысловой нагрузкой, помещенные в некоторый контекст. Как правило, получение информации связывают с уменьшением неопределенности существующего выбора; ответ на какой-либо заданный либо подразумеваемый вопрос. (При этом то, что для одних личностей (или с одной точки зрения) может быть данными, для других вполне может быть информацией).

Знание – зафиксированная и проверенная практикой информация, которая может многократно использоваться людьми для решения тех или иных задач (глубинные знания, процедурные знания).

Несистематизированная информация – это просто *сведения.* Только определенным образом организованные сведения имеют смысл. Однако в зависимости от формы организации одна и та же информация может иметь разный смысл.

Знание, которое не используется и не возрастает, в конечном счете устаревает и становится бесполезным, точно так же, как деньги, которые хранятся, не превращаясь в оборотный капитал, в конечном

счете обесцениваются. Знание же, которое распространяется, приобретается и обменивается, наоборот, генерирует новое знание.

База знаний – наиболее важная компонента экспертной системы (ЭС), на которой основаны ее «интеллектуальные способности». В отличие от всех остальных компонент ЭС, база знаний – «переменная » часть системы, которая может пополняться и модифицироваться инженерами знаний и опыта использования ЭС, между консультациями (а в некоторых системах и в процессе консультации). Существует несколько способов представления знаний в ЭС, однако общим для всех них является то, что знания представлены в символьной форме (элементарными компонентами представления знаний являются тексты, списки и другие символьные структуры). В результате в ЭС реализуется принцип *символьной природы* рассуждений, который заключается в том, что процесс рассуждения представляется как последовательность символьных преобразований.

Наиболее распространенный способ представления знаний – в виде конкретных *фактов и правил*, по которым из имеющихся фактов могут быть выведены новые. Факты представлены, например, в виде *троек*:

(АТРИБУТ–ОБЪЕКТ–ЗНАЧЕНИЕ)

Такой факт означает, что заданный объект имеет заданный атрибут (свойства) с заданным значением. Например, тройка

(ТЕМПЕРАТУРА–ПАЦИЕНТ1 37.5)

представляет факт «температура больного, обозначаемого ПАЦИЕНТ1, равна 37.5».

В более простых случаях факт выражается неконкретным значением атрибута, а каким либо простым *утверждением*, которое может быть *истинным* или *ложным*, например: «Небо покрыто тучами». В таких случаях факт можно обозначить каким-либо кратким *именем* (например, ТУЧИ) или использовать для представления факта сам текст соответствующей фразы.

Правила в базе знаний имеют вид:

ЕСЛИ A **ТО** S, где A – *условие*; S – *действие*. Действие S исполняется, если A истинно. Наиболее часто действие S, так же как и условие, представляет собой *утверждение*, которое может быть *выведено* системой (то есть становится ей известной), если истинно условие правила A. Правила в базе знаний служат для представления *эвристических знаний (эвристик)*, т.е. неформальных правил рассуждения, вырабатываемых экспертом на основе опыта его деятельности.

Простой пример правила из повседневной жизни:

ЕСЛИ небо покрыто тучами,

ТО скоро пойдет дождь.

В качестве условия А может выступать либо факт (как в данном примере), либо несколько фактов A1...AN, соединенных логической операцией *и*:

A1 *и* A2 *и* ... *и* A_N.

В математической логике такое выражение называется *конъюнкцией.* Оно считается истинным в том случае, если истинны *все* его компоненты. Пример предыдущего правила с более сложным условием:

ЕСЛИ небо покрыто тучами **и** барометр падает,

ТО скоро пойдет дождь. (Правило 1).

Действия, входящие в состав правил, могут содержать новые факты. При применении таких правил эти факты становятся известны системе, т.е. включаются во множество фактов, которое называется рабочим множеством. Например, если факты «Небо покрыто тучами» и «Барометр падает» уже имеются в рабочем множестве, то после применения приведенного выше правила в него также включается факт «Скоро пойдет дождь».

Если система не может вывести некоторый факт, истинность или ложность которого требуется установить, то система спрашивает о нем пользователя. Например:

ВЕРНО ЛИ, ЧТО небо покрыто тучами?

При получении положительного ответа от пользователя факт «Небо покрыто тучами» включается в рабочем множество.

Существуют *динамические* и *статические* базы знаний. Динамическая база знаний изменяется со временем. Ее содержимое зависит и от состояния окружающей. Новые факты, добавляемые в базу знаний, являются результатом вывода, который состоит в применении правил к имеющимся фактам.

В системах с монотонным выводом факты, хранимые в базе знаний, статичны, то есть не изменяются в процессе решения задачи. В системах с немонотонным выводом допускается изменение или удаление фактов из базы знаний. В качестве примера системы с немонотонным выводом можно привести ЭС, предназначенную для составления перспективного плана капиталовложения компании. В такой системе по вашему желанию могут быть изменены даже те данные, которые после вывода уже вызвали срабатывание каких-либо правил. Иными словами, имеется возможность модифицировать значения атрибутов в составе фактов,

находящихся в рабочей памяти. Изменение фактов, в свою очередь, приводит к необходимости удаления из базы знаний заключений, полученных с помощью упомянутых правил. Тем самым вывод выполняется повторно для того, чтобы пересмотреть те решения, которые были получены на основе подвергшихся изменению фактов.

Управление знаниями – процесс создания условий для выявления, сохранения и эффективного использования знаний и информации в сообществе, это стратегия, направленная на предоставление знаний в нужное время тем членам сообщества, которым эти знания необходимы, для того чтобы повысить эффективность деятельности сообщества.

Понятие *«управление знаниями»* родилось в середине 90-х годов в крупных корпорациях, где проблемы обработки информации приобрели особую остроту, став критическими. Выяснилось, что основное узкое место – это обработка знаний, накопленных специалистами компании (именно такие знания обеспечивают ей преимущество перед конкурентами). Тогда и появились первые базы знаний – информационные системы, содержащие в структурированном виде информацию об имеющихся в организации знаниях, компетенциях, лучших практиках, эвристиках, а также о сотрудниках – носителях знаний и компетенций, экспертах и консультантах в различных предметных областях. База знаний позволяет сохранять и эффективно использовать электронные документы (текстовые, изобразительные, звуковые, видео и др.), локализованные как в самой системе, так и доступные ей через телекоммуникационные сети, и представлять их пользователю в удобном для него виде.

Представление знаний. Вопрос о принятии решений экономической информационной системой в современных системах управления объектом требует наличия знаний об этом управляемом объекте и реализации моделей принятия решений, характерных для специалиста (инженера, технолога, экономиста, бухгалтера). Система понятий *для представления знаний* существенно отличается от понятий для представления данных, поэтому отображение знаний производится в базу знаний. Кроме того, база знаний хранит данные как простую разновидность знаний.

Запросы, которые формулируются пользователями информационной системы, реализуются одним из двух возможных способов:

- сообщения, являющиеся ответом на запрос, хранятся в *явном виде* в БД, и процесс получения ответа представляет собой выделение подмножества значений из файлов БД, удовлетворяющих запросу;

- ответ не существует в явном виде в БД и формируется в процессе *логического вывода* на основании имеющихся данных.

Последний случай принципиально отличается от технологии использования обычных баз данных и рассматривается в рамках представления знаний, т. е. информации, необходимой в процессе получения новых фактов.

База знаний содержит сведения следующих типов:

1) отражающие существующие в предметной области закономерности и позволяющие выводить новые факты, справедливые в данном состоянии предметной области, но отсутствующие в БД, а также прогнозирующие потенциально возможные состояния предметной области;

2) о структуре экономических информационных систем (ЭИС) и БД (метаинформация);

3) обеспечивающие понимание входного языка, т. е. перевод входных запросов во внутренний язык.

Принято говорить не о «знаниях вообще», а о знаниях, зафиксированных с помощью той или иной модели знаний.

Принципиальными различиями обладают три *модели представления знаний*:

- продукционная модель;
- модель фреймов;
- модель семантических сетей.

Продукционная модель состоит из трех основных компонентов. Первый из них – это *набор правил*, представляющий собой в продукционной системе базу знаний. Вторым компонентом является *рабочая память*, в которой хранятся исходные факты и результаты выводов, полученных из этих фактов. Третьим компонентом служит *механизм логического вывода*, использующий правила в соответствии с содержимым рабочей памяти и формирующий новые факты. Каждое правило содержит *условную и заключительную части.* В условной части правила находится *одиночный факт* либо несколько фактов (условий), соединенных логической операцией «И». В заключительной части правила находятся *факты*, которые необходимо *дополнительно сформировать* в рабочей памяти, если условная часть правила является истинной.

Если для получения вывода правила применяются к фактам, записанным в рабочей памяти, и в результате применения правил добавляются новые факты, то такой способ действий называется *прямым выводом.* Возможен также *обратный вывод* целей. В качестве

цели выступает подтверждение истинности факта, отсутствующего в рабочей памяти. При обратном выводе исследуется возможность применения правил, подтверждающих цель, необходимые для этого дополнительные факты становятся новыми целями, и процесс повторяется.

В случае обратного вывода условием завершения работы системы является окончание списка правил, которые относятся к доказываемым целям. При прямом выводе завершение работы происходит по окончании списка применимых правил. Следует отметить, что на каждом шаге вывода количество одновременно применимых правил может быть любым. Последовательность выбора подходящих правил не влияет на однозначность получаемого ответа, однако может существенно увеличить требуемое число шагов вывода. В реальных базах знаний с большим числом правил это может существенно снизить быстродействие системы. В системах с обратным выводом есть возможность исключить из рассмотрения правила, не имеющие отношения к выводу требуемых целей, и тем самым несколько ослабить указанный отрицательный эффект. По этой причине системы с обратным выводом целей получили большее распространение.

Представление знаний в виде набора правил имеет следующие преимущества:

- простота создания и понимание отдельных правил;
- простота механизма логического вывода.

К недостаткам этого способа организации базы знаний относятся:

- неясность взаимных отношений правил;
- отличие от человеческой структуры знаний.

Модель фреймов. В основе теории фреймов лежит фиксация знаний путем сопоставления новых фактов с рамками, определенными для каждого объекта в сознании человека. Структура в памяти ВМ, представляющая эти рамки, называется *фреймом.* С помощью фреймов представляют процесс систематизации знаний в форме, максимально близкой к принципам систематизации знаний человеком.

Фрейм представляет собой таблицу, структура и принципы организации которой являются развитием понятия отношения в реляционной модели данных. Новизна фреймов определяется двумя условиями:

- имя атрибута может в ряде случаев занимать в фрейме позицию значения;
- значением атрибута может служить имя другого фрейма или имя программно-реализованной процедуры.

Структура фрейма показана в таблице 5.1.

Таблица 5.1. Структура фрейма

Имя слота	Указатель наследования	Указатель типа	Значение слота
Слот 1			
Слот 2			
…			
Слот N			

Слотом фрейма называется элемент данных, предназначенный для фиксации знаний об объекте, которому отведен данный фрейм.

Параметрами слотов являются следующие три компоненты.

Имя слота. Каждый слот должен иметь уникальное имя во фрейме, к которому он принадлежит. Имя слота в некоторых случаях может быть служебным. Среди служебных имен отметим имя пользователя, определяющего фрейм, дату определения или модификации фрейма, комментарий.

Указатель наследования. Он показывает, какую информацию об атрибутах слотов во фрейме верхнего уровня наследуют слоты с теми же именами во фрейме нижнего уровня.

Типичными указателями наследования являются:

S (тот же). Слот наследуется с теми же значениями данных;

U (уникальный). Слот наследуется, но данные в каждом фрейме могут принимать любые значения;

I (независимый). Слот не наследуется.

Указатель типа данных. К типам данных (язык Пролог) относятся:

FRAME (указатель) – указывает имя фрейма верхнего уровня;

ATOM (переменная);

TEXT (текстовая информация);

LIST (список);

LISP (присоединенная процедура).

С помощью механизма управления наследованием по отношениям «есть–нек» осуществляются автоматический поиск и определение значений слотов фрейма верхнего уровня и присоединенных процедур.

Преимущества фреймовых систем по сравнению с продукционной моделью представления знаний:

1) знания организованы на основе концептуальных объектов;

2) допускается комбинация представления декларативных (как устроен объект) и процедурных (как взаимодействует объект) знаний;

3) иерархия фреймов соответствует классификации понятий, привычной для восприятия человеком;

4) система фреймов легко модифицируется и расширяется.

Модель семантических сетей. Особенность семантической сети как модели знаний состоит в единстве базы знаний и механизма вывода новых фактов. На основании вопроса к базе знаний строится семантическая сеть, отображающая структуру вопроса, и ответ получается в результате сопоставления общей сети для базы знаний в целом и сети для вопроса.

Например, семантическая сеть, отображающая подчиненность преподавателей на кафедре в вузе, приведена на рис. 5.13, *а*. Приводятся связи, показывающие подчиненность первого преподавателя. Остальные преподаватели кафедры вуза связываются через вершины сети «руководит 2», «руководит 3» и т.д.

Вопрос «Кто руководит Молниной?» представлен в виде подсети, показанной на рис. 5.13, *б*.

Сопоставление общей сети с сетью запроса начинается с фиксации вершины «руководит», имеющей ветвь «объект», направленную к вершине «Молнина Е.В.». Затем производится переход по ветви «руководит», что и приводит к ответу «Мицель А.А.»

Преимущества семантических сетей состоят в том, что это достаточно понятный способ представления знаний на основе отношений между вершинами и дугами сети. Однако с увеличением размеров сети ухудшается ее обозримость и увеличивается время вывода новых фактов с помощью механизма сопоставления.

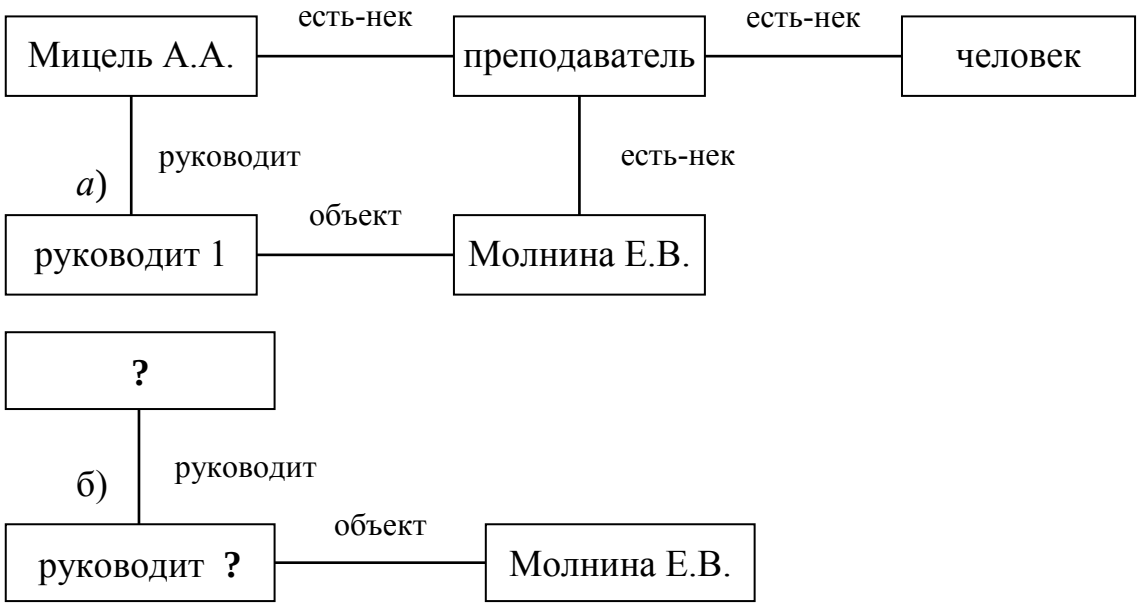


Рис. 5.13. Примеры семантической сети (а) и сети логического вывода для запроса (б)

Модели знаний – продукционная, фреймовая и модель семантических сетей – обладают практически равными возможностями представления знаний, использующих отношения «есть–нек» и «есть–часть». Кроме того, каждая модель знаний содержит средства усиления этой «базовой» конфигурации, а именно:

- продукционная модель позволяет легко расширять и усложнять множество правил вывода;
- фреймовая модель позволяет усилить вычислительные аспекты обработки знаний за счет расширения множества присоединенных процедур;
- модель семантических сетей позволяет расширять список отношений между вершинами и дугами сети, приближая выразительные возможности сети к уровню естественного языка.

5.2.8. Примеры сред интеллектуальной обработки данных

Яркими представителями программ ИИ являются электронные переводчики и словари. Считается, что история машинного перевода началась с эксперимента, проведенного в Джорджтаунском университете в 1954 г. Впервые текст, написанный на русском языке, был переведен на английский язык с помощью ЭВМ.

В России наибольшее распространение получили две программы перевода с одного естественного языка на другой: Stylus (фирма «ПроМТ» или ПРОект МТ) и Socrat (фирма «Арсеналъ»).

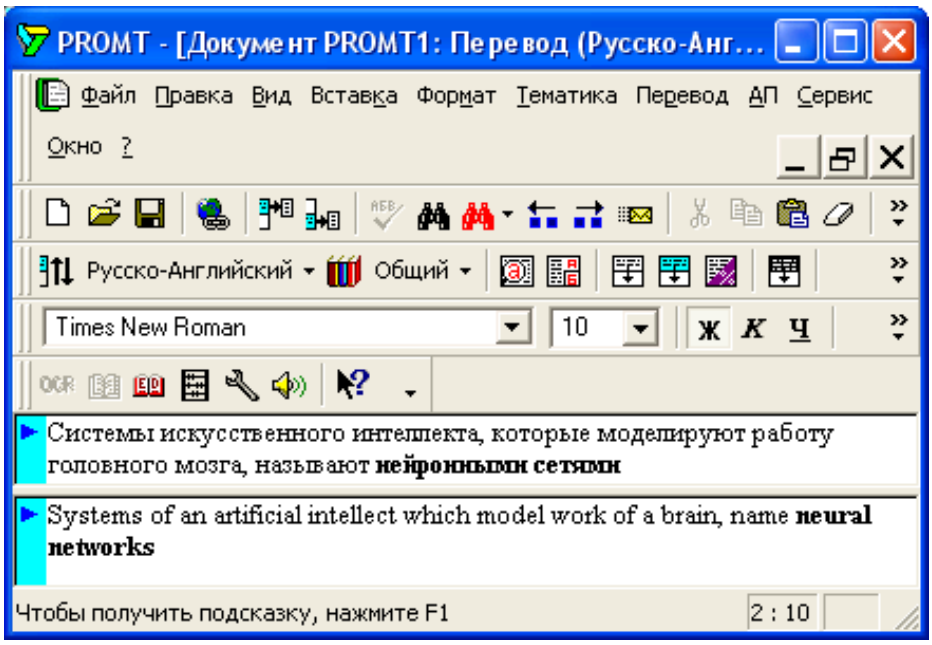


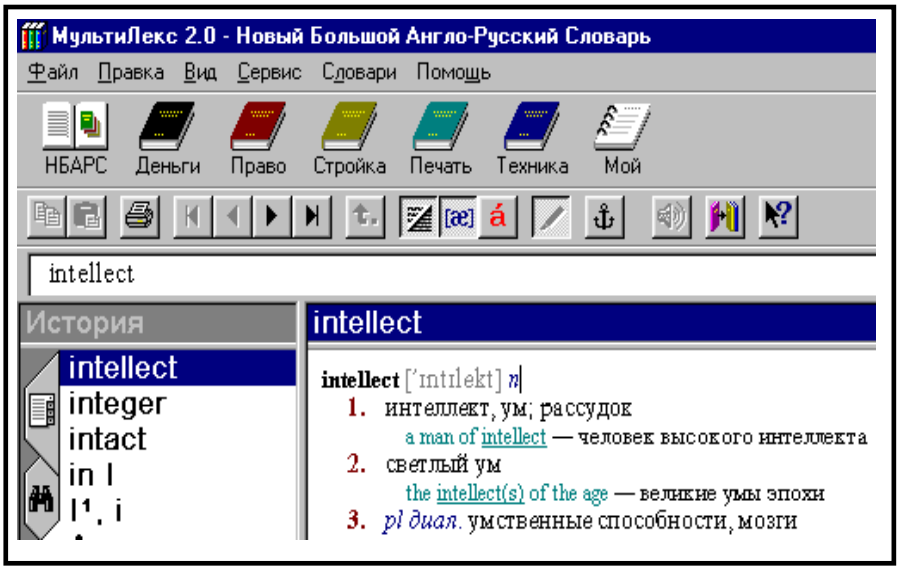
Рис. 5.14. Пользовательский интерфейс переводчика PROMT XT

Системы Stylus (последние версии получили имя PROMT или ПРОМТ) переводят со многих языков и обрабатывают документы большинства распространенных форматов, число специализированных словарей составляет несколько десятков.

На рисунке 5.14 показан пользовательский интерфейс переводчика PROMT XT. Видны два окна, причем в верхнем содержится исходный текст на русском языке, а во втором окне – его перевод на английский язык. Система PROMT позволяет подключать несколько специализированных словарей для перевода текстов, относящихся к различным предметным областям (математика, информатика, электротехника, коммерция и т.д.).

Конечно, качество перевода еще далеко от идеального варианта. В этом легко убедиться, сделав перевод с одного языка на другой, а затем выполнив обратный перевод. Так, поговорка «Сделал дело, гуляй смело» после двойного перевода (на английский язык, а затем обратно) выглядит так: «Сделал бизнес, выходят на прогулку безопасно». В переводе слышится тюремный мотив (шутка).

Еще один пример. Перевод на немецкий язык, а затем обратный перевод на русский язык поговорки «Любопытной Варваре на базаре нос оторвали» дал такой результат: «Любопытным варварам на рынке нос имеют оборванно». На следующем рисунке 5.15 показан электронный словарь МультиЛекс 2.0 (компания МедиаЛингва). Его особенность состоит в том, что он позволяет прослушать произношение



английского слова (так называемый говорящий словарь).

Рис. 5.15. Электронный словарь МультиЛекс 2.0

При переводе можно выбирать предметную область (финансы, право, техника, строительство, печать), к которой относится переводимый термин.

В 2006 году английские ученые из Университета Лидса продемонстрировали программу, которая самостоятельно изучила правила игры в шашки, наблюдая за людьми. Агентство перспективных военных исследований (DARPA) при Минобороны США недавно объявило о завершении первого этапа работ по созданию системы, способной «вдумчиво» читать. Ее прототип сумел пройти несложный IQ-тест. Кажется, еще чуть-чуть – и мыслящие машины перестанут быть фантастикой.

За 10 лет DARPA планирует потратить на разработку искусственного интеллекта около $8 млрд. Европейский союз на эти цели выделил 2 млрд. евро на пять лет. Примерно столько же собираются вложить Япония и Южная Корея.

Программы интеллектуальной обработки текста. Абриаль – инструментальная среда для конструирования сложных баз знаний в виде активных объектных сетей (рис.5.16).

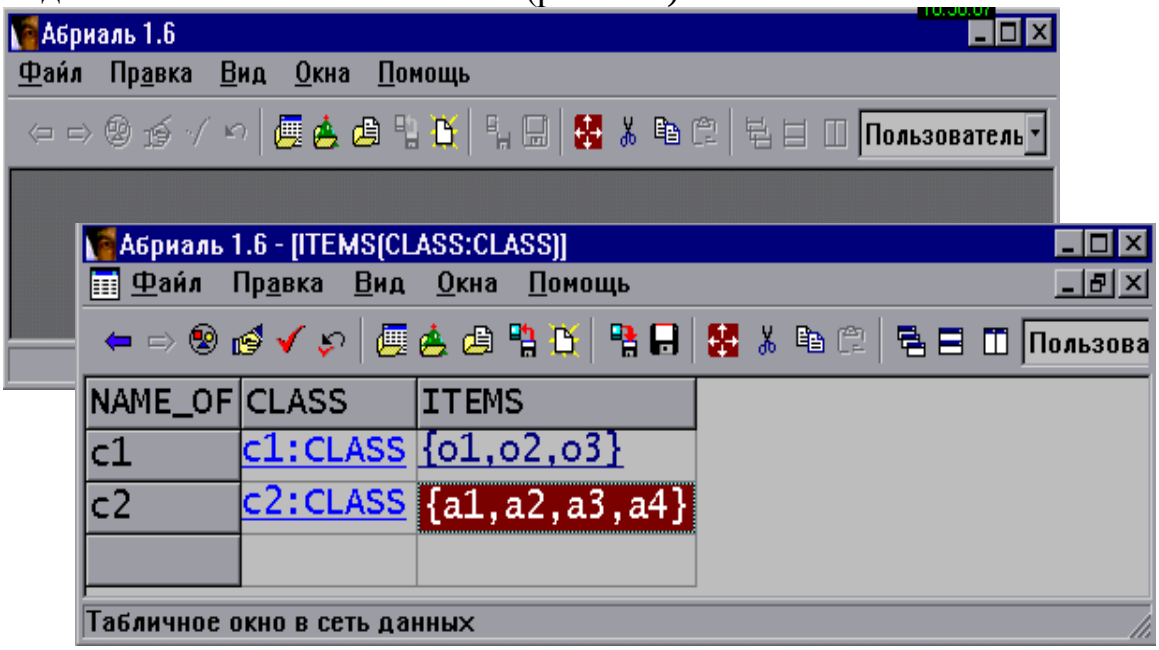


Рис.5.16. Окно среды «Абриаль»

Абриаль можно рассматривать как принципиально новое хранилище данных, в котором пользователь работает с информацией через автоматически формируемый интерфейс *гипертаблиц*, позволяющий осуществлять навигацию в любых направлениях, отражающих ассоциативные связи данных.

Оригинальная высокоуровневая система программирования Абриаля, основываясь на принципиально новой организации данных, делает ассоциативно-продукционные вычисления такими же эффективными, как классические императивные программы.

Вычислительная модель Абриаля, с одной стороны, обобщает продукционную, логическую и функциональную модели, и с другой стороны – снимает известные ограничения как по объёму данных, так и по сложности их структуры и функциональности.

Абриаль предназначен для исследования семантических сетей, лингвистических баз данных, т.е. прежде всего для тех областей, где до сих пор объём данных или размер продукционной системы являются камнем преткновения. Способность эффективно работать с большими объёмами данных Абриаль показал на примере Тезауруса английского языка, который был полностью конвертирован в базу данных Абриаля.

После запуска программы появляется пустое рабочее поле, в котором по ходу работы будут появляться окна трех типов: гипертабличные окна  , древовидные окна  и окна текстового редактора 

Alex – технология лексического анализа (рис.5.17).

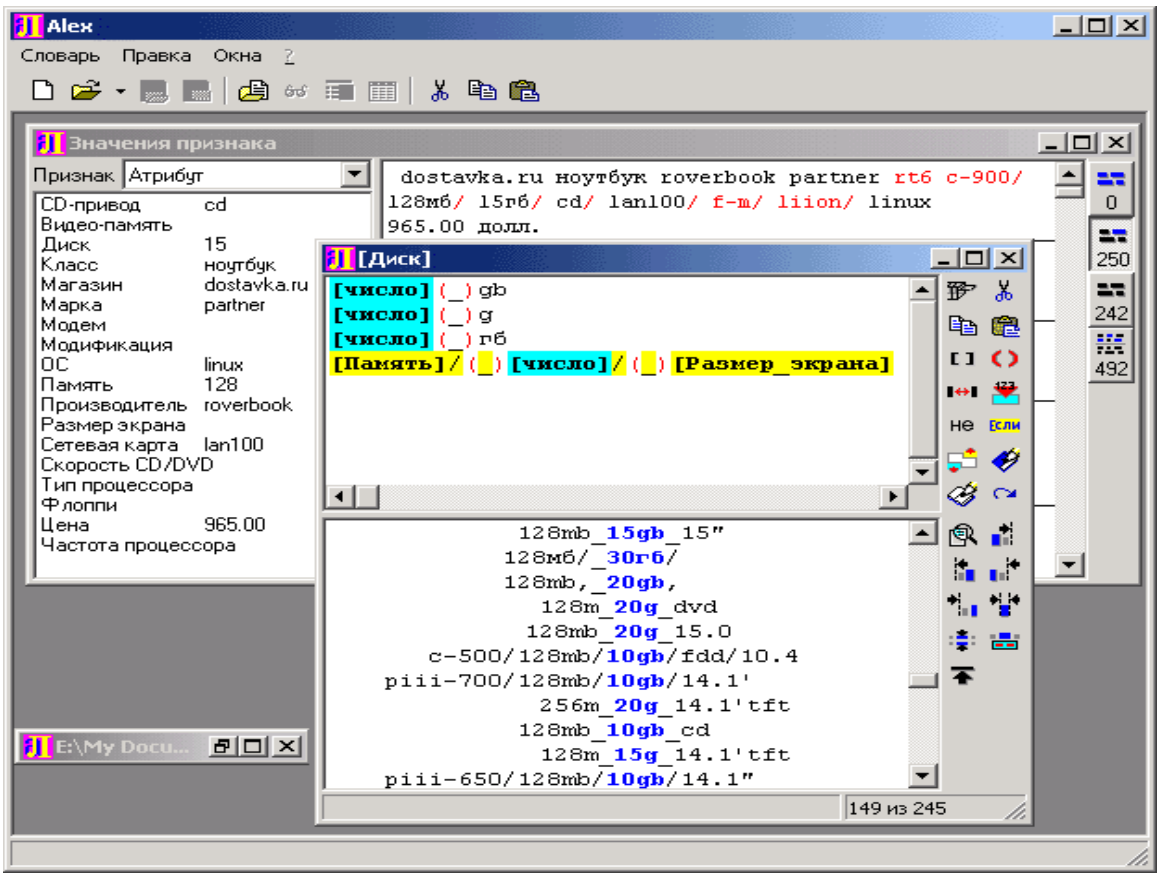


Рис.5.17. Визуальная среда ALEX

Технология лексического анализа позволяет с помощью настраиваемых лексических шаблонов произвольной сложности решать следующие задачи (рис.5.17):

- поиск в текстовых массивах различной степени структуризации определенных фрагментов, извлечение знаний;
- нормализация слабоструктурированных массивов данных как с точки зрения структуры, так и с точки зрения качества их наполнения.

Развитые поисковые системы, извлечение текстовых данных и знаний, некоторые типы анализа текста зависят от возможностей аппарата распознавания сложных лексических конструкций, который должен учитывать вхождения как самого образца, так не только словоизменительных, но и эквивалентных по содержанию (синонимичные) его вариантов.

Технология ALEX, обеспечивающая эти возможности, доказала свою эффективность на нескольких пилотных проектах и вышла на этап коммерческой реализации.

InBASE – естественно-языковая оболочка для распространенных СУБД и Интернета. Уникальная технология InBASE позволяет создать для прикладной базы данных интерфейс, понимающий произвольные запросы на *естественном языке* и обеспечивающий прямой доступ к данным для непрофессионального пользователя (руководителя, чиновника, рядового гражданина).

Оболочка легко переносится на любой естественный язык. В сочетании с системами распознавания голоса может послужить основой интерфейсов, понимающих устные запросы/сообщения/команды пользователя.

Технология ориентирована на широкое применение – внедрение естественно-языковых интерфейсов в деятельность предприятий и организаций любого типа и размера на всех уровнях административного управления, в интеллектуальных системах Интернет, в частности, в системах электронной коммерции.

На сайте проекта InBASE можно познакомиться с подробным описанием этой технологии, соответствующими публикациями и с несколькими демонстрационными версиями для различных областей приложений.

Естественно-языковой интерфейс к базам данных – что это такое? Естественно-языковой интерфейс является посредником между человеком и базой данных. Он переводит поступающие запросы на естественном языке в формальное представление, обращается с ним к базе данных и представляет результат в виде, пригодном для просмотра и анализа.

Иными словами, перевод произвольных запросов, которые на обычном естественном языке формулируются секунды, и традиционно для формализации и воплощения в отчеты требуют часы работы высококвалифицированных программистов, – этот перевод в технологии InBASE осуществляется автоматически за доли секунды.

- *Semp-T* – технология разработки сложных интеллектуальных систем на основе интегрированной модели представления знаний (рис.5.18).

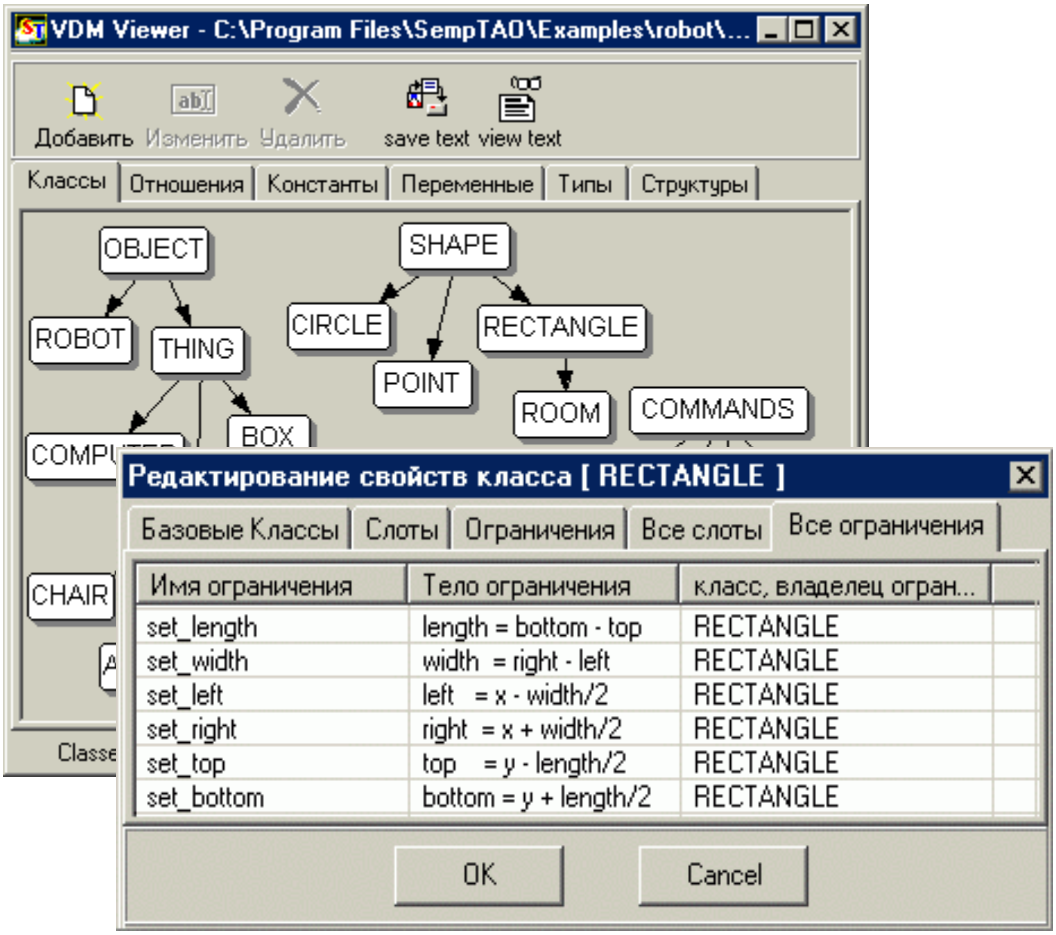


Рис.5.18. Архитектура программной среды SemP-T

Технология Semp-T объединяет уникальный по мощности комплекс средств и методов представления и обработки знаний, включающий:

- высокоуровневые средства задания семантики объектов предметной области путем спецификации ограничений на значения их параметров и локальных правил вывода;
- иерархическую семантическую сеть с определяемыми свойствами отношений;
- аппарат для работы с неточно заданными (недоопределенными) значениями числовых, символьных и множественных типов;

- динамические типы данных;
- развитый аппарат продукционных правил с двумя уровнями средств динамического управления;
- средства генерации и проверки гипотез;
- объектная графика и высокоуровневые средства создания пользовательских интерфейсов;
- визуальный интерфейс разработчика.

Semp-T естественно сочетает мощный логический вывод и вычисления над неточно заданными параметрами, что обеспечивает ее эффективное применение в таких областях:

- экспертные системы и их проблемно-ориентированные оболочки;
- интеллектуальные базы данных и знаний;
- сложные диагностические системы;
- системы планирования и принятия решений;
- моделирование процессов в технике, экономике, биологии и социологии;
- интеллектуальные системы управления сложными объектами, в том числе роботами;
- компьютерная поддержка учебных курсов искусственный интеллект, инженерия знаний и др.

Технология Semp-T ориентирована на конструктора интеллектуальных систем. Обеспечивает значительное повышение качества и многократное сокращение трудозатрат при создании сложных систем обработки знаний.

Экономика – технология нового поколения для построения макроэкономических моделей страны, региона, отрасли, корпорации.

Цель проекта «Экономика» – разработка на основе аппарата недоопределенных вычислений технологии создания компьютерной модели макроэкономики страны, региона, отрасли, корпорации, а также базирующегося на ней бюджета с обеспечением его корректировки и сопровождения. Представляет собой специализированную версию решателя UniCalc. Обеспечивает планирование структуры экономики и бюджета как на предстоящий год (в шкале квартал/месяц), так и на перспективу, охватывающую период до десяти и более лет.

При поддержке Совета Федерации была создана демонстрационная версия на 150 макропараметров экономики РФ. Использование этой технологии для создания математической модели экономики и бюджета делает возможными как оптимизацию стратегии экономического развития региона, отрасли или крупной организации, так и содержательную автоматизацию основных стадий формирования,

согласования и утверждения очередного бюджета, а также контроль на всех этапах его исполнения.

Не менее важной областью применения данной технологии является моделирование развития экономики на перспективу с определением влияния выбранной стратегии на взаимосвязь и ограничения, накладываемые на ключевые параметры экономики и бюджета на ближайшие годы.

Технология способна обеспечивать решение задач объемом во много тысяч отношений и параметров на интервале 10 и более лет. Учитывая прикладную важность этих задач и обеспечиваемый новой технологией качественный скачок в их решении, общий эффект ее использования может оказаться равным нескольким процентам общего объема бюджета.

ESWin 2.0 – программная оболочка-интерпретатор для работы с продукционно-фреймовыми экспертными системами. Программа предназначена для решения задач обратного логического вывода на основе фреймов и правил-продукций (рис. 5.19.).

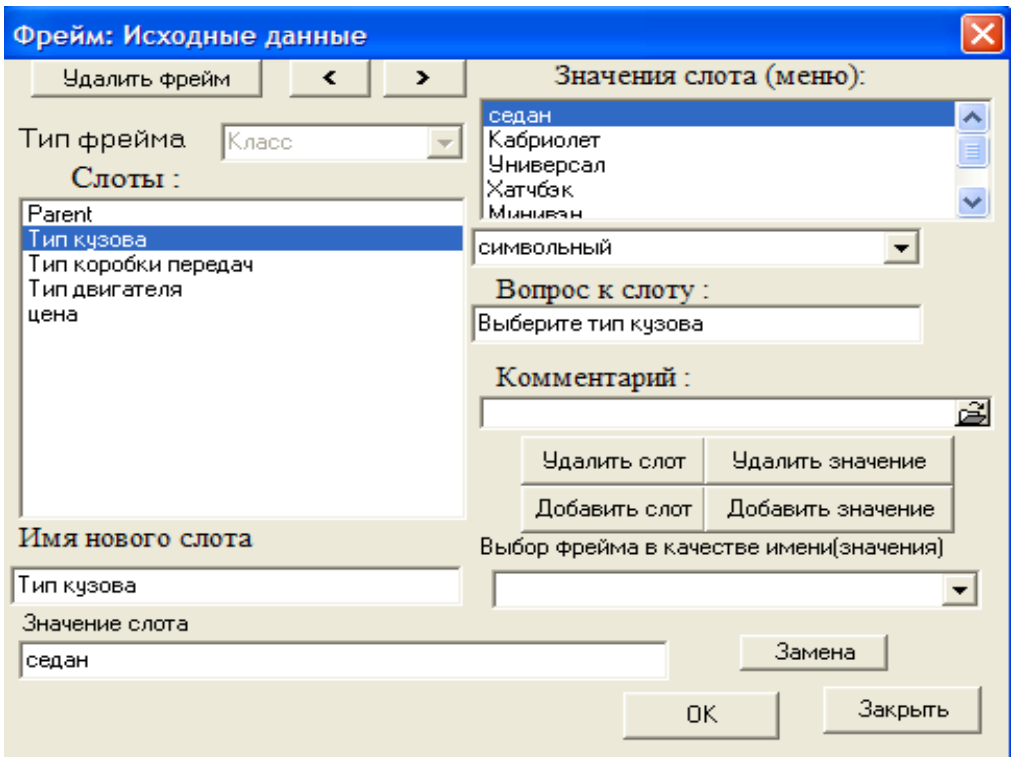


Рис. 5.19. Заполнение исходных данных в «ESWin 2.0»

Фреймом называется структура для описания стереотипной ситуации (события, объекта, понятия) состоящая из характеристик этой ситуации (события, объекта, понятия) и их значений. Характеристики называются слотами, а значения – значениями слотов.

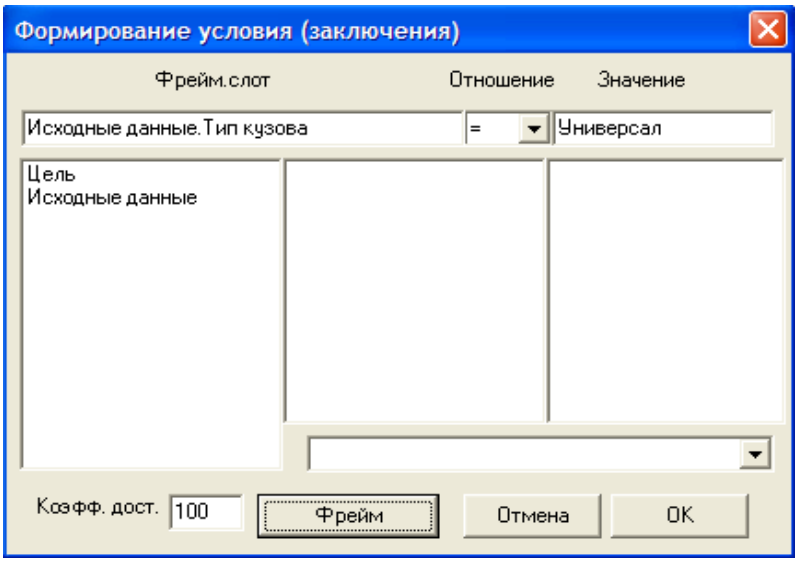


Рис. 5.20. Окно формирования условия

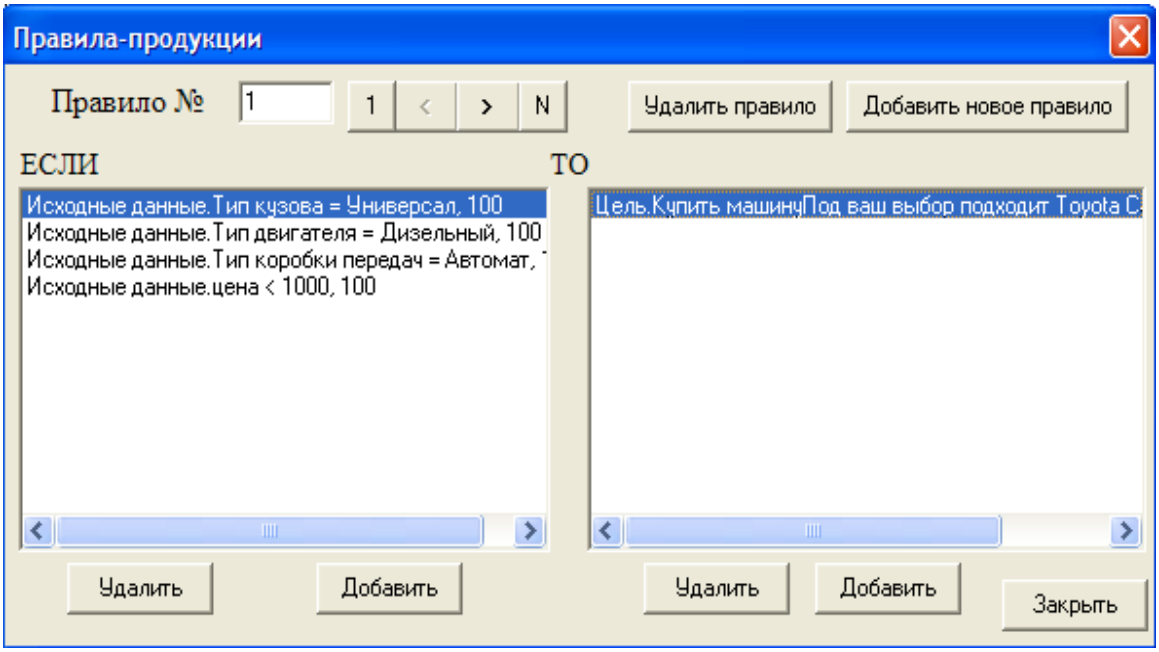


Рис. 5.21. Окно формирования правил-продукций

В пакете ESWin используются фреймы трех типов: фрейм-класс, фрейм-экземпляр и фрейм-шаблон. Слот фрейма-экземпляра имеет единственное значение, слот фрейма-класса и фрейма-шаблона имеет неограниченное число значений. Правила-продукции позволяют представить знания в виде предложений ЕСЛИ (условие)? ТО (заключение), где условие – это образец, по которому осуществляется поиск в базе знаний, а заключение – действия или процедуры, выполняемые при успешном исходе поиска (могут быть промежуточными, выступающими далее как условия, или целевыми,

завершающими работу системы и являющимися результатом решения задачи).

«Малая экспертная система» v 2.0 представляет собой простую экспертную систему, использующую байесовскую систему логического вывода. Она предназначена для проведения консультации с пользователем в какой-либо прикладной области (на которую настроена загруженная база знаний) с целью определения вероятностей возможных исходов и использует для этого оценку правдоподобности некоторых предпосылок, получаемую от пользователя.

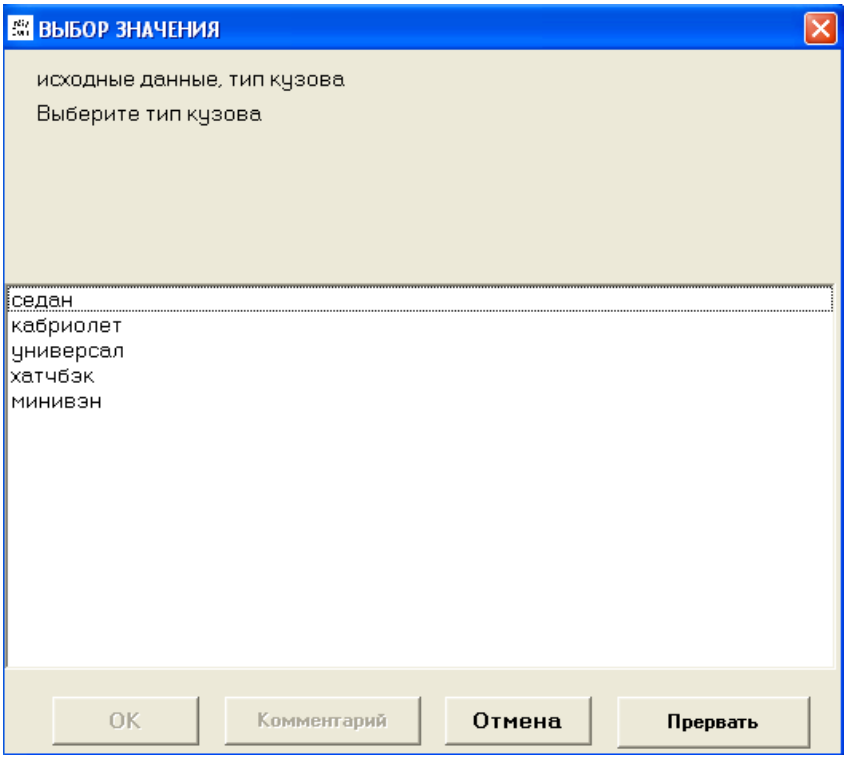


Рис. 5.22. Окно работы пользователя

База знаний представляет собой текстовый файл, включающий три секции со следующей структурой:

1. Описание базы знаний, имя автора, комментарий и т.п.

(можно в несколько строк, общая длина которых не должна превышать 10000 символов).

2. Свидетельство № 0 (любой текст (не более 1000 символов), заканчивающийся переносом строки).

Свидетельство № 1.

Свидетельство № 2.

3. Исход № 0, P [, i, Py, Pn].

Исход № 1, P [, i, Py, Pn].

Исход № 2, P [, i, Py, Pn],

где i – номер вопроса (свидетельства);

P – априорная вероятность данного исхода;

Py = P(E / H) – вероятность получения ответа «Да» на вопрос, если возможный исход верен;

Pn = P(E / неH) – вероятность получения ответа «Да» на вопрос, если возможный исход не верен.

Рис. 5.23. Окно редактора «ESWin 2.0»

И все же, несмотря на фантастические возможности ВМ, вероятно, они останутся лишь мощным инструментом. А мыслить и творить будет человек.

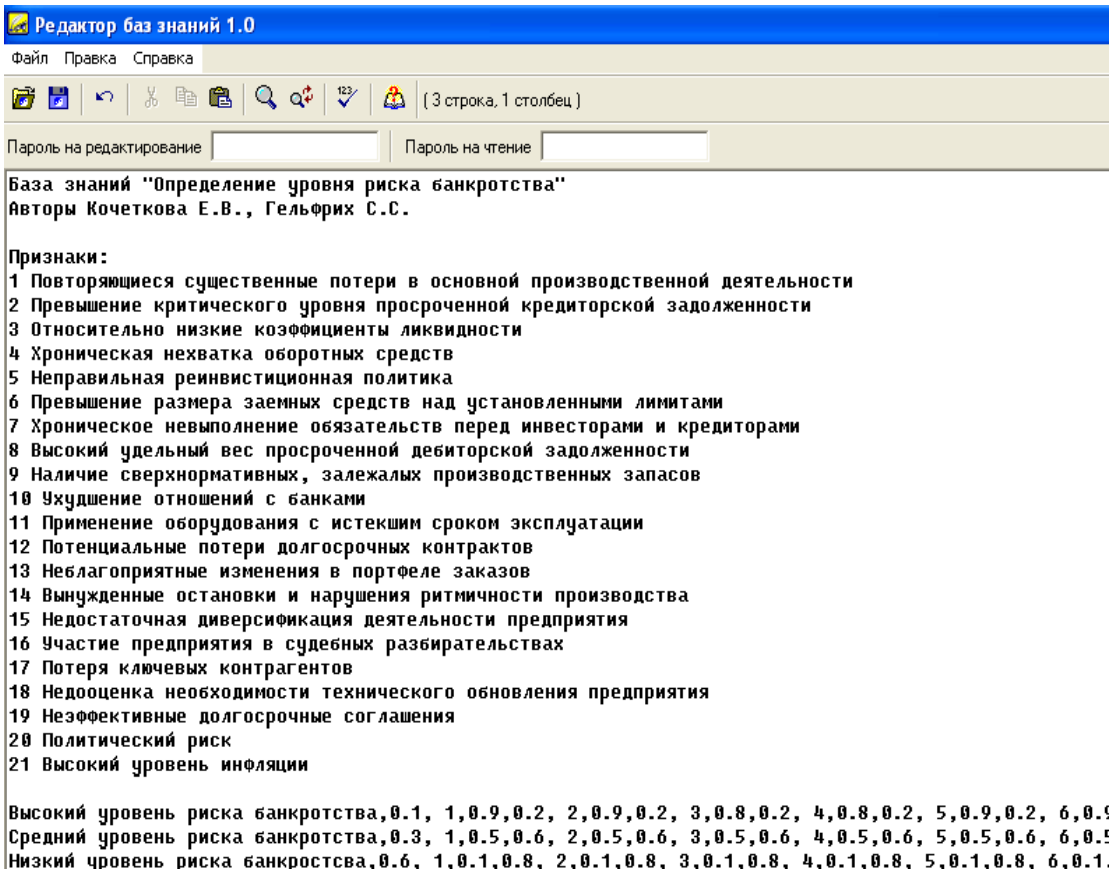


Рис. 6.24. Окно редактора базы знаний «Малой экспертной системы» v 2.0

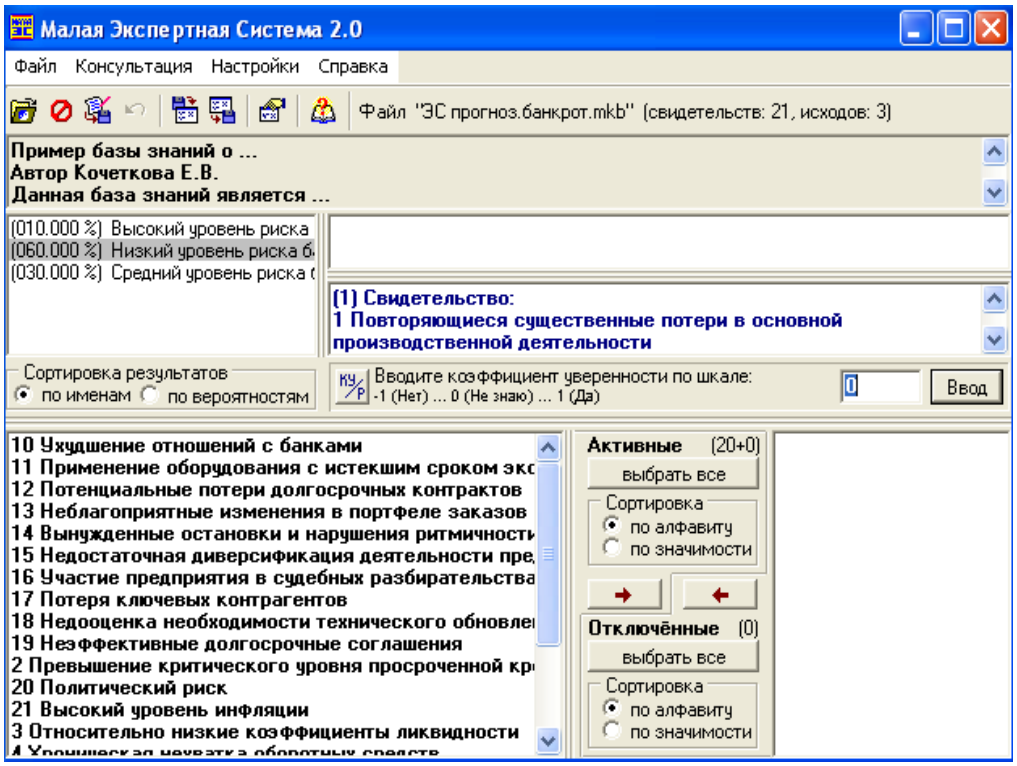


Рис. 5.25. Окно работы пользователя в «Малой экспертной системы» v 2.0

5.3. Программные средства профессионального уровня

Каждая прикладная программа этой группы ориентируются на достаточно узкую предметную область, но проникает в нее максимально глубоко. Так функционируют *АСНИ* – автоматизированные системы научных исследований, каждая из которых «привязана» к определенной области науки. *САПР* – системы автоматизированного проектирования, каждая из которых также работает в узкой области. *АСУ* – автоматизированные системы управления. *АРМ* – автоматизированное рабочее место. *АСУП ТП* – автоматизированные системы управления производством и технологическими процессами.

Системы автоматизированного проектирования

Деятельность научно-исследовательских институтов, конструкторских бюро, проектных организаций связана с разработкой новых технологий, устройств, приборов, конструкций. Проектирование сопровождается оформлением большого объема технической документации: чертежей, схем, планов.

Для облегчения труда конструкторов, проектировщиков, изобретателей и рационализаторов разработаны системы автоматизированного проектирования (САПР). Этому термину соответствует английская аббревиатура CAD – Computer–Aided Design. Эти три буквы входят в названия многих иностранных программ, предназначенных для конструирования, черчения, трехмерного моделирования объемных объектов и оформления инженерной документации, например, P-CAD, OrCAD, AutoCAD, CADdy, ArchiCAD, T-FLEX CAD.

В отличие от автоматических систем проектирования, САПР способны решать задачи, не поддающиеся полной формализации. Проектирование в таких системах является автоматизированным и осуществляется под непосредственным контролем пользователя, чаще всего в интерактивном режиме.

Наиболее широко системы автоматизированного проектирования используются в электронике, электротехнике, архитектуре, строительстве, машиностроении, автомобилестроении, нефтехимической промышленности, аэрокосмической технике. В ряде САПР из области электроники заложен принцип сквозного проектирования. При этом с помощью САПР выполняют полный цикл проектирования и производства: составление технического задания, разработку объекта, моделирование его работы, автоматизированное

изготовление объекта, оформление документации. По этой причине все чаще говорят о системе CAD/CAM (Computer–Aided Design/Computer–Aided Manufacturing) – системе автоматизированного проектирования и производства, которая охватывает широкий спектр задач от начального конструирования до подготовки данных, необходимых для реального производства изделия.

Существует большое число средств проектирования электронных устройств. Они позволяют автоматизировать разработку важного элемента радиоэлектронных устройств – печатных плат. Системы проектирования «умеют» автоматически размещать радиоэлементы на печатной плате, а затем составлять рисунок электрических проводников, соединяющих радиоэлементы в соответствии с принципиальной электрической схемой. На заключительном этапе проектирования с помощью САПР осуществляется *верификация* разработанной конструкции (производится окончательная проверка соблюдения технологических норм, рассчитываются характеристики устройства с учетом параметров, присущих конкретной конструкции, т. е. с учетом взаимного расположения радиоэлементов и печатных проводников).

Пакет *OrCAD* (фирма OrCAD Inc.) представляет собой интегрированный комплекс программ для разработки электронных устройств. Он состоит из нескольких автономных модулей: SDT, VDT, PCB, PLD.

Пакет *P-CAD* (фирма Accel Technologies) предназначен для разработки электронных устройств, начиная от составления принципиальной схемы, вплоть до подготовки документации на разработанное устройство.

Программа *AutoCAD* фирмы Autodesk является лидером среди инженерных графических пакетов. При построении чертежей система позволяет использовать графические примитивы. Система позволяет легко изменять масштаб изображения, производить панорамирование (плавное перемещение видимой части изображения по большому чертежу).

Детали, поэтажные планы здания и другие объекты можно прорисовывать разным цветом на различных слоях чертежа, что дает возможность отслеживать их совместимость и взаимосвязь при общей компоновке. AutoCAD «умеет» не только разрабатывать двумерные плоские чертежи, но и моделировать сложные трехмерные каркасные, полигональные (поверхностные) и объемные (твердотельные) конструкции. Система обеспечивает автоматическое построение

диметрической, изометрической, косоугольной и аксонометрической проекций агрегатов, узлов и деталей.

Программа 3D-Dream House Designer позволяет проектировать офис (дом, квартиру) в трехмерном виде. Вокруг созданного жилища можно обустроить приусадебный участок. Есть возможность заглянуть в созданный дом через окно и, наоборот, осмотреть ландшафт с балкона. Созданный дом можно оснащать мебелью, радиоаппаратурой, картинами, наклеивать на стены разнообразные обои.

Педагогические комплексы

Обучающие и тестирующие программы предназначены для получения новых знаний, для тестирования по различным дисциплинам, для приема экзаменов, зачетов и т.д. Пример обучающих систем: TeachPro Word, TeachPro Windows, 1С:Школа, 1С:Высшая школа, тестирующих программ Test, 1С:Репетитор, АИСТ, КОП, TestComplex.

E–LEARNING – обучение (или элемент обучения) при помощи современных компьютерных технологий, Internet.

К основным преимуществам E-learning обычно относят:

- отсутствие личной встречи с преподавателем (преимущество особенно актуально, если для личной встречи необходима командировка в другой город);
- возможность обучения в удобное время и в удобном учащемуся темпе.

Обычно для *E-learning* используются следующие средства: электронная почта, Internet-форумы, Internet-сообщества, видеолекции, видеоконференции, case study, онлайн-тестирование, онлайн-консультиро-вание, экспертные системы и т.п.

Наконец, еще раз подчеркнем не только условность предложенной выше классификации, но и наличие пересечений. Так, каждую конкретную экспертную систему вполне можно отнести к ППО профессионального уровня; принцип гипертекста реализован в ряде авторских систем и т.д.

5.4. Проблемно-ориентированные программные средства. Обзор программных продуктов ведущих фирм

Фирма «1С». Фирма «1С», основанная в 1991 году Борисом Нургалиевым, сегодня является бесспорным лидером по известности и тиражу продаж. «1С:Предприятие» – это специализированная объектно–

ориентированная система управления базами данных (СУБД), предназначенная для автоматизации деятельности предприятия (кадровый учет, расчет зарплаты, бухгалтерский учет, складской учет).

Фирма «1С» предлагает следующий набор программных продуктов: «1С:Бухгалтерия», «1С: Электронная почта», «1С:Торговля», «1С: Документооборот», «1С:АФС», «1С:Электронный справочник бухгалтера», «1С:Предприятие» и другие.

Основной особенностью системы «1С:Предприятия» является ее конфигурируемость. Конфигурация обычно поставляется фирмой «1С» в качестве типовой для конкретной области применения, но может быть изменена, дополнена пользователем системы, а также разработана заново.

Функционирование системы «1С» делится на два процесса – конфигурирование (описание модели предметной области средствами системы) и исполнение (обработку данных предметной области). Результатом конфигурирования является конфигурация, которая представляет собой модель предметной области. В процессе конфигурирования формируется структура информационной базы, алгоритмы обработки, формы диалогов и выходных документов. Информационная структура проектируется на уровне предусмотренных в системе типов обрабатываемых объектов предметной области (константы, справочники, документы, регистры, перечисления, журналы расчетов, бухгалтерские счета, операции, проводки).

Конфигурацией в системе «1С:Предприятие» называется совокупность трех взаимосвязанных составных частей:

- структуры метаданных;
- набора пользовательских интерфейсов;
- набора прав.

Структура метаданных. Метаданные представляют собой совокупность объектов метаданных (рис. 5.26). Под объектом метаданных в системе «1С:Предприятие» понимается формальное описание группы понятий предметной области со сходными характеристиками и одинаковым предназначением. На этапе конфигурирования система оперирует такими универсальными понятиями (объектами), как «Документ», «Журнал документов», «Справочник», «Реквизит», «Регистр» и другие. Совокупность этих понятий и определяет концепцию системы.

Набор пользовательских интерфейсов. Под пользовательским интерфейсом в системе «1С:Предприятие» понимается совокупность команд главного меню и панелей инструментов, настроенных на работу

с конкретными объектами данных – документами, справочниками, журналами и т.д. (рис. 5.27).

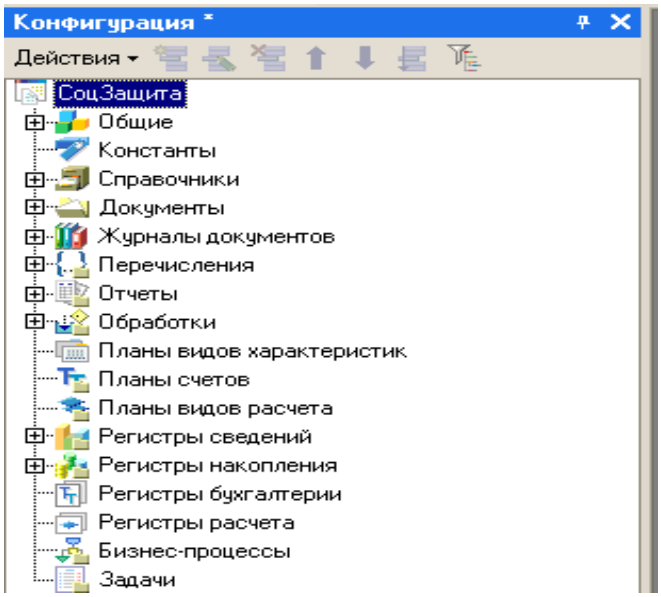


Рис. 5.26. Структура метаданных

Как правило, пользовательский интерфейс создается для конкретной категории пользователей. Цель создания интерфейса – обеспечить быстрый доступ пользователей к той информации, которая необходима им в соответствии с их обязанностями (рис. 5.28).

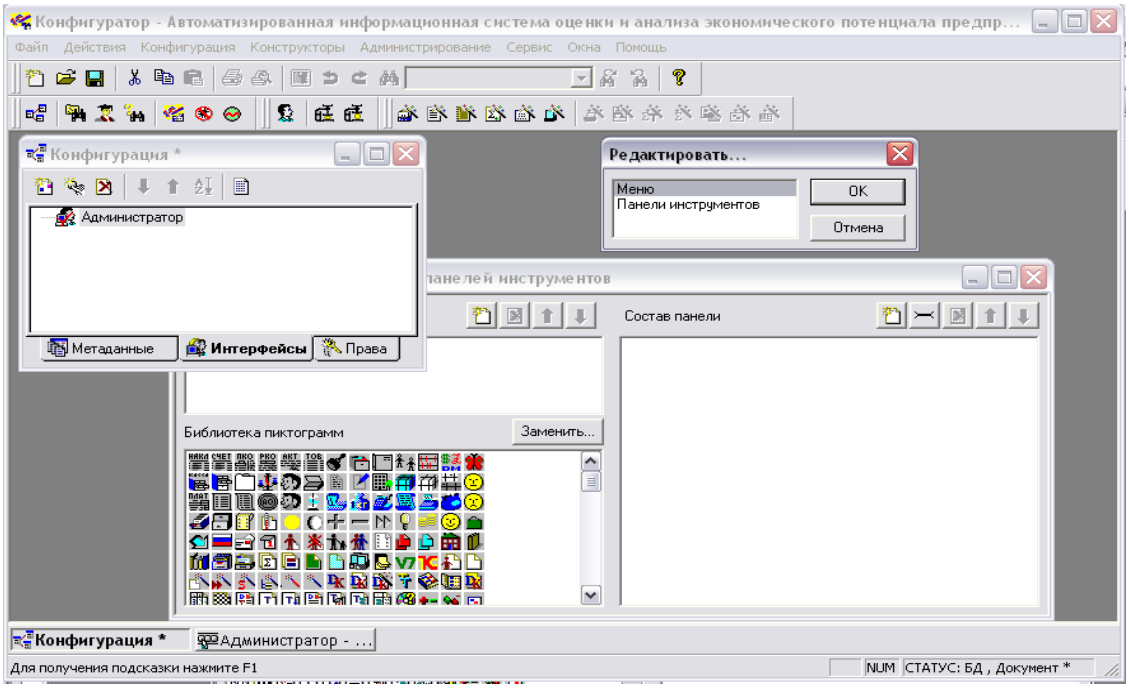


Рис. 5.27. Набор пользовательских интерфейсов

Набор прав. Под набором прав в системе «1С:Предприятие» понимается определение полномочий пользователей на работу с информацией, которая обрабатывается в системе.

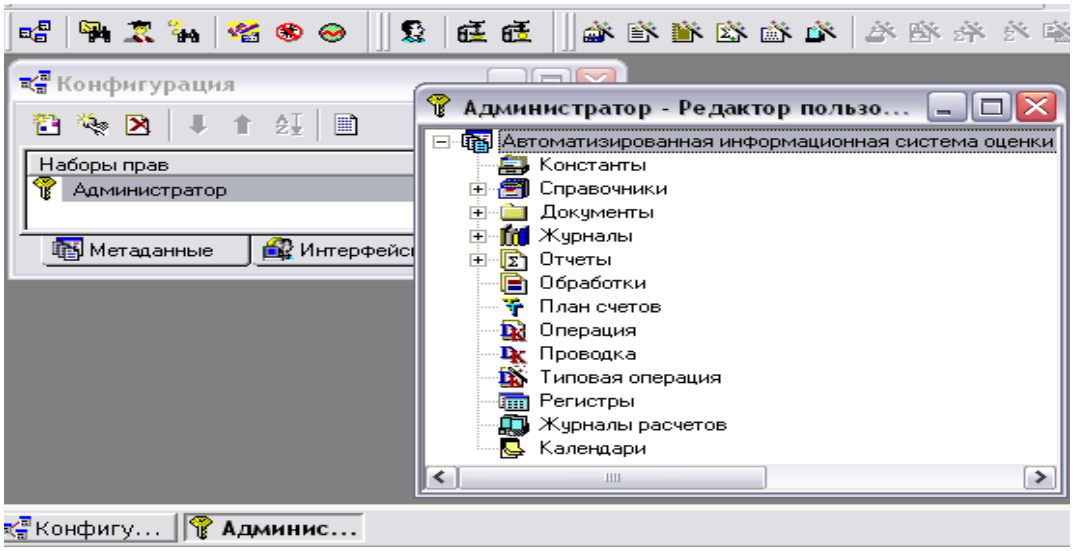


Рис.5.28. Набор прав

Визуальное представление данных в системе «1С». Большинство объектов метаданных в системе «1С:Предприятие» могут иметь визуальное представление. В самом лучшем случае визуальное представление состоит из следующих частей, представленных на рис. 5.29:

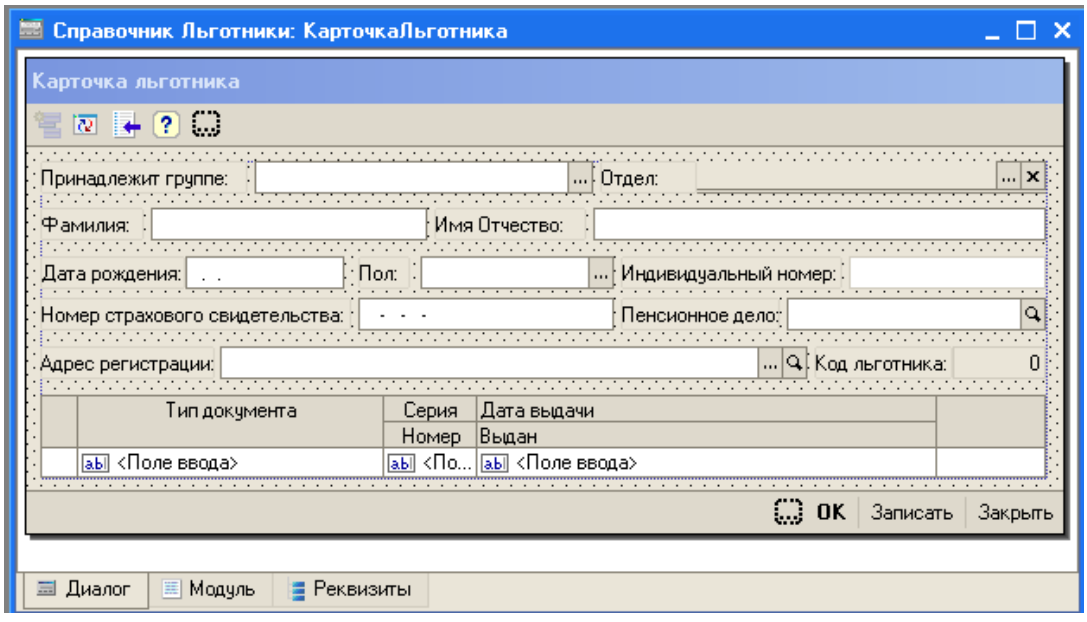


Рис. 5.29. Экранный диалог объекта метаданных

- экранный диалог, используемый для ввода и редактирования информации, хранящейся в объекте метаданных (рис. 5.29, на примере предметной области отдела социальной защиты администрации г. Юрги);
- печатная форма объекта метаданных, пример на рис. 5.30 (печатных форм может быть несколько);

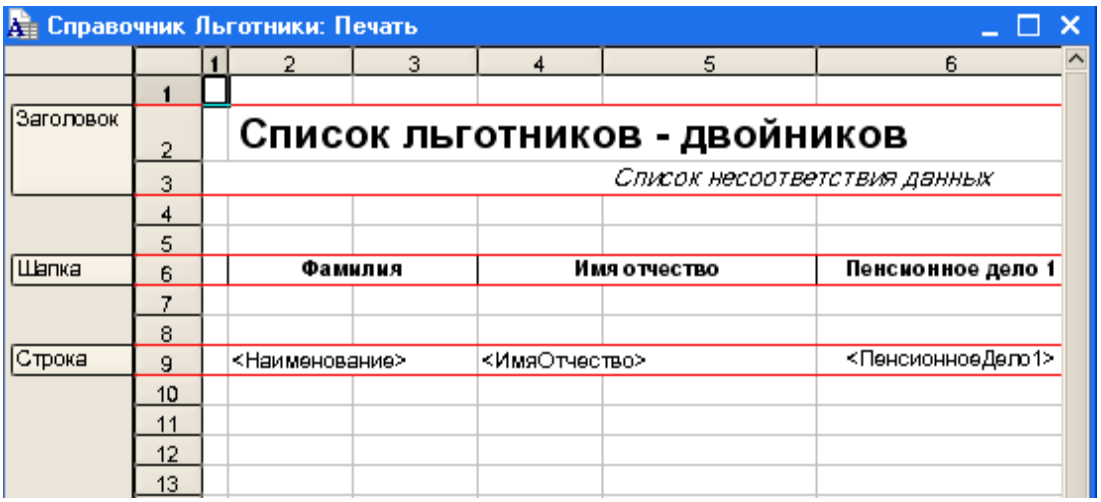


Рис 5.30. Печатная форма объекта метаданных

- модуль формы – программа на встроенном языке системы «1С:» (рис. 5.31.). Как правило, модуль формы содержит алгоритм построения печатной формы объекта метаданных, а также может выполнять обработку вводимой в диалог информации для целей входного контроля, выполнения расчетов и т.п.

Встроенный язык системы «1С:Предприятие» предназначен для описания (на стадии разработки конфигурации) алгоритмов функционирования прикладной задачи. Встроенный язык представляет собой предметно-ориентированный язык программирования, специально разработанный с учетом возможности его применения не только профессиональными программистами. В частности, все операторы языка имеют как русское, так и англоязычное написание, которое можно использовать одновременно в одном исходном тексте.

В процессе исполнения система уже оперирует конкретными понятиями, описанными на этапе конфигурирования (справочниками товаров и организаций, счетами, накладными).

При работе пользователя в режиме исполнения конфигурации обработка информации выполняется как штатными средствами системы, так и с использованием алгоритмов, созданных на этапе конфигурирования.

«1С» предлагает программные средства для автоматизации бухгалтерского учета «1С:Бухгалтерия», набор программных продуктов, таких как «1С:Электронная почта», «1С:Торговля», «1С:Документооборот», «1С:АФС», «1С:Электронный справочник бухгалтера», «1С:Предприятие» и др.

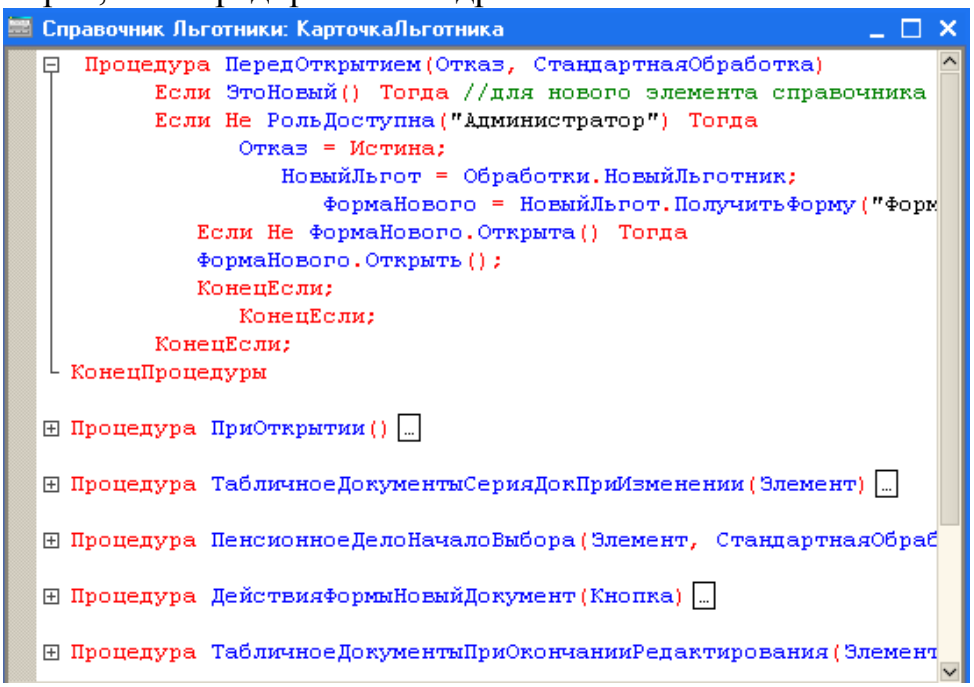


Рис. 5.31. Модуль формы объекта метаданных

Программный продукт «1С – ВИП Анатех: AB18.ABC». Управленческий учет и расчет себестоимости» – совместная разработка фирмы «1С» и российской консалтинговой компании «ВИП Анатех» на платформе «1С:Предприятие 8.0» является первым программным продуктом класса ABIS (Activity–Based Information System).

Электронная почта «1С:ЭП» – это интегрированная система, объединяющая в себе электронную почту (ЭП), БД и различные сервисные решения для большинства информационных задач (пакетная и диалоговая работа с ЭП, посылка сообщений, документов и т.п. индивидуальному адресату, группе абонентов, на доску объявлений или в другие почтовые системы). Кроме того, в системе «1С:ЭП» имеется широкий набор интегрируемых подсистем и функций, которые существенно расширяют возможности абонентов: доступ к БД (правовым, личным, офисным, отраслевым, общегосударственным и др.), диалоговый режим работы с ними и с почтовым ящиком, шлюз входа в сеть «Релком». Система проста в установке и удобна в эксплуатации, например, она позволяет разослать за ночь в

автоматическом режиме без участия человека 1000 факсов различным адресатам.

Пользователи могут подключаться и к сети самой фирмы «1С», и использовать огромные базы данных оперативно обновляемой правовой и коммерческой информации. Отличительной особенностью этих баз данных является то, что способы работы с ними ориентированы на пользователей, слабо разбирающихся в компьютерной технике.

Пакет «1С:Торговля» представляет собой гибкую универсальную систему автоматизации учета в торговле, складском хозяйстве и смежных областях деятельности предприятия. Пакет создан на базе новой технологии. Пользователи могут быстро расширять возможности работы (учет комплектации изделий/заказов, услуг и издержек обращения, выписка доверенностей, работа со счетами-фактурами и др.) без изменения программы. Есть возможность подключения торгового оборудования (сканеров штрихкодов, недорогих отечественных кассовых аппаратов).

Программа «1С:АФС» представляет собой специальную версию программы «Анализ финансового состояния предприятий», разработанную фирмой «ИНЭК» специально для семейства бухгалтерских программ «1С». Исходной информацией для анализа служат данные годовой, квартальной или месячной бухгалтерской отчетности предприятий любой организационно-правовой формы, включая предприятия с иностранным участием. Программа рассчитывает свыше 90 показателей, автоматически формирует аналитический баланс-нетто, освобожденный от статей, искажающих реальное финансовое состояние предприятия. Встроенная графика позволяет наглядно представить динамику данных или данные на определенную дату.

Система программ «1С:Предприятие» предназначена для комплексной автоматизации экономической деятельности предприятий различных направлений деятельности и форм собственности. Она позволяет организовать в единой системе эффективный бухгалтерский, кадровый, оперативный торговый учет, а также расчет заработной платы.

В данную поставку входят компоненты «Бухгалтерский учет», «Оперативный учет» и «Расчет», работающие в единой конфигурации.

Компонент *«Расчет»* предназначен для расчета заработной платы и ведения кадрового учета и обеспечивает:

- автоматизацию расчета начислений и удержаний по любым алгоритмам;
- проведение расчетов «задним» числом;

- формирование расчетных листков любого вида;
- расчеты как индивидуальных, так и групповых начислений типа бригадных нарядов;
- формирование платежных ведомостей с упорядочиванием информации по разным критериям с разбиением ее по категориям, подразделениям и другим признакам;
- расчет больничных листов, отпусков, оплаты по среднему заработку на основе данных за прошлые расчетные периоды;
- полный расчет заработной платы как по месячному, так и по недельному циклу;
- стандартные отчеты для налоговой инспекции и Пенсионного фонда РФ;
- ведение штатного расписания предприятия;
- распределение задачи ввода исходной информации и расчета между кадровиком и расчетчиком;
- статистическую информацию по сотрудникам предприятия;
- фиксацию кадровых перемещений сотрудников и их продвижения по службе;
- создание отчетов по кадровым перемещениям сотрудников и их продвижению по службе.

«1С:Предприятие 7.7» (сетевая версия). Данный продукт предназначен для одновременной работы нескольких пользователей в единой информационной базе.

«1С:Предприятие 7.7 + MS SQL Server 7.0» *(5 пользователей).* Основное отличие данного продукта состоит в том, что он обеспечивает возможность хранения таблиц базы данных на специализированном сервере (*MS SQL Server 7.0*), в результате чего можно достичь большей надежности хранения данных, свести к минимуму риск их повреждения или потери в случае возникновения неполадок в работе компьютерной сети, аварий источников питания и т.п., а также уменьшить простои системы, вызванные упомянутыми причинами. Кроме того, при одновременной работе большого числа пользователей с большими объемами данных улучшаются показатели производительности системы.

В комплект поставки входит также информационная база ***«1С:Гарант – Правовая поддержка»,*** издаваемая совместно фирмами «1С» и «Гарант-Сервис» и интегрированная с «1С:Предприятием 7.7». Это подразумевает контекстно-зависимую систему помощи для бухгалтера и специальный справочник нормативных актов, т.е. на любом этапе работы пользователь может просмотреть правовые документы, касающиеся конкретной хозяйственной операции.

«1С:Предприятие 8.0». В программе реализованы простые и эффективные решения по организации управленческого учета, основывающиеся на глубоком предварительном анализе бизнес-процессов на вашем предприятии, особенностях вашего бизнеса и ваших потребностях.

На сайте absolut1c.ru приведено описание типовых конфигураций программ «1С:Предприятие 8».

- «1С:Бухгалтерия предприятия 8»;
- «1С:Зарплата и Управление персоналом 8»;
- «1С:Управление торговлей 8»;
- «1С:Управление производственным предприятием 8».

Введен универсальный механизм обмена данными:

- для создания территориально распределенных систем (на основе «1С:Предприятие 8»);
- для организации обмена данными с другими информационными системами (не основанными на «1С:Предприятие 8»).

Пример интерфейса программы представлен на рисунке 5.32.

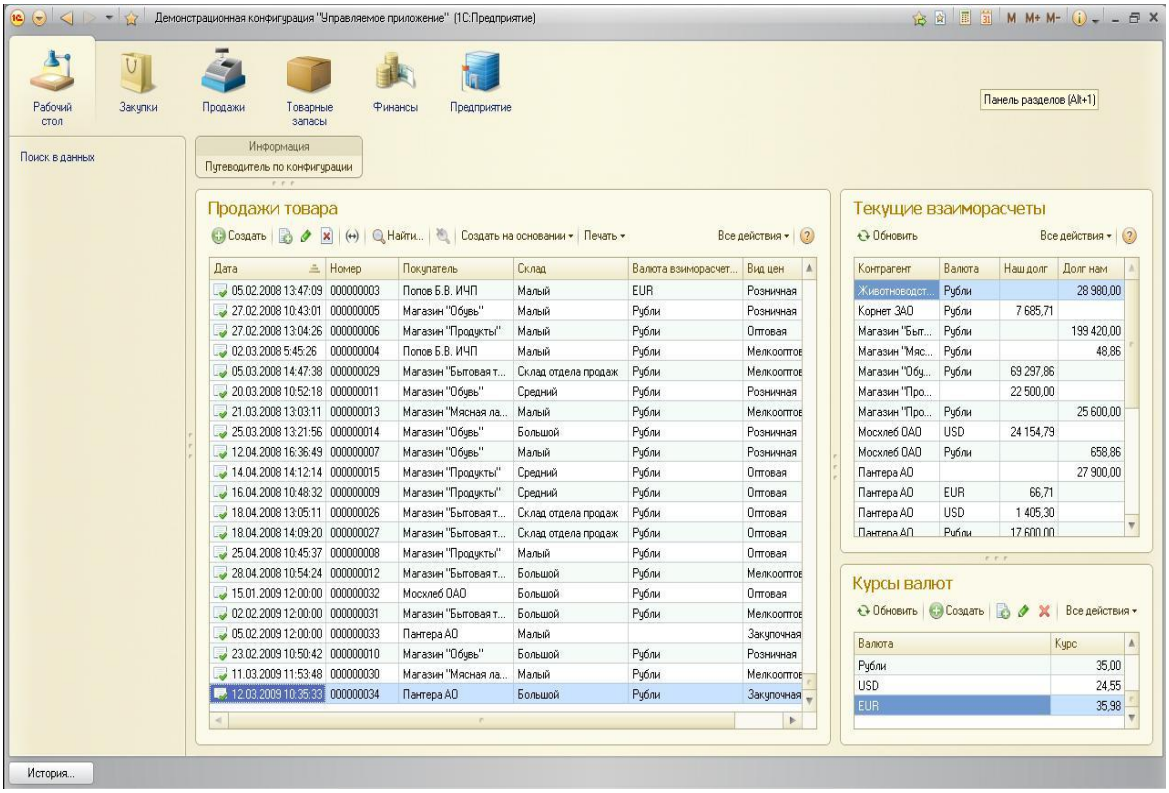


Рис. 5.32. Интерфейс 1С:Предприятие 8

Этот механизм позволяет перебрасывать только данные «1С:Предприятия». В качестве формата обмена используются известные XML документы. При обмене данными между информационными

базами «1С:Предприятие 8» не накладывается ограничений на идентичность конфигурации и структуры конкретных объектов (т.е. необязательно, чтобы конфигурации были одного релиза или одной компоненты, скажем, бухгалтерия с зарплатой). Благодаря введению новых объектов конфигурации «1С:Предприятие 8» (план обмена), одна информационная база может входить в состав нескольких схем обмена, реализующих различные стратегии обмена информацией между базами данных.

«1С:Бухгалтерия 8» – универсальная программа массового назначения для автоматизации бухгалтерского и налогового учета, включая подготовку обязательной (регламентированной) отчетности. Она представляет готовое решение для ведения учета в организациях, осуществляющих любые виды коммерческой деятельности: оптовую и розничную торговлю, комиссионную торговлю (в том числе субкомиссию), оказание услуг, производство и т.д.

Бухгалтерский и налоговый учет реализованы в соответствии с действующим законодательством Российской Федерации. «1С:Бухгалтерия 8» обеспечивает решение всех задач, стоящих перед бухгалтерской службой предприятия, а также складской учет, учет торговых операций, учет операций с денежными средствами, учет расчетов с контрагентами, учет основных средств и нематериальных активов, учет производства, учет хозяйственной деятельности нескольких организаций в единой информационной базе 1с v8 и др.

«1С:Бухгалтерия 8», благодаря возможности учета деятельности нескольких организаций в единой информационной базе, может использоваться как в небольших организациях, так и в холдингах со сложной организационной структурой.

«1С:Предприятие 8.1». Современный бизнес не стоит на месте и поэтому его развитие требует новых форм автоматизации типовых задач учета и управления предприятиями. Одной из основных целей создания «1С:Предприятия 8.1» являлось внесение существенных архитектурных изменений без изменения "видимой" функциональности технологической платформы. Это необходимо для того, чтобы более полно соответствовать современным представлениям о построении корпоративных систем, заложить фундамент будущего развития системы и проверить новые подходы к построению системы «1С:Предприятие».

Главными направлениями развития новой версии стали:

- повышение масштабируемости работы системы; работа с новыми платформами;

- включение принципиально новых возможностей по анализу и поиску информации;
- использование современных подходов к интеграции; повышение удобства администрирования системы.

Версия разработана на основе анализа опыта использования «1С:Предприятия 8» партнерами и пользователями. При разработке версии 8.1 учтены многочисленные пожелания и предложения специалистов, осуществляющих внедрение и разработку прикладных решений.

Новые возможности:

- кластер серверов «1С:Предприятие» по сравнению с кластером серверов «1С:Предприятие 8.0» обеспечивает более высокую надежность, масштабируемость и эффективность использования аппаратных ресурсов. Кластер серверов может предоставлять один или несколько рабочих процессов, для обслуживания клиентских соединений. Рабочие процессы кластера могут функционировать на одном или нескольких компьютерах;
- кластер серверов «1С:Предприятие 8.1» работает как под управлением операционной системы Microsoft Windows, так и под управлением операционной системы Linux;
- работа с СУБД PostgreSQL. СУБД PostgreSQL может работать под управлением операционных систем Microsoft Windows или Linux. Используется модифицированная версия PostgreSQL, дистрибутив включен в комплект поставки;
- система компоновки данных является принципиально новым механизмом, основанным на декларативном описании и предназначена для построения отчетов, а также вывода информации, имеющей сложную структуру;
- отчет в конфигурации 1С может быть описан декларативно, без написания кода на встроенном языке и создания форм. Процесс создания готового отчета может быть разбит на несколько этапов и выполняться по отдельности. Так, построение отчета может выполняться отдельно от его вывода;
- реализован конструктор настроек компоновки данных. Использование конструктора существенно упрощает и облегчает настройку отчетов пользователями;
- реализованы новые возможности построения отчетов;
- механизм фоновых заданий (позволяет инициализировать выполнения процедур общих модулей асинхронно (без ожидания завершения);

- механизм регламентных заданий (позволяет организовать автоматический вызов процедур общих модулей по расписанию);
- механизм Web-сервисов позволяет создавать Web-сервисы в конфигурации «1С:Предприятия 8.1», а также взаимодействовать в конфигурации «1С:Предприятия 8.1» с веб-сервисами, опубликованными сторонними поставщиками. Реализована возможность публикации Web-сервисов на веб-сервере (MS IIS или Apache2) через файловую систему и по протоколу FTP;
- механизм XDTO (XML Data Transfer Objects). Это механизм объектного моделирования данных, описываемых с помощью схемы XML. Механизм XDTO используется для взаимодействия с различными внешними источниками данных и программными системами, а также для работы с Web-сервисами;
- режим управляемых блокировок в транзакции. Позволяет управлять блокировками данных и повышает параллельность работы пользователей;
- механизм полнотекстового поиска в базе данных. Поддерживает указание поисковых операторов (И, ИЛИ, НЕ, РЯДОМ и др.). Возможно выборочное включение прикладных объектов и реквизитов в полнотекстовый поиск;
- существенно переработан механизм отладки;
- улучшенная эргономика и удобство работы пользователей;

– расширена связь с функциями Microsoft Office:

1) реализован механизм событий, позволяющий в прикладных решениях предоставить пользователю удобную возможность редактирования документов с помощью таких приложений, как Microsoft Word, Microsoft Excel;

2) реализована возможность сохранения табличного документа в формате Microsoft Excel. При этом поддерживается корректное сохранение таких элементов оформления табличного документа, как строки с различной шириной колонок, группировки строк и колонок, примечания, разрывы страниц, содержимое ячеек длиной более 255 символов, фоновый рисунок и др.;

3) реализовано запоминание в буфер обмена числовых значений, позволяющее использовать их в различных программах (например, Microsoft Excel, калькулятор Microsoft Windows), а также в формульном калькуляторе и табло «1С:Предприятия».

При выборе системы автоматизации требуется принять решение о разделении различных подсистем автоматизации или, наоборот, о централизации путем внедрения комплексного решения. Современные

тенденции развития экономических систем и мировой опыт показывают, что универсального рецепта для решения этой проблемы не существует.

Использование обособленных решений проще и эффективнее, если отдельные задачи автоматизации на предприятии мало пересекаются. Комплексные решения эффективнее при сильной увязке различных задач автоматизации и готовности предприятия к формированию единого информационного пространства. Система программ «1С: Предприятие 8» предоставляет возможность реализации обоих подходов: как внедрение комплексного решения, так и внедрение отдельных прикладных решений, которые будут работать автономно, или интегрировано с другими решениями «1С».

Фирма «Интеллект-Сервис». Спектр и качество продукции этой фирмы, комплексный подход к проблемам удовлетворяют запросы самых малых, средних фирм (программы БЭМБИ+, БЭСТ) и крупных компаний (БЭСТ–3, БЭСТ–4). Среди пользователей программных средств этой фирмы – торговые и страховые компании, промышленные предприятия и строительные фирмы, бюджетные организации, издательства, инвестиционные компании, фонды и др.

БЭСТ – это комплексная система автоматизации оперативного и бухгалтерского учета. Гибкая технология автоматизированного учета позволяет объединить в единой системе специализированный программный продукт, компьютерную сеть, кассовые аппараты, устройства считывания штрих-кодов и другое оборудование.

Система БЭСТ-4 – развитие системы БЭСТ-3. В отличие от предыдущих программных комплексов БЭСТ-4 поставляется в двух вариантах: в конфигурации «файл-сервер» и конфигурации «клиент-сервер». Применение конфигурации «клиент-сервер» рекомендуется для организаций с числом рабочих мест более десяти. При этом значительно повышаются характеристики целостности баз данных при увеличении быстродействия их обработки.

Для эксплуатации системы БЭСТ-4 требуются рабочие станции с такими же техническими характеристиками, как и для системы БЭСТ-3. Допускается использование различных операционных систем: MS DOS, Windows. В архитектуре «файл-сервер» программа может эксплуатироваться в локальных сетях на базе Windows NT, Nowell Net Ware и т.п.

Система БЭСТ-4 отличается развитыми функциями учета движения товаров на складе и в торговом зале, обеспечивает работу со счетами-фактурами и автоматический партионный учет и продажу товаров комплектами. Имеет генератор отчетов, возможность формирования большого количества различных встроенных отчетных форм (около 300

видов) и наглядного представления данных в графическом виде, возможность обмена данными с банковскими системами, расчета фактической себестоимости расходных материалов методами FIFO, LIFO, средних цен, возможность анализа финансового состояния и др.

В то же время поддерживаются и все прочие разделы бухгалтерского учета: расчет зарплаты, учет основных средств, материалов (малоценных и быстроизнашивающихся предметов), работа с расчетными и валютными счетами в банке, формирование баланса и документов публичной отчетности и т.д.

Руководителю система предоставляет широкие возможности по оперативному контролю и управлению магазином. Например, она позволяет оперативно определить номенклатуру товаров, пользующихся наибольшим спросом, выяснить, сколько их осталось в наличии и у какого поставщика наиболее выгодно купить очередную партию этого товара.

В системе БЭСТ-4 введен полноценный бухгалтерский учет по забалансовым счетам. На забалансовых счетах, так же как и на остальных счетах, может быть открыт аналитический учет, получены любые отчеты и справки. Создан новый АРМ «Управление закупками», предоставляющий пользователю множество дополнительных возможностей. Ведется реестр счетов кредиторов. Можно выписывать счета в валюте и отслеживать взаиморасчеты даже в том случае, если платежи проводятся в рублях. К счетам кредиторов «привязаны» товарные документы поставщиков и подрядчиков – счета-фактуры. При вводе счетов-фактур поддерживаются справочники товаров, материалов и услуг. На основе реестра счетов-фактур формируются отчеты по книге покупок. На основе счетов-фактур можно автоматически формировать приходные документы на складах. При этом предусмотрена возможность неполного учета, т.е. пользователь может указать количество брака и недостачи в поставке и автоматически генерировать новые возможности в АРМ «Товары. Готовая продукция». Счет-фактуру можно автоматически сформировать и на основе документа складского учета на реализацию товара. Во всех блоках складского учета предусмотрена возможность ведения партионного учета запасов на складе.

Для партионного учета предусматривается ведение параллельного учета, как в национальной валюте, так и в валюте поставки. Обеспечивается учет себестоимости по средним ценам на уровне номенклатурного номера. В АРМ «Торговый зал» появилась возможность ведения для каждого магазина своего прайс-листа. Можно построить отчеты за каждую дату в тех ценах, в которых реально

продавался товар. В системе БЭСТ пользователь имеет возможность учитывать один и тот же товар на разных счетах учета. Для пользователей, ведущих одновременно учет товаров на складе и в магазине, предусматривается разделение учета: по фактическим ценам – на складе и по учетным (по ценам реализации) – в розничной торговле.

Специализированная программа «Финансовый анализ» обеспечивает оперативный анализ товарных потоков и издержек обращения. Руководитель получает мощный инструмент, позволяющий проанализировать оптимальность состава запасов товаров, их количество и периодичность пополнения, выявить группы наиболее прибыльных товаров, оценить тенденции изменения прибыли в зависимости от различных факторов (например, от изменения цены, себестоимости закупок). Программа предоставляет возможность оценить динамику издержек обращения в сравнении с динамикой товарооборота, определить резервы по сокращению этих издержек.

Система «БЭСТ-ОФИС» предназначена для ведения оперативного и бухгалтерского учета на предприятиях малого бизнеса. Программа ориентирована именно на управление и позволяет перейти от интуитивного подхода при ведении дел к регулярному менеджменту. Это поможет снизить издержки, увеличить объемы продаж и, в конечном итоге, повысить прибыльность предприятия. Система позволяет:

- полностью контролировать финансовые потоки;
- управлять товарными запасами предприятия;
- регулировать отношения с бизнес-партнерами;
- формировать необходимую бухгалтерскую и налоговую отчетность.

В состав системы «БЭСТ-ОФИС» входят следующие подсистемы:

- учет кадров;
- учет финансов;
- учет взаиморасчетов;
- управление запасами;
- управление продажами;
- налоговый учет;
- учет заработной платы;
- учет имущества.

Система реализована средствами MS Visual Studio 6.0, в качестве СУБД используется MS Access 2000.

Фирма «Инфософт». Эта фирма успешно работает в промышленной сфере, предлагает широкий спектр программных средств, в

том числе с исходными текстами: бухгалтерские системы, программные комплексы для автоматизации управления предприятиями, строительными организациями, программы для бюджетных организаций и др.

Программа «Многовалютная финансовая бухгалтерия» предлагается для небольших фирм, только постигающих азы автоматизации учета. Это недорогая программа, снабженная доходчиво составленным учебным пособием «Финансовая бухгалтерия». Она работает во многих организациях.

«Интегратор» – новая многопользовательская сетевая бухгалтерская система, вобравшая в себя последние достижения фирмы. «Интегратор» – многопользовательская компьютерная бухгалтерия, программный продукт нового поколения, отвечающий современным требованиям. Эта система изначально проектировалась как сетевая, она построена в архитектуре «клиент-сервер» и предназначена не только для предприятий, впервые приступающих к автоматизации, но и для тех, кто не удовлетворен результатами работающих у них компьютерных комплексов.

«Интегратор» содержит все необходимые средства для автоматизации бухгалтерского учета предприятия, обеспечивает полноту функций и требуемый уровень детализации учета на каждом участке. Система строится в единой информационной среде, где все пользователи имеют доступ к общей информации в режиме реального времени: новые или измененные данные, введенные на одном рабочем месте, сразу же могут использоваться персоналом, работающим на других компьютерах.

Программа поддерживает сложившееся на предприятии разделение бухгалтерии на участки учета, имеет средство разграничения доступа к информации, поддерживает суверенность и ответственность каждого исполнителя, устанавливает необходимые взаимосвязи для обмена данными, обеспечивает согласованную работу бухгалтеров, обладает средствами гибкой настройки входных и выходных документов, автоматизирует процедуры формирования проводок и выполнение расчетов на основе использования типовых хозяйственных операций.

Пользователь самостоятельно может определить количество и названия участков учета, удалить из базовой поставки одни участки и ввести новые, организовать на каждом из участков любое количество рабочих мест. На реальных объектах, эксплуатирующих систему, можно встретить настройку как на десятки участков учета, что характерно для больших предприятий, так и на одно–два рабочих места, что вполне достаточно для небольших фирм.

«Интегратор» обеспечивает суверенность работы персонала: каждый бухгалтер отвечает за достоверность информации по счетам своего участка. Когда бухгалтер вводит в компьютер хозяйственные операции, сальдо и обороты пересчитываются только по тем счетам, которые относятся к его участку. Проводки вводимой операции, затрагивающие счета других участков, остаются отложенными до тех пор, пока их не подтвердят бухгалтеры смежных участков.

В системе применен удобный механизм отработки спорных ситуаций, что создает хорошие условия для обеспечения согласованной работы большого количества бухгалтеров. В части аналитического учета для каждого балансового счета пользователь устанавливает требуемый уровень глубины аналитики. Например, для материальных ценностей и товаров можно установить ведение натурального и стоимостного учета по складам и материально ответственным лицам; для взаиморасчетов – ведение учета по организациям и договорам и т.д. В системе предусмотрена возможность ввода новых аналитических параметров и показателей, не включенных в базовую поставку.

Система обеспечивает автоматическое формирование проводок и выполнение расчетов, есть возможность включения в типовые хозяйственные операции любого количества проводок, в том числе циклически повторяющихся. При создании новой типовой операции используются маски ввода для копирования повторяющейся в проводках аналитики, а также разнообразные типовые алгоритмы расчетов. Существуют средства создания новых алгоритмов. «Интегратор» позволяет модифицировать старые и создавать новые входные формы и выходные документы с помощью генератора отчетов.

Очень существенно и то, что версия 3.0 системы позволяет работать не только «от операции», но и «от документа». Причем пользователь может создавать любые собственные формы документов, устанавливать технологическую последовательность их обработки, настраивать систему на автоматическое формирование проводок. А это значит, что с помощью «Интегратора» можно теперь автоматизировать не только работу бухгалтерии, но и других служб, которые не имеют прямого отношения к бухгалтерским проводкам, таких, например, как финансовый отдел или отдел сбыта.

Для небольших фирм, где бухгалтерский учет ведут один–два бухгалтера, фирма «Инфософт» предлагает однопользовательский вариант системы «Интегратор». Он содержит практически те же функции и возможности, что и сетевой вариант, но работает на одном компьютере.

Фирма «КОМТЕХ+». Эта фирма известна сетевой версией программы «Комплексная планово-экономическая и бухгалтерская система» и ее различными вариациями поставок: бухгалтерская система, склад и анализ финансовой деятельности.

Основные отличия системы: гибкость настройки, автоматизация ведения многих трудоемких процессов и операций, наличие удобных механизмов распределения и анализа информации и др. В частности, пользователь может самостоятельно определить реквизиты – поля номенклатурного справочника; особенности формирования учетной или отпускной цены, состава и способов расчета реквизитов приходных и расходных документов; алгоритма автоматического распределения сумм, формирования новых проводок в журнале хозяйственных операций с пообъектным распределением сумм и др. Эта технология и возможности ПС позволяют бухгалтеру отобрать необходимые проводки, объединить по заданному признаку, распределить указанную или рассчитанную сумму пропорционально удельным весам сумм проводок.

Задачу контроля взаиморасчетов с контрагентами, как правило, вызывающих трудности при формировании финансовых результатов, помогает решать соответствующий модуль, предоставляющий необходимую аналитическую информацию для пользователей.

Модуль «Калькуляция изделий в производстве» обеспечивает эффективное решение задач, где требуется определение себестоимости изделий на основе цен входящих в них комплектующих. Большая потребность в решении так называемых калькуляционных задач обычно характерна для общепита, однако необходимость в автоматизации подобного типа задач имеется и в промышленности, и в строительстве и т.п.

Очень удобен механизм проведения групповых операций, что существенно облегчает и ускоряет работу пользователя; такое, к сожалению, нечасто можно встретить в бухгалтерских программах.

Фирма «АйТи». Это одна из немногих отечественных фирм, работающих в классе заказных финансовых систем в архитектуре «клиент-сервер», она имеет опыт реализации более 200 проектов различной сложности на базе собственной разработки *«Бухгалтерская офисная сетевая система» (БОСС).* Основная линия фирмы – комплексная автоматизация предприятий среднего масштаба и разработка информационных систем «под, ключ».

Программный комплекс содержит необходимый набор модулей для автоматизации бухгалтерского учета и управления предприятием. Отличие системы «БОСС» состоит в том, что для каждого заказчика

создается свой программный продукт, учитывающий его специфику и пожелания.

Фирма «Галактика». Эта российско-белорусская корпорация разработала одноименную корпоративную систему, успешно эксплуатирующуюся на сотнях средних и крупных предприятий России и ближнего зарубежья. Среди них торговые предприятия, предприятия сферы услуг, а также различных отраслей промышленности: машиностроительные, горнодобывающие, металлургические, нефтеперерабатывающие и многие другие.

Отличительной особенностью системы *«Галактика»* является комплексный подход к проблеме автоматизации, охватывающий все сферы управления современным предприятием, включая финансовое и хозяйственное планирование, управление кадрами, бухгалтерский учет, оперативное управление и др.

Концептуальная модель системы базируется на тщательно проработанной технологии компьютерного ведения бухгалтерского и оперативного учета, легко адаптируется под специфику различных типов предприятий. В основу архитектурного построения системы «Галактика» заложен принцип разделения комплексной системы автоматизации на ряд взаимосвязанных контуров:

- бухгалтерский учет;
- административное управление;
- оперативное управление;
- управление производством;
- управление автотранспортом;
- розничная торговля.

Рассмотрим назначение некоторых контуров.

Назначение *контура «Бухгалтерский учет»* понятно из его названия – с его помощью осуществляют обработку данных первичного учета и формируют всю необходимую финансовую отчетность предприятия. В этот же контур включен модуль по расчету заработной платы.

Контур «Административное управление» решает задачи финансового и хозяйственного планирования, учета и управления кадрами, организации электронного документооборота предприятия и т.п. В состав контура входят также модули управления маркетингом и анализа финансовой и хозяйственной деятельности.

Контур «Оперативное управление» предназначен в первую очередь для решения задач учета наличия и движения товарно-материальных ценностей, включая управление материально-техническим снабжением

и реализацией, а также для контроля взаиморасчетов с поставщиками и покупателями в соответствии с заключенными с ними договорами.

В системе «Галактика» имеется *контур «Управление производством»,* ориентированный на решение задач управления производственным процессом. Рассмотрим более подробно основные модули этого контура и их функциональные возможности.

Модуль «Технико-экономическое планирование» (ТЭП) является центральным в контуре «Управление производством». Основное назначение модуля – автоматизация формирования плана производства и производственных программ, расчет потребностей в материальных и трудовых ресурсах, калькуляция плановой себестоимости выпускаемой продукции. Отличительная особенность данного модуля – возможность его адаптации, осуществляемой на уровне настройки программы, к различным типам предприятий и методам планирования. Например, план производства можно составлять на основе результатов прошлого года, по сумме договоров на поставку продукции, по сумме производственных заказов и т.п. Производственная программа по цехам может формироваться (с учетом незавершенного производства) либо на основе плана производства, либо по сумме производственных заказов. Как и большинство модулей, модуль ТЭП может использоваться в составе системы «Галактика» и автономно.

Важнейшей задачей, решаемой модулем, является расчет плановой себестоимости производства в целом и отдельных изделий. Расчет основывается на нормах расхода материалов и трудовых затрат на изготовление продукции с учетом планово-учетных или средних за месяц цен на материалы и тарифных ставок оплаты труда. И, наконец, имеется возможность расчета плановых отпускных цен на готовую продукцию на основе рассчитанной плановой себестоимости.

Модуль «Учет затрат на производство» позволяет производить расчет фактических затрат на производство по итогам деятельности предприятия за отчетный период и анализировать отклонения фактической себестоимости от ее плановых показателей. Практически вся необходимая для расчетов информация (в частности, о прямых и косвенных затратах) поступает из модуля «Технико-экономическое планирование» и из контура «Бухгалтерский учет».

Учет фактического выпуска готовой продукции и полуфабрикатов осуществляется на основе данных складского учета. Калькуляция может вестись как по предприятию в целом, так и по структурным подразделениям. Имеется возможность расчета фактической себестоимости отдельных изделий и производственных заказов.

Специальный *модуль «Техническая подготовка производства»* позволяет автоматизировать решение задач, связанных с конструкторской и технологической подготовкой производства. Модуль необходим как при освоении серийного производства изделий, так и при подготовке единичных производственных заказов. Для автоматизации формирования конструкторской документации в стандарте ЕСКД ведется база данных по номенклатуре выпускаемых изделий.

Технологическая подготовка производства в стандартах ЕСКД предполагает описание последовательности изготовления изделий на уровне видов работ, технологических операций и переходов. При этом можно производить расчет потребности в материалах, трудовых ресурсах, оборудовании, оснастке и инструменте практически в любых разрезах: предприятия, подразделения, изделия группы продукции, заказа и т.п.

В первую очередь модуль ориентирован на предприятия машиностроительного профиля, однако гибкость настройки позволяет эффективно использовать его и на других типах промышленных предприятий.

Модуль «Оперативное управление производством» предназначен для работников планово-диспетчерских служб предприятия. Он позволяет контролировать *код* выполнения производственной программы, следить за движением материальных потоков по цехам производства (на уровне маршрутных карт, листов и т.п.), а также осуществлять детальный учет незавершенного производства.

Центр информационных технологий «Ост-Ин». Центр представил на рынок *Корпоративную информационную систему «КхЗ»,* отвечающую задачам производственной системы, ориентированной на нужды заказчика.

«КхЗ» – комплексная система управления корпорацией, созданная с учетом изменяющихся технологий, рынков и деловой практики. Система дает возможность разработать собственную информационную стратегию, эффективно решающую задачи пользователя. Система построена на технологии открытых систем, что позволяет выбрать оптимальную комбинацию модулей для построения необходимой информационной среды предприятия.

Система «КхЗ» решает задачи комплексной автоматизации всех бизнес-процессов крупных организаций, имеет архитектуру «клиент-сервер», в качестве СУБД используется Orac1e8 Server Enterprise Edition (Workgroup Edition), для клиентской части Windows.

Прикладная часть создана при помощи Orac1e CASE – средств Designer/2000 и Developer/2000, которые обеспечивают полный цикл разработки сложных систем: от моделирования и перестройки бизнес–процессов на предприятии до получения функционально завершенных программ. В качестве инструмента для проведения анализа финансово–хозяйственной деятельности предприятия применена технология многомерного анализа OLAP (on–line analytical processing) на базе Orac1e Express.

Система «КхЗ» представляет собой ряд модулей программного обеспечения «клиент-сервер», которые поддерживают широкий спектр процессов, позволяющих соединить в одно целое множество управленческих задач:

- производственных,
- сбыта,
- снабжения,
- бухгалтерского учета,
- учета затрат (управленческий учет) и др.

Система динамично развивается и продолжает совершенствоваться.

Корпорация «Oracle». Основным финансовым продуктом этой фирмы является *Финансово-аналитическая система «Oracle Financial Analyzer» (OFA)*. Система OFA – признанный инструмент для формирования финансовой отчетности, проведения детального финансового анализа, ведения бюджета, финансового планирования и прогнозирования.

Благодаря объединению централизованного хранилища финансовой информации с мощным аналитическим инструментарием система позволяет организации достичь максимальной эффективности в управлении финансовыми потоками и их производными (ценовая политика, контроль затрат, анализ эффективности и т.д.) при составлении исчерпывающих финансовых отчетов и разработке приближенных к реальности финансовых моделей, в оценке возможностей и формировании будущей стратегии.

Среди решаемых с помощью системы OFA задач необходимо отметить следующие:

- создание, изменение, обработка, пересылка бюджетов, прогнозируемых финансовых моделей;
- быстрое и эффективное решение «горячих» вопросов управления финансовыми потоками;
- выявление и обработка информации по различным «профит-центрам»;

- разделение и систематизация информации по произвольным критериям;
- составление бюджета любого уровня сложности с использованием оперативной финансовой информации из единого информационного хранилища;
- поддержка обмена финансовыми данными с модулем ведения бухгалтерской отчетности «General Ledger», наличие обратной связи.

Основные возможности системы OFA:

- *разграничение доступа к данным* – система поддерживает хранение единой финансовой информации в центральном информационном хранилище и проверяет наличие прав доступа к информации, которая требуется пользователю;
- *адаптируемые экономические модели* – гибкая организация экономических моделей и информации, являющихся результатом работы системы OFA, позволяет мгновенно отразить оперативные изменения приоритетов в деятельности или организационной структуре объекта моделирования;
- *распространяемое бюджетирование и прогнозирование* – OFA предельно облегчает проведение процедур составления бюджета, прогнозов, обзоров, их создания, изменения, обмена данными с другими информационными модулями. Результаты работы OFA будут очень точно отвечать целям составления бюджета как на достаточно длинные, так и на малые временные периоды;
- *простые и доступные отчеты, глубокий анализ* – многомерные модели данных, обрабатываемые OFA, дают возможность пользователю выбрать необходимую информацию, используя произвольные комбинации ключевых параметров («куб данных») – линейное подчинение, последовательность временных интервалов, последовательность производимых продуктов, географическое подразделение, принадлежность к различным ценовым категориям и т.д.;
- *функциональные связи между таблицами и внутри них* – OFA напрямую соединяется с таблицами внутриотраслевых стандартов, и, таким образом, пользователь отслеживает данные и составляет отчеты по данным системы OFA прямо из этих таблиц. Этот доступ обеспечивается таким мощным средством управления многомерными базами данных, как Selektor, входящим в инструментарий таблиц;

- *гибкий доступ к новым данным* – позволяет легко загружать и выгружать данные из Главной книги, таблиц, реляционных баз данных и других операционных систем, используя рас ширенные возможности чтения и трансформации информационных файлов Orac1e Financial Analyzer;
- *единое информационное пространство с Главной книгой* – документы, подготовленные и обрабатываемые системой OFA, полностью интегрируются с Главной книгой. Данная интеграция позволяет избежать повторного ввода данных и тем самым обеспечивает наиболее эффективное функционирование системы управления финансовыми потоками. Информация, обрабатываемая в Главной книге (Orac1e Genera1 Ledger), в точности соответствует иерархическим структурам и размерностям, используемым при обработке соответствующих документов системой OFA;
- *средства графической обработки информации* – OFA использует знакомый графический интерфейс, который предоставляет легкий и быстрый доступ к мощному аналитическому инструментарию. Специальные Мастера системы, такие, как Se1ec1or, обеспечивают качественный анализ по характерным параметрам и анализ по методу «от противного». Отчеты же имеют знакомый пользователю внешний вид и функционально похожи на электронные таблицы.

Компания «Про-Инвест-ИТ». Программный комплекс *«Projectc Expert»* – это набор профессиональных инструментов для финансового управления бизнесом, для планирования и анализа инвестиционных проектов, подготовки бизнес-планов и оценки стоимости компании. Комплекс «*Projectc Expert*» позволяет описать деятельность практически любого предприятия независимо от его размера и отраслевой принадлежности.

С помощью этого программного комплекса можно:

- построить финансовую модель предприятия и окружения, в котором оно работает;
- оценить и проанализировать последствия и результаты планируемых решений, не производя реальных затрат;
- сравнить на основе сценарного подхода решения между собой и выбрать для реализации наиболее эффективное решение;
- определить устойчивость бизнеса к изменениям пара метров внешней и внутренней среды;
- рассчитать стоимость бизнеса и доходы его участников;

- определить эффективность работы подразделений, вклад каждого продукта или услуги и др.

В результате составляется стратегический план развития компании. В ходе его реализации можно ввести фактические данные, оперативно рассчитать расхождения от плана и оценить последствия этих расхождений.

При этом используются данные бухгалтерской отчетности, результаты представлены в соответствии с международными стандартами бухгалтерского учета, при расчете применяется методика, рекомендованная Международным банком реконструкции и развития,

В зависимости от целей и потребностей из семейства «*Projectc Expert*» можно выбрать программный продукт с подходящим набором функциональных свойств и по мере развития решаемых задач наращивать мощь используемых финансовых инструментов.

Project Expert Micro (PE Micro) позволит быстро и качественно провести обобщенный экспресс-анализ бизнес идеи, определить потребности в финансировании и подобрать подходящую схему финансирования, получить все данные для финансового раздела бизнес-плана, освоить методы финансового моделирования и анализа. С его помощью можно:

- кратко описать проект, указать его начало и длительность, ввести список производимых в будущем продуктов или услуг;
- выбрать две валюты проекта, указать их курсовое соотношение, учесть при расчетах ставки дисконтирования по ним и указать их изменения на период действия проекта;
- указать для каждой валюты проекта на время действия проекта инфляцию по различным объектам (сбыт, издержки, недвижимость и т.п.);
- описать налоговое окружение и его возможное изменение во время действия проекта;
- построить календарный план проекта в виде списка работ, указав для каждой начало, продолжительность и стоимость; каждую работу плана отнести к соответствующему активу и указать параметры линейной амортизации и порядок отнесения их на издержки;
- описать различные варианты сбыта товарной продукции, в том числе используя шаблон быстрого ввода данных, основанный на жизненном цикле продукта;
- описать прямые издержки на производство товарной продукции, в том числе в виде списка материалов и комплектующих, их

расходы, порядок оплаты и потери в цикле производства, а также сдельную заработную плату, в том числе по операциям;

- описать в трех разрезах (управление, производство, маркетинг) общие издержки, указав их размер, периодичность и время выплат;
- определить потребность в финансировании проекта и подобрать схему финансирования в виде займов или инвестиции, при этом указать схему поступления денежных средств, условия их возврата и использования;
- описать схему распределения прибыли, учесть льготы на прибыль. Описать схему формирования резервов. Определить временно свободные денежные средства и разместить их в виде инвестиций, указав сроки, условия их размещения и возврата;
- получить аналитические финансовые таблицы (Баланс, Отчет о прибылях и убытках, Кэш-фло, Отчет об использовании прибыли). Отобразить содержащуюся в них информацию в графическом виде;
- рассчитать финансовые показатели за все время действия проекта. Получить показатели эффективности инвестиций, определить их чувствительность от различных факторов внешней и внутренней среды. Составить график изменения стоимости бизнеса за время реализации проекта;
- сформировать и напечатать на различных языках (русский, английский, немецкий и т.д.) финансовые отчеты.

Project Expert Lite (PE Lite) позволяет быстро и подробно разработать инвестиционный план развития бизнеса, учесть начальное финансовое состояние предприятия, определить доходы и эффективность инвестиций для каждого участника проекта, подготовить качественное описание инвестиционного проекта.

Обладая всеми возможностями «PE Micro», «PE Lite» дополнительно позволяет:

- подготовить по заданной структуре на нескольких языках (русском, английском, украинском) подробное текстовое описание разделов инвестиционного проекта, при необходимости передать его в систему MS Word;
- описать детально финансовое состояние компании на начало проекта, указав запасы, готовую продукцию, условия погашения (возврата) кредиторской (дебиторской) задолженности, текущие займы и их состояние, структуру акционерного капитала;
- определить начало финансового года и принципы учета запасов (LIFO, FIFO, по среднему);

- учесть при описании финансового окружения по валютам проекта учетные ставки, задав параметры их изменения во время Действия проекта, и специфику отражения в учете выплаты процентов по займам;
- учесть при описании календарного плана проекта временные взаимосвязи между работами, назначить им ресурсы, определив их тип, количество, порядок использования, их стоимость и порядок оплаты. При описании ресурса можно задать индивидуальное описание его инфляции, а также особенности налогообложения, описать амортизацию активов по остаточному принципу или на производство конкретных продуктов или услуг, выбрать срок переоценки активов;
- учесть сезонность при сбыте продукции, детально описать условия оплаты (задержки платежей, авансовые и кредитные поставки), время и потери на сбыт, страховые запасы готовой продукции;
- вводить, кроме материалов и сдельной оплаты, дополнительно другие прямые издержки на производство продукции. При описании материалов и комплектующих учесть порядок, минимальную партию закупки, объемы и график закупки, что позволяет смоделировать складские запасы;
- отобразить особенности участия государства в финансировании проекта. При описании финансовых потоков учесть другие поступления и выплаты;
- оценить доходы и эффективность инвестиций для каждого участника проекта;
- экспортировать и импортировать данные в формате txt, dbf, для совместимости с электронными таблицами, базами данных и учетными системами пользователя.

Project Expert Standard (PE Standard) позволяет группе сотрудников тщательно и детально разрабатывать стратегические планы развития бизнеса, проводить статистический анализ проектов в условиях неопределенных (случайных) данных, подготавливать подробные и развернутые аналитические отчеты.

«PE Standard» дополнительно к возможностям «PE Lite» позволяет:

- расширить существенно возможности детального описания налогового окружения за счет использования настройки расчета налогов на основе формул, указания частных (локальных) налогов на каждый вид продукции, прямых и общих издержек, возможности указания способа списания налогов при амортизации активов;

- детализировать значительно описание инфляции за счет указания частной (локальной) инфляции на каждый вид продукции, прямых и общих издержек;
- учесть при описании сбыта продукции сезонные и скач-кообразные изменения цен, скидки;
- описать работу вспомогательного производства и раз личные изменения основного производства. Описание графика производства позволит смоделировать работу склада готовой продукции;
- учесть при описании закупок материалов и комплектующих сезонные и скачкообразные изменения цен, скидки;
- учесть задержку платежей и сезонность при описании общих издержек;
- определять по формуле размер общих издержек и относить их на различные статьи учета, в том числе на снабжение и сбыт;
- использовать при описании источников финансирования лизинг оборудования, указывая условия лизинговых платежей, страхования и выкупа оборудования;
- провести статистический анализ устойчивости проекта для выбранного набора неопределенных данных;
- детализировать результаты в разрезе любой строки аналитических таблиц;
- формировать собственные аналитические таблицы, используя исходные данные и результаты расчетов;
- использовать импорт стартового баланса и плана сбыта соответственно из систем «Audit Expert» и «Marketing Expert»;
- преобразовать подготовленные отчеты в формат html;
- создавать библиотеки проектов. Работать нескольким сотрудникам над одним проектом;
- обеспечить защиту проекта.

Project Expert Professional (PE Prof) позволяет создать систему финансового управления компанией на основе разработки стратегического финансового плана как комплекса инвестиционных проектов и контроля за его выполнением.

Возможности комплекса «PE Prof» значительно расширены по сравнению с системой «PE Standard», что дополнительно позволяет:

- описать внутреннюю структуру компании и степень участия каждого подразделения в производстве продукции;

- разнести общие издержки и издержки подготовительно го периода, а также доходы от инвестиций и другие доходы по подразделениям компании или производимой продукции;
- провести анализ деятельности подразделений компании и оценить их вклад в общий финансовый результат;
- провести анализ безубыточности для каждого вида продукции в любой период плана;
- оценить вклад каждого вида продукции в общем финансовом результате;
- осуществить контроль за ходом выполнения проекта путем ввода фактических данных о выполнении проекта и календарного плана;
- получить аналитические таблицы рассогласования фактических и плановых данных, рассчитать аналитические финансовые таблицы с учетом фактических данных;
- провести сравнительный анализ различных вариантов проекта, определить экономическую эффективность инноваций;
- провести сравнительный анализ различных проектов и выбрать подходящую структуру проектов для реализации;
- рассчитать консолидированные аналитические таблицы и показатели эффективности инвестиций по группе проектов.

PIC Holding – модификация системы Project Expert Professional, предназначенная для создания системы финансового управления холдинговой компанией.

В дополнение к возможностям системы «PE Prof» «PIC Holding» позволяет:

- создавать и анализировать финансовую модель, описывающую деятельность нескольких предприятий, реализующих различные проекты. Одно из них распределяет финансовые ресурсы, необходимые для выполнения проектов;
- выбирать наиболее эффективную структуру проектов, обеспечивать контроль за их выполнением, своевременно принимать решения о прекращении финансирования;
- организовывать рационально работу кредитных отделов банка и инвестиционных компаний.

Project Expert Tutorial (PE Tutor) – учебная версия программы. Она упрощает процесс обучения, поскольку в систему не нужно заносить формулы и увязывать между собой значения. Понятный, гибкий, хорошо структурированный интерфейс программы позволяет практически сразу приступать к процессу обучения.

Audit Expert – инструмент комплексного анализа финансового состояния и результатов деятельности предприятия. Базовой информацией для проведения анализа служат финансовые отчеты предприятия. Для более детального анализа используется дополнительная информация: описание структуры активов, собственного капитала и долгов компании, специальные таблицы, формат которых определяется самостоятельно.

Основным принципом работы системы является преобразование исходной финансовой информации в аналитические таблицы, соответствующие требованиям международных стандартов бухгалтерского учета – стандартов, поддерживаемых САА Р большинства развитых стран.

Такой подход делает результаты работы «Audit Expert» понятными во всем мире. Это единственная система, которая позволяет провести переоценку различных статей активов и пассивов предприятия для проведения анализа на основании реальных данных.

Система «Audit Expert» позволяет:

- рассчитать стандартные показатели ликвидности, финансовой устойчивости, рентабельности деятельности и деловой активности предприятия;
- рассчитать любые собственные финансовые показатели;
- сравнить значения финансовых показателей со средне отраслевыми показателями, показателями других предприятий или нормативными значениями и отобразить результат в наглядном виде;
- провести горизонтальный (динамический) и вертикальный (структурный) анализ финансовых данных;
- построить разнообразные графики, диаграммы и отчеты, которые могут быть распечатаны, переданы в MS Word или сохранены в формате html;
- импортировать исходные данные в систему Audit Expert напрямую из большинства бухгалтерских программ, текстовых файлов или вводить вручную;
- получить автоматически сформированное Заключение о финансовом состоянии анализируемого предприятия.

Система «Audit Expert» автоматически приводит Бухгалтерский баланс (форма № 1) и Отчет о прибылях и убытках (форма № 2) за все периоды к виду, соответствующему международным стандартам финансовой отчетности (International Accounting Standards – IAS).

По данным финансовой отчетности Audit Expert поможет оперативно провести вертикальный и горизонтальный анализ баланса, рассчитать показатели ликвидности, платежеспособности, рентабельности и деловой активности.

Sa1es Expert – информационно-аналитическая система предназначена для проведения мониторинга сбытовой деятельности предприятия и даст оценку эффективности маркетинговых мероприятий и результативности сбыта. Это первая российская система класса CRM (Customer Relationship Management).

Система «Sales Expert» реализована в архитектуре «клиент-сервер» на СУБД InterBase фирмы Inprise. Многопользовательский характер программного продукта позволяет использовать его в рамках всего предприятия и накапливать актуальную информацию о клиентах и контрактах с ними в единой базе данных. В базе данных накапливается вся информация о развитии отношений с каждым клиентом. В системе реализованы функции выписки счета, накладной и счет-фактуры, репликации данных по продажам, выполняемым в разных офисах компании, средствами модуля рассылок автоматически рассылаются типовые документы и демонстрационные материалы по электронной почте и т.д.

С помощью Marketing Expert, на основании данных управленческого учета, проводится сегментация рынка, анализ доходности товара, портфеля продукции и компании в целом, сравнительный анализ с конкурентами, определяется эластичность спроса и многие другие показатели эффективности маркетинга. Система позволяет определить доходность и прибыльность различных сегментов рынка и товаров; долю рынка компании, темпы роста доли рынка.

Опираясь на прогноз внешних экономических факторов, полученный с помощью «Forecast Expert», система «Marketing Expert» помогает сформировать и оценить варианты стратегии развития предприятия. При этом для различных сегментов рынка, вариантов цен, действий конкурентов оцениваются риски, связанные с осуществлением стратегий, и определяются цели, которых должна достичь компания. В системе применяются общепринятые аналитические методики: GAP-анализ, сегментный анализ, SWOT-анализ, Portfolio-анализ.

Разработанный в системе «Marketing Expert» стратегический план маркетинга используется для прогнозирования объемов сбыта в программе Project Expert. Перечень программных продуктов приведен на рис. 5.33. Project Expert – инструмент прогнозирования экономических показателей и анализа тенденций рынка. Система может определить будущий объем продаж, спрос на услуги или изделия, курсы

валют, акций или фьючерсов, остатки денежных средств на счетах, другие значимые показатели.

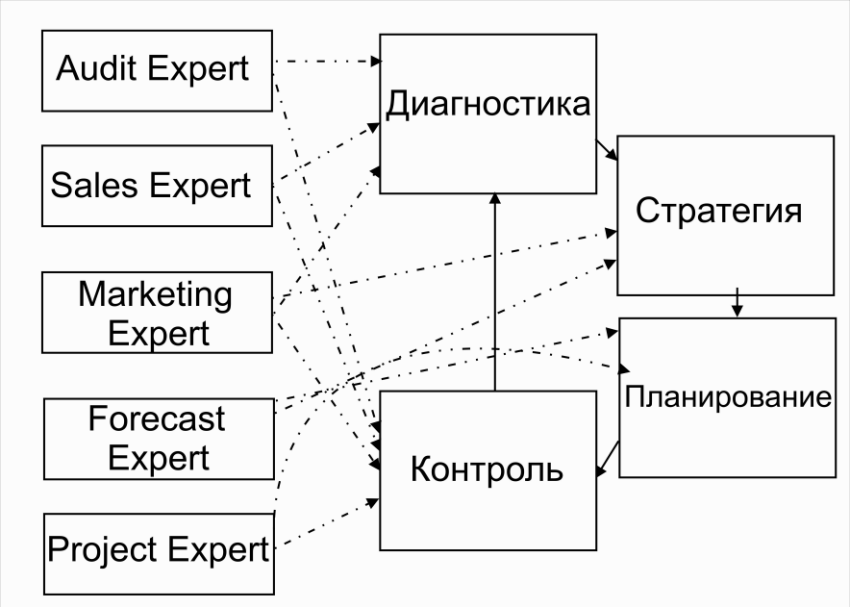


Рис. 5.33. Программные продукты Project Expert

Корпорация «Epicor Software». Она входит в число десяти крупнейших фирм – производителей программного обеспечения для автоматизации предприятий и занимает устойчивое лидирующее положение на рынке программных продуктов, ориентированных на компании среднего масштаба. В настоящее время «Epicor» является производителем интегрированных систем для автоматизации управления всеми аспектами деятельности предприятия, включая электронную коммерцию. Основные продукты корпорации – ***«Active Planner»*** и ***«Platinum SQL».***

«Active Planner» – этот программный пакет позволяет в ходе составления бюджета предприятия проигрывать разные варианты по принципу «что, если...». Система обеспечивает комбинацию разных подходов к составлению бюджета «сверху вниз» и «снизу вверх». В результате компания получает возможность наиболее оптимальным образом строить бюджет: общие цели бюджетирования задаются на верхнем уровне исходя из стратегии компании и общих экономических прогнозов, а конкретный проект бюджета создается в подразделениях.

Пакет «Active Planner» существенно расширяет возможности электронных таблиц Excel, которые на данный момент представляют собой наиболее распространенный способ построения бюджетов. В то же время «Active Planner» может работать с Excel, а общение с новой системой мало чем будет отличаться от работы с этой программой, так что на обучение не придется тратить много часов. «Active Planner»

предусматривает разные уровни доступа к данным, позволяя при необходимости ограничивать возможности менеджеров разного уровня.

Эта система позволяет упростить составление бюджета для предприятия, применяя привычные функции работы с электронными таблицами в сочетании с мощными средствами интеграции информации по всей компании.

Система поддерживает также работу с данными в режиме удаленного доступа через Интернет. С помощью пакета «Active Planner» можно:

- создавать бюджетные таблицы на основании оперативных и исторических данных;
- просматривать их текущий статус и отслеживать внесенные изменения;
- управлять слиянием и распределением значений входящих в таблицы ячеек;
- определять права доступа пользователей к бюджетным таблицам.

«Platinum SQL» – полнофункциональная система финансово-управленческого учета. Она обеспечивает полную автоматизацию бухгалтерских операций, автоматизирует оперативный управленческий учет, складской учет, расчеты с поставщиками и заказчиками, учет основных средств и нематериальных активов, предоставляет исчерпывающую финансовую и управленческую отчетность и многое другое.

Система «Platinum SQL» позволяет вести учет и формировать отчеты в неограниченном количестве валют. Возможна работа с несколькими курсами обмена (курс покупки, курс продажи, льготный курс, курс ЦБ РФ и т.д.). Каждая операция отражается в системе одновременно в трех валютах: валюте операции и двух независимых – национальной и управленческой (в этом качестве отечественные предприятия чаще всего используют доллары США).

«Platinum SQL» полностью поддерживает российские и международные стандарты бухгалтерского учета. Благодаря развитым иерархическим межмодульным связям пользователь может при вводе и обработке данных в одном модуле оперировать информацией, хранимой в других модулях.

Первичной информацией для системы «Platinum SQL» является не проводка, а хозяйственная операция, которая сопровождается формированием проводок. Это позволяет вводить в систему дополнительную информацию, характеризующую каждую конкретную хозяйственную операцию, обеспечивая при этом автоматическое

создание бухгалтерских записей, распечатку первичных документов, ведение архива операций.

Каждая операция в системе проходит две стадии: ввод и разноску. Неразнесенную операцию можно в любой момент удалить без сохранения в архиве. Разнесенную операцию можно только откорректировать, так как разноска данных приводит к изменениям в базах данных нескольких модулей «Platinum SQL», позволяет учитывать не только прошедшие, но и будущие операции.

Отличительной чертой программы является параллельное ведение бухгалтерского учета неограниченного числа компаний (баз данных) с разными настройками плана счетов, используемыми валютами, системами учета основных средств, товарными единицами и системами их классификации, разными складами, различными поставщиками и заказчиками.

«Platinum SQL» формирует финансовую отчетность в соответствии с российскими и международными стандартами бухгалтерского учета. Предусмотрена возможность проведения детального аналитического учета на уровне реальных объектов и субъектов деловой активности компании.

Предусмотренная в системе функция Drill-Down позволяет при построении финансовых отчетов мгновенно получать подробную аналитическую расшифровку операций, осуществление которых привело к анализируемому финансовому результату. Процедура консолидации компаний позволяет получать консолидированные финансовые отчеты в необходимой для пользователя валюте и согласно выбранным стандартам учета.

«Platinum SQL» обладает интуитивно понятным Windows-интерфейсом. В системе возможны: настройка экранных форм с помощью Visual Form Designer (Form Editor), изменение встроенных правил и процедур с помощью Visual Basic, а также настройка запуска из внешних приложений (Microsoft Word, Microsoft Exce1) с помощью OLE-интерфейса.

Опыт внедрения этой системы в России показывает, что она может эффективно работать как в крупных корпорациях, так и в компаниях среднего размера. Система может свободно поддерживать более ста одновременно работающих пользователей и осуществлять разноску нескольких тысяч операций в минуту.

С помощью настроек системы можно автоматизировать деятельность предприятия практически любой отрасли, формы собственности и структуры. Пользователь может самостоятельно выполнить эти настройки в соответствии с требованиями его бизнеса. С

помощью специального редактора в интерактивном режиме можно добавлять, удалять или переименовывать поля, пункты меню и другие элементы пользовательского интерфейса.

«Platinum SQL» создавалась специально под технологии Microsoft и использует все преимущества этой платформы. Благодаря этому система имеет интуитивно понятный интерфейс, отличается легкостью настройки и требует относительно небольших затрат на аппаратную часть и внедрение. Финансовые данные из системы могут быть легко конвертированы в такие популярные приложения, как Microsoft Word, Microsoft Exce1.

Для работы «Platinum SQL» на центральном сервере должен быть установлен Microsoft SQL Server, а на рабочих местах пользователей – Windows NT или Windows 95/98/2000.

Компания «Общероссийская сеть Консультант Плюс». Системы *«Консультант Плюс»* используются как опытными, так и начинающими пользователями, так как они предоставляют богатейшие возможности для поиска документов и анализа законодательства. Интуитивно понятные принципы общения с системой позволяют даже неподготовленному пользователю освоить базовые операции работы с системой после 20-минутного предварительного обучения.

Поисковые возможности систем «Консультант Плюс» позволяют найти документ или подборку документов по любым характеристикам: от официальных реквизитов документов до отдельных слов, встречающихся в текстах. Поэтому успешно работать с системой могут и юристы-профессионалы, и пользователи без специального юридического образования. Кроме того, предусмотрена возможность одновременного использования всех видов поиска, что позволяет найти все необходимые документы. Поиск по реквизитам документов дает возможность найти необходимый документ (или подборку документов) по реквизитам. При проведении поиска заполняется карточка запроса, в которой указываются один или несколько известных пользователю реквизитов: тематика документа, вид документа, принявший орган, дата принятия, номер документа, дата регистрации в Минюсте, регистрационный номер, присвоенный в Минюсте, и название документа.

Поиск по тематике позволяет найти документ по тематическому рубрикатору, составленному на основе Общеправового классификатора отраслей законодательства и имеющего четыре уровня детализации (616 рубрик).

При интеллектуальном поиске по текстам документов, наряду с их реквизитами, можно задавать любые слова и словосочетания,

встречающиеся в текстах. В системы встроены специальные индексные словари, включающие все слова из текстов всех документов. Благодаря им поиск происходит практически мгновенно и пользователь застрахован от ошибок ввода при формировании запроса.

Поиск по названию позволяет найти необходимый документ по любым словам или словосочетаниям, встречающимся в его названии.

Поиск по ключевым словам является вспомогательным видом поиска по выбранным специалистами «Консультант Плюс» ключевым словам, описывающим основные понятия. При каждом изменении или введении любого параметра поиска система мгновенно показывает количество документов, удовлетворяющих формируемому запросу. Это дает возможность понять, стоит ли дальше уточнять запрос или можно воспользоваться полученной выборкой.

Если у пользователя установлено несколько систем «Консультант Плюс», то поиск документов возможен в нескольких базах одновременно. Выбирая пункт меню «Поиск по нескольким базам», пользователь отмечает в специальном окне те системы «Консулътант Плюс», по которым будет проводиться одновременный поиск документов. Далее весь поиск выполняется по стандартной методике.

Для проведения поиска по нескольким базам используются все стандартные методы и приемы технологии «Консультант Плюс», включая сложный поиск по тексту (с указанием логических условий: И, ИЛИ, КРОМЕ, РЯДОМ).

Система представляет результаты поиска в удобном структурированном виде. Найденные документы автоматически распределяются по спискам, составленным в соответствии с теми информационными массивами, по которым проводился поиск. При этом нормативные документы, консультации, справки и формы документов помещаются в отдельные списки. Пользователь может самостоятельно сформировать удобное для себя рабочее пространство, используя возможности системы. Пользователь может создавать свои собственные постоянные подборки документов по какой-либо проблеме. При этом поиск возможен как по всей базе, так и по конкретным папкам. В системах реализованы операции пересечения и объединения папок документов.

Пользователи, работающие на различных компьютерах, могут обмениваться папками документов. Это позволяет организовать коллективную работу нескольких специалистов над общей проблемой.

Расставленные в текстах документов ссылки позволяют моментально переходить в тексты документов, на которые ссылается законодатель. При этом пользователь попадает именно в ту главу, часть,

параграф, статью, к которой относится ссылка. Системы «КонсультантПлюс» позволяют распечатать или записать в файл текст документа любой его фрагмент, а также список документов. Печать возможна на любом принтере. Перенос документов в Word осуществляется простым нажатием кнопки на панели управления системы «Консультант Плюс». При этом происходит запуск Word, и весь документ или его выделенный фрагмент копируется из системы в текстовый редактор. Если Word был ранее запущен, то копируемый текст или таблица вставляется в документ сразу за курсором.

Компания «Гарант» – одна из крупнейших российских информационных компаний. Направление деятельности – производство и поддержка компьютерной *правовой системы «Гарант»,* информационно-правовое обслуживание предприятий, общественных объединений и организаций. Названная система создана специально для тех, кто работает с нормативными документами и решает правовые вопросы. Это колоссальный информационный банк, охватывающий весь спектр российского законодательства и основных норм международного права. Все документы поступают в систему непосредственно из 186 органов власти и управления.

В 17 тематических базах находится правовая и экономическая информация по всем разделам законодательства. Законодательства 41 субъекта федерации представлены в собственных региональных базах данных. Работа с правовыми базами осуществляется в многофункциональной гипертекстовой среде с перекрестными ссылками и мощными поисковыми возможностями. Уникальную возможность для перевода и толкования юридических и экономических терминов предоставляют толковые словари «Бизнес и право».

Все документы представлены в действующей редакции. Перед включением в информационный банк каждый из них проходит сложную юридическую обработку. Поступление новой информации происходит ежедневно. А еженедельное обновление всего банка данных (около 2 000 документов) исключает возможность применения застаревшей информации. Система предъявляет минимальные требования к аппаратному обеспечению. Можно приобрести базы-системы для персонального компьютера, установить интранет-версию на внутренней сети компании, а также работать с правовыми базами в сети Интернет.

Разработанная компанией компьютерная правовая система «Гарант» представляет собой полный комплекс программ по всем отраслям российского и международного права. Высокое качество программных продуктов марки «Гарант» подтверждено результатами

российских и международных конкурсов. Уникальный цикл юридической обработки поступающей информации позволил создать и поддерживать в действующем состоянии компьютерную модель российского законодательства.

5.5. Коммерческий статус программ. Виды распространения

Помимо тематического деления программ существует еще одна классификация. Связана она со способом распространения программы и теми условиями, приняв которые, потребитель получает возможность оной воспользоваться. И, следовательно, с ее стоимостью.

Программы бывают платные и бесплатные.

Бесплатное программное обеспечение (freeware). Первоначально к бесплатным программам и пользователи, и разработчики относились довольно скептически. Как правило, в виде freeware распространялись небольшие вспомогательные программы-утилиты, разработанные независимыми программистами, и изредка – бесплатные дополнения к известным коммерческим пакетам. Однако сегодня статус freeware имеют и весьма серьезные пакеты известных производителей – например, офисный пакет StarOffice корпорации Sun, операционные системы семейства Linux и практически все программы, созданные для них.

*Условно-бесплатное программное обеспечение (shareware) – с*амая массовая группа программ, в которую входят практически все утилиты, а часто и весьма серьезные программные пакеты. Как правило, shareware-программы распространяются в виде полнофункциональных версий, ограниченных либо по времени работы, либо по количеству запусков. По истечении отведенного на тестирование срока (как правило – от 15 до 45 дней) программа либо просто перестает запускаться, либо утрачивает часть своих функций, превращаясь в менее функциональную freeware-версию. В самом благоприятном случае программа полностью сохраняет работоспособность, однако время от времени «напоминает» пользователю об оплате. Так поступает, например, популярный файловый менеджер Total Commander.

Если же пользователь приобретает программу, перечислив на счет автора определённую сумму, то в обмен он получает специальный цифровой код (ключ), который необходимо ввести в специальное регистрационное окошко программы (как правило, найти его можно в меню *Help* или *About*). В качестве варианта может быть выслан специальный «ключевой» файл, который необходимо скопировать в папку с установленной программой.

В любом случае, после этих действий программа становится «зарегистрированной».

В России, правда, значительно чаще программы не покупают, а «ломают», используя либо украденные чужие регистрационные номера, либо специальные программы, предназначенные для взлома. Моральный аспект этого дела можно оставить на совести пользователей, но необходимо знать некоторую практическую информацию (см. п. 5.6)

«Рекламно-оплачиваемые» программы (ad ware). Этот вид распространения программ появился сравнительно недавно и стремительно завоевал популярность как у пользователей, так и у разработчиков. Ибо в этом случае потребитель получает возможность работать с программой бесплатно, а на счет производителя перечисляется определённая сумма, которую оплачивают программистам крупные фирмы-рекламодатели. Программист за эту услугу внедряет рекламную картинку-«баннер» данной фирмы в свою программу. Таким образом, пользователи вынуждены смотреть внедрённую рекламу, а иногда – дополнительно осуществив щелчок по особо понравившимся картинкам, может перейти на сайт фирмы-рекламодателя.

Модификацией ad ware является еще один статус распространения программ – *homepage ware***.** При установке программа автоматически устанавливает свою страницу в Интернете в качестве стартовой страницы вашего браузера, например Microsoft Internet Explorer. То есть при запуске браузера вы обнаружите страницу, на которой помещена реклама.

Еще не так давно по принципу ad ware распространялись лишь мелкие утилиты, однако сегодня этот статус имеют такие известные пакеты, как Интернет-пейджер ICQ или менеджер докачки файлов Get Right.

Для проверки вашего компьютера на предмет наличия spyware-программ и возможного их удаления шведская фирма Lavasoft (www.lavasoft.de) разработала специальную программу Ad-aware, которую желающие могут совершенно бесплатно загрузить с сайта компании в сети Интернет. Использование этой программы сами производители программ приравнивают к обычному взлому, что, в общем, вполне закономерно. Ведь, удаляя рекламные модули из программы, вы тем самым нарушаете соглашение с ее разработчиком – он тоже имеет право на получение законной прибыли от рекламы! Однако Ad-aware помогает определить, насколько глубоко запрятана в программе «шпионская» начинка: если после удаления с помощью Ad-

aware программа не утрачивает функциональность, вполне вероятно, что с ее прежним вариантом можно работать и дальше. Наоборот, нельзя оставлять на компьютере программы, которые после удаления из них рекламных модулей отказываются работать: это значит, что такая программа – не более чем оболочка для программного «шпиона».

Коммерческое программное обеспечение (commercialware). За эти программы всегда надо платить, и чаще всего – довольно значительные суммы. Сюда относятся все крупные программные пакеты известных производителей и ряд утилит. Программы этого типа можно приобрести в красивых коробках или в любом компьютерном супермаркете. Однако сегодня все чаще и чаще программные продукты продаются через сеть Интернет. Сделать это можно либо на сайтах производителей программ, либо в больших Интернет-магазинах программного обеспечения (например, как российский сервер www.bolero.ru). При этом, как и в случае с shareware-программами, вы получаете урезанную (Demo) или ограниченную по времени работы (Trial) версию. Trial, как и shareware-программу, можно превратить в полнофункциональный вариант с помощью регистрации.

Другие виды программ. Помимо четырех основных видов программных статусов существует еще несколько весьма экзотических способов распространения ПО и оценки их стоимости. *«Условно-платные» программы (donation ware).* Автор таких программ платить никого не принуждает и функциональность программы не ограничивает, но и напоминает об оплате. *«Открыточные» версии (cardware).* Весьма экзотический вид программ: в качестве вознаграждения за пользование которыми вас просят отправить автору красивую почтовую открытку.

5.6. Лицензионное и нелицензионное ПО

Возникающие трудности при покупке программных продуктов весьма разнообразны:

- количество программных продуктов (ПП), их версий, линеек, разновидностей растет с каждым днем;
- одна и та же версия ПП может поставляться в «облегченном», «среднем» и «полном» вариантах, которые включают различные возможности, инструменты и параметры;
- процесс поиска и выбора необходимой для пользователя программы даже среди линейки однотипных ПП одного производителя является в последнее время довольно непростым;

- желание пользователей самостоятельно разобраться во всем богатстве предлагаемых ПП и выбрать именно тот, который подойдет ему, по своим функциям, цене, системным требованиям и условиям эксплуатации;
- в отличие от заказного ПП, тиражные ПП выпускаются партиями, в пределах которых все экземпляры идентичны, в связи с этим у покупателя нет возможности влиять на набор функциональных возможностей и на качество ПП до момента его приобретения и начала использования;
- особое значение приобретает достоверность и объем информации, которая позволит ему сделать осознанный выбор в пользу конкретного ПП и будет доведена до потребителя в момент совершения сделки.

Преимущества использования лицензионного софта

Сегодня все больше и больше компаний и организаций в России отдают предпочтение лицензионным программным продуктам. Как профессионалы в области информационных технологий, так и обычные пользователи, использующие компьютеры для работы или отдыха, принимают во внимание три основных причины, по которым лицензионные продукты оказываются более рентабельными в использовании по сравнению с пиратскими копиями.

Рис. 5.34. Диаграмма уровня потерь производителей ПО от пиратов

Сравнение представлено на рис. 5.34 и 5.35):

1) при работе с серьезными проектами, особенно при сотрудничестве с международными организациями, использование

пиратского ПО не допустимо. Это обусловлено тем, что все иностранные партнеры требуют наличия сертификатов на используемое ПО, в связи с чем наличие хотя бы на одном рабочем месте пиратского ПО может стать основанием для разрыва отношений и потери выгонных контрактов;

2) работа с лицензионным ПО является обязательным требованием для прохождения сертификации организацией на соответствие требованиям международных стандартов ISO;

3) только лицензионное ПО дает возможность получения оперативной технической поддержки в решении любых вопросов и устранении возможных неполадок.

Рис. 5.35. Диаграмма уровня пиратства восточноевропейских стран

Работа с нелицензионным ПО противоречит требованиям законодательства о соблюдении прав интеллектуальной собственности и может привести к уголовной ответственности

Приобретение лицензионного ПО позволяет подключиться к огромному информационно-техническому ресурсу, который предоставляет компания, внедряющая ПО, имеющая опыт внедрений и постоянный контакт с огромным количеством пользователей из различных отраслей.

Пользователи лицензионного ПО могут быть полностью уверены в устойчивости его работы, что дает возможность спокойной работы в серьезных проектах. Деньги, вложенные в лицензионное программное обеспечение, способствуют дальнейшему совершенствованию продукта.

В России и странах СНГ уровень использования пиратских компьютерных программ один из самых высоких в мире. По этому

показателю мы уступаем лишь Китаю и нескольким странам Африканского и Азиатского регионов.

Виды компьютерного пиратства

Можно выделить как минимум шесть видов компьютерного пиратства (рис. 5.36).

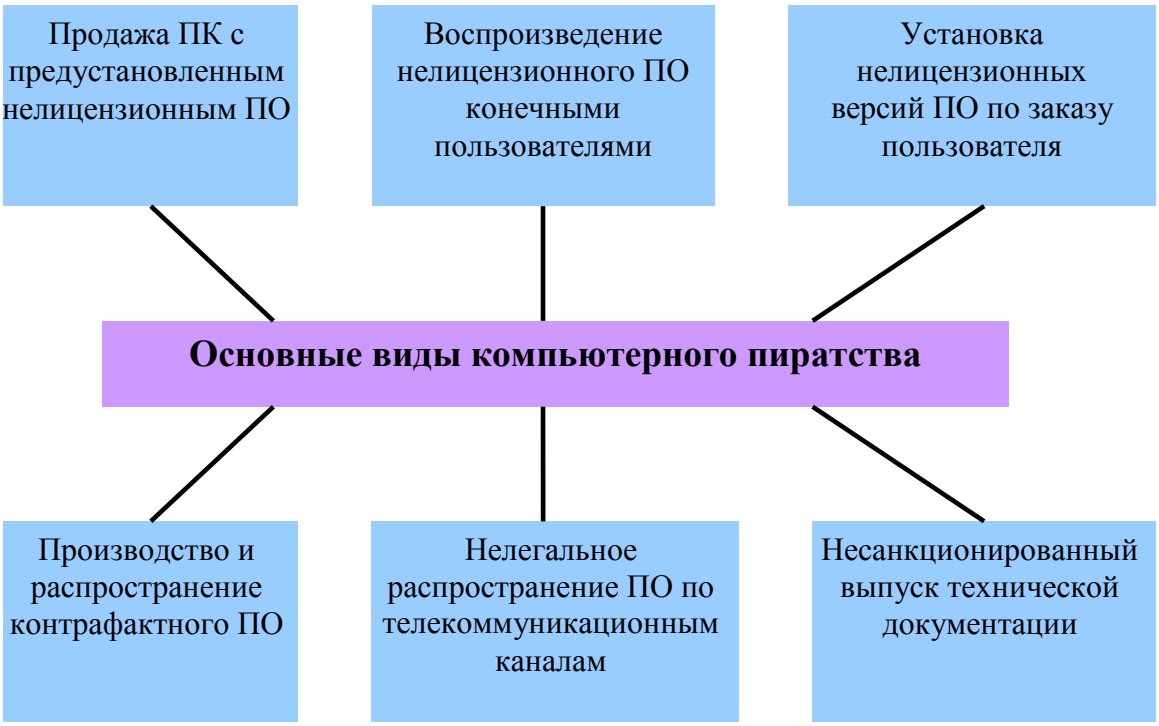


Рис. 5.36. Виды компьютерного пиратства

Производство и распространение контрафактного ПО

Контрафактная продукция может представлять собой грубую подделку, но часто ее внешний вид практически не отличим от легальных образцов. Незаконное происхождение продукции второго вида без соответствующей экспертизы выявить довольно сложно, и потому подобные продукты иногда распространяются через легальные дистрибьюторские каналы.

Выпуск контрафактных тиражей может осуществляться как «подпольными» заводами, так и зарегистрированными предприятиями. В тех случаях, когда оборудование легального завода используется для выпуска грубых подделок, с носителей, как правило, удаляется вся информация, идентифицирующая производителя. А если завод по собственной инициативе незаконно печатает «дополнительный» тираж дисков, партия которых ранее была заказана правообладателем, такие диски внешне в точности соответствуют легальной продукции.

Воспроизведение нелицензионного ПО конечными пользователями

Согласно п. 1 ст. 25 ЗоАП, допускается установка лишь одной копии программы и только на один компьютер, если договором с правообладателем не предусмотрено иное. Если же лицо, которое приобрело один экземпляр программы для компьютера, предполагает произвести большее количество инсталляций, ему необходимо заключить отдельный лицензионный договор.

Таким образом, легальный пользователь должен располагать либо определенным числом коробочных или OEM-версий продукта (по числу компьютеров, где он используется, или по числу локальных сетей, если приобретаются сетевые версии с неограниченным количеством пользователей), либо иметь отдельную лицензию на право использования определенного числа копий продукта (независимо от того, объединены ли компьютеры в сеть).

Ответственность за нарушение авторских прав

Выявление нарушений в сфере защиты интеллектуальной собственности, как правило, осуществляется на основании проверки информации о правонарушении. При этом следует отметить, что наличие заявления потерпевшего не является обязательным условием инициирования проверки и возбуждения уголовного дела.

В случае если правонарушение выявлено правообладателем, он обращается в правоохранительные органы с целью проведения проверочной закупки, которая также может принимать форму контрольной установки нелицензионного ПО со стороны «черного внедренца». Если установлено, что незаконное использование объектов авторских прав совершено в крупном размере, лицо привлекается к уголовной ответственности по ст. 146 УК РФ «Нарушение авторских и смежных прав», а при незначительном размере наступает административная ответственность. Независимо от привлечения к уголовной или административной ответственности правообладатель может требовать возмещения причиненного ущерба в гражданском порядке, подготовив и направив соответствующий иск в суд по месту жительства ответчика.

Опасности использования пиратского бизнес-софта

В отличие от игр или обучающих домашних программ, проблемы с которыми обычно не приносят значительных убытков пользователям, бизнес-программы (при их ненадлежащей работе) могут угрожать самому существованию компании.

Существует реальный риск, что фирма может быть привлечена к юридической ответственности за использование нелицензионного ПО. Приговоры по ст. 146 УК РФ выносятся в отношении как пользователей, так и лиц, занимающихся установкой нелицензионных

программ; и даже если лишение свободы назначается директорам фирм условно, то последствия в виде судимости подрывают их репутацию, затрудняют выезд за границу, приносят массу других проблем, в том числе в виде многотысячных штрафов.

Следует также помнить, что пираты не отвечают за качество продукта; при изменении кода они не всегда проверяют функциональность программы и наличие вирусов. И если у владельцев нелицензионного софта будут потери важной корпоративной информации, то ее восстановлением разработчик применяемого программного обеспечения заниматься не будет.

Контрольные вопросы и задания

1. Состав СПО и ППО.
2. Дайте определение понятию «Прикладные программы»
3. Чем отличаются проблемно-ориентированные ППП от методо-ориентированных ППП?
4. Перечислите и охарактеризуйте основные программы, входящие в комплекс MS Office.
5. Назовите состав и инструментальные программные средства общего назначения.
6. Назовите состав и приведите примеры проблемно-ориентированных ПС. Перечислите элементы издательского дела.
7. Что представляет собой принцип WYSIWYG?
8. Каково назначение электронных таблиц и СУБД?
9. Приведите примеры интегрированных пакетов программ.
10. Назовите телекоммуникационные и сетевые программы.
11. Перечислите инструментальные ПС специального назначения.
12. Каково основное назначение мультисред?
13. Каково назначение авторских систем?
14. Дайте понятие «Экспертная система».
15. Что означает термин «интеллект»? Соотнесите понятия интеллекта человека и искусственного интеллекта.
16. В чём суть понятия интеллекта в информатике?
17. Перечислите цели и задачи искусственного интеллекта.
18. Дайте определение понятию «База знаний».
19. Приведите примеры программ, использующих искусственный 1
20. Перечислите проблемно-ориентированные программные средства.
21. Сделайте обзор программных продуктов ведущих фирм.
22. Перечислите виды распространения ПО.
23. Перечислите преимущества лицензионного ПО.

6. ПОДХОДЫ К ОЦЕНКЕ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ВЛОЖЕНИЙ В ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

6.1. Методы оценки экономической эффективности

Тезис о том, что информация в современных условиях является важнейшим ресурсом предприятия, не подвергается сомнению. Большинство современных успешных организаций любого уровня (предприятие, регион, государство и т.д.) в любой отрасли экономики не мыслит своей деятельности без применения информационных технологий и информационных систем, как минимум – для обеспечения нормального функционирования бизнес-процессов. Более того, многие современные успешные руководители понимают важность и роль ИТ и ИС в создании конкурентных преимуществ их организации.

ИТ-бюджеты могут занимать достаточно большую долю в обороте компании. Так, можно привести следующие цифры в качестве критерия достаточности размера затрат на ИТ на основании западного опыта:

- 0,9 – 3,4 % от оборота компании в зависимости от размера и динамичности;
- $1600 – $3900 на одного работающего в год.

Те же показатели для российских компаний естественно ниже:

- 0,6 – 1,5 % от оборота компании;
- $200 – $1000 на одного работающего в год.

Имеет место интересная шкала градации состояния ИТ-организаций (подразделений предприятия, занимающихся сферой обработки информации) в зависимости от доли затрат на ИТ в выручке предприятия. Так, доля затрат в выручке менее 1 % свидетельствует о «нищенском» состоянии организации и возможном ее крахе в ближайшей перспективе.

Нормальной считается доля 2–3 %, которая позволяет поддерживать как стратегические, так и тактические планы организации, обеспечивать долгосрочные и краткосрочные инвестиции в сферу ИТ. Идеальным для предприятий, использующих ИТ в качестве одного из орудий в конкурентной борьбе, является интервал от 3 до 6 %. Доля расходов от 6 до 15 % расценивается как агрессивные инвестиции в ИТ, что может быть вызвано разными причинами, часто это свойственно начинающим ИТ-компаниям. Доля расходов свыше 15 % – «путь к самоубийству».

Сразу нужно оговорить, что речь не пойдет о предприятиях, для которых ИТ являются технологией основного производства (например, телекоммуникационные организации, компании, специализирующиеся на Интернет-услугах). Для таких компаний, если не выделять затраты, направленные на извлечение доходов, в отдельную статью (привязывать к стоимости проданных товаров) доля ИТ-бюджетов может достигать 20 %.

И в то же время, согласно данным отчета McKinsey, на типовом предприятии примерно 15–20 % компьютерных проектов можно спокойно свернуть, поскольку они никак не влияют на общее благосостояние компании. Еще 25 % проектов соответствуют поставленным целям лишь отчасти, следовательно, нет смысла реализовывать их полностью. На основании этих исследований McKinsey делает вывод о том, что среднестатистическое предприятие вполне может безболезненно сократить свои затраты на информационные технологии (ИТ) на 30 % за два–три года только благодаря пересмотру портфеля проектов и объединению систем.

Однако, не выстроив в рамках корпоративного бюджета правильной структуры расходования средств на ИТ и не просчитав предполагаемый эффект от различных технологических нововведений, сделать это невозможно. Больше не касаясь вопроса целесообразности обоснования инвестиций в сферу ИТ, попробуем предложить некоторую классификацию подходов к оценке эффективности вложений. Прежде всего, эволюция подходов к оценке эффективности ИТ связана с историческим изменением роли информационных технологий и информационных систем в организации.

На первых этапах развития информационным технологиям и системам в организации отводилась роль «робота», выполняющего рутинные повседневные операции, главная их задача состояла в ускорении процессов расчетов и формирования отчетности. Отсюда легко можно было найти эффекты от внедрения ИС, которые чаще всего заключались в сокращении трудовых и стоимостных затрат на обработку информации. С повышением значения ИТ для реализации бизнес-стратегии организации, для достижения конкурентных преимуществ организации изменились и эффекты, получаемые от внедрения ИТ.

Так, например, Денис Ганстер (Dennis Ganster), председатель совета директоров и Президента Comshare Inc., полагает, что для осознания эффективности инвестиций в ИТ необходимо рассматривать шесть «бонусов» повышения эффективности организации:

- повышение «интеллектуальности» бизнеса (оперативное наличие больших объемов релевантной информации позволяет управленцу принять перспективное, упреждающее решение);
- оптимизация планирования (своевременный доступ всех заинтересованных пользователей к важной информации, находящейся в одной централизованной БД);
- усовершенствование процессов принятия решений (решения становятся более обоснованными, если они подкреплены достоверной и оперативной информацией, экономится время на анализ второстепенных деталей);
- повышение рыночной привлекательности компании – рынок благосклонен к тем компаниям, которые демонстрируют внимание к деталям своей деятельности, и, более того, их полноценный анализ;
- расширение информационной компетентности – чем большее количество сотрудников имеет доступ к корпоративным данным, тем совершеннее и мобильнее становится организация в целом.
- создание единой среды сотрудничества (организация приобретает мощный заряд развития, так как каждый из ее членов работает на достижение прозрачных, понятных общих целей).

Вообще понятие эффективности ИС связано с соотношением «затраты/результат», а точнее с достижением его оптимального уровня. Целесообразные варианты построения ЭИС выбираются путем балансирования показателей приращения эффекта (Э), получаемого за счет создания или совершенствования экономической информационной системы, и затрат (Q). Математически эту задачу формулируют в виде

$$MAX\ Э\ при\ Q = CONST$$

или (в виде обратной задачи)

$$MIN\ Q\ при\ Э = CONST.$$

С определением затрат проблем, как правило, не возникает. Достаточно легко посчитать стоимость разработки, эксплуатации и поддержки системы. А задача расчета эффекта от внедрения ИС не так проста, так как получить количественные оценки, например, такого эффекта, «повышение рыночной привлекательности компании», с первого взгляда представляется практически невозможным. Это объясняется тем, что информационные технологии воздействуют на финансово-экономические показатели деятельности организации не прямо, а опосредованно через бизнес-технологии. Степень сложности определения эффекта зависит также от типа ИС с точки зрения

значимости для бизнеса сегодня и в будущем. Согласно классификации Макфарлана выделяют четыре типа ИС:

- производственные или ключевые (важные для преуспевания в настоящий момент – складские системы, базы данных и т.п.);
- поддерживающие или вспомогательные (необходимые, но не критические в настоящий момент – бухгалтерские программы, программы управления персоналом, текстовые редакторы, электронная почта и т.п.);
- стратегические (критические для достижения целей бизнес-стратегии – электронный обмен данными с партнерами, системы для анализа и прогнозирования рынка и т.п.);
- ИС для переходных задач или потенциальные (возможно, важные для достижения успеха в будущем – экспертные системы и т.п.).

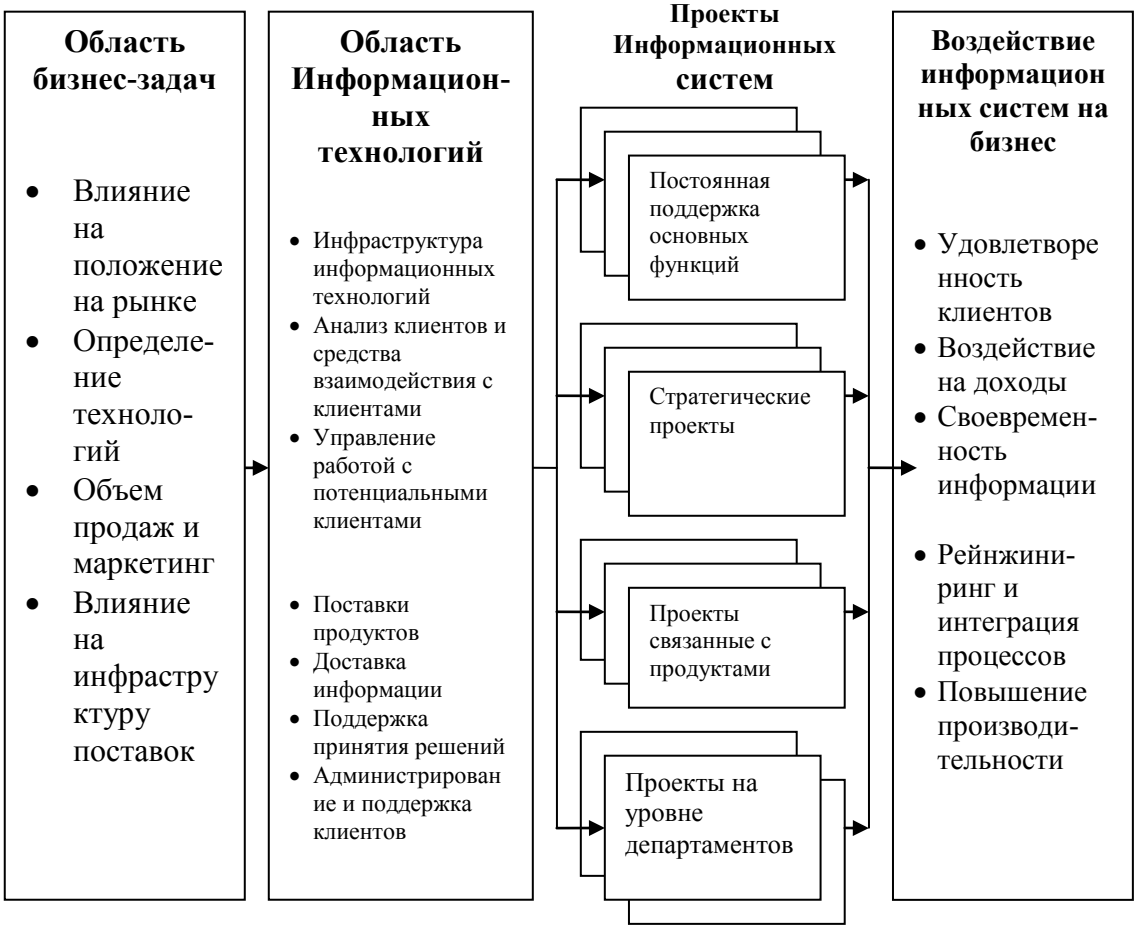


Рис. 6.1. Связь ИТ-проектов с областью бизнес-задач

Легче всего определить эффект для производственных систем – там он в большей степени материальный и легко может быть переведен в

«деньги». Что касается стратегических и потенциальных систем – там эффекты в большей степени нематериальны.

Среди наиболее часто встречающихся методик оценки эффективности вложений в ИТ можно условно выделить две большие группы методик: экономические и процессные. К первой группе отнесем методики, связанные с определением экономической (финансовой) эффективности внедрения, а ко второй – методики, пытающиеся увязать результаты внедрения с целями и задачами предприятия.

В экономических методах выделим три группы: затратные, инвестиционные и методы экономического анализа. Наиболее известным представителем затратных методов является метод ТСО (совокупной стоимости владения). Методология ТСО предполагает оценку стоимости приобретения, внедрения, эксплуатации, обучения, сопровождения и других затрат.

Минусом является то, что не учитывается результат от внедрения, ИТ не соотносятся со стратегическими целями бизнеса. Поэтому этот метод хорош для сравнения аналогов как при выборе поставщика ИТ-услуг, так и при рассмотрении способа создания ИС в организации. Другим представителем этих методов является EVA (экономически оправданная стоимость). Инвестиционные методы предлагают рассматривать проект по внедрению ИТ как инвестиционный проект. При этом используется типовая методика оценки инвестиционных проектов, включая ROI (отдача от инвестиций), NPV (чистый приведенный доход), IRR (внутренняя норма доходности), PB (срок окупаемости).

Эти методы дополняют друг друга и поэтому всегда используются в комплексе. Причиной, по которой сфера применения этих методов ограничена, заключается в трудности оценки будущих денежных потоков. По оценкам примерно от 60 до 75 % функционального объема проекта можно перевести в будущий денежный поток. И от 25 до 40 % остается на качественную и вероятностную оценку эффекта.

Методы экономического анализа позволяют оценить эффективность внедрения информационной системы на основе сравнения затрат и выгод внедрения ИС с альтернативными показателями. При этом в качестве критерия минимизации альтернативных издержек (opportunity costs) могут приниматься, в частности:

- сравнение показателей работы предприятия с использованием информационной системой и без нее;

- анализ выгодности других проектов по улучшению работы предприятия (например, использование аналогов) по сравнению с результатами внедрения данной ИС;
- сопоставление выгод от внедрения системы в денежном эквиваленте с доходом от инвестиций, например, в ценные бумаги или другие активы. Существенным недостатком экономического анализа является сложность сопоставления выгод от внедрения информационной системы с вложениями в другие активы.

В качестве одного из примеров процессных методов рассмотрим метод BSC (Balanced ScoreCard), позволяющий перевести стратегию развития ИТ в набор целей и мероприятий. Это инструмент стратегического управления, связывающий стратегические цели с процессами и повседневными действиями сотрудников на каждом уровне управления за счет полноценной системы ключевых показателей эффективности (KPI).

Предлагается развить практическое применение этого метода за счет построения многоуровневой детальной структуры «цели – задачи – подзадачи – функции/бизнес-процессы» «сверху-вниз» и агрегации отдельных экономических выгод «снизу-вверх» (пример такой структуры представлен в таблице 6.1).

Таблица 6.1.

Фрагмент дерева «Цели-факторы эффективности – количественные показатели эффективности»

Уровни дерева	Формирование «Дерева» – содержательная информатизация бизнес-процессов	Преобразование качественных факторов эффективности в количественные показатели «выгоды»
1	2	3
Бизнес-стратегия	Интенсификация использования ресурсов	Сокращение потребности в оборотных средствах
Цель	Сокращение размера дебиторской задолженности	Высвобождение оборотных средств в размере 430 млн.руб.
Задача	Контроль за сроками и условиями исполнения контракта	Сокращение среднегодовой величины дебиторски с 2,15 до 1,72 млрд.руб.

Окончание табл. 6.1.

1	2		3
Функция	Автоматизированное ведение реестра дебиторов и условий контрактов	Механизм регулярных уведомлений о наступлении срока платежа	Сокращение срока сбора дебиторской задолженности на 10 дней
Бизнес-процесс I	Автоматизированное ведение списка «критичных» дебиторов (нарушителей условий контрактов)		Уменьшение нарушений условий оплаты у 35 % покупателей
Бизнес-процесс I.1	Информирование отдела сбыта о «попадании» дебитора в список «критичных» и нарушениях контракта	Информирование «критичных» дебиторов о нарушении условий контракта	Повышение дисциплины оплаты поставок
Бизнес-решение	Прекращение поставок «критичным» дебиторам; отказ в возобновлении контрактов хроническим неплательщикам	Информирование грубых «нарушителей» о возбуждении претензийно-исковой процедуры	Увеличение оперативности реагирования компании на нарушение условий контрактов со стороны покупателей

Недостатком данного метода является трудоемкость и сложность этого процесса. Обязательным условием применения метода является наличие системы стратегического менеджмента на предприятии.

Выделяют также еще одну группу методов оценки экономического эффекта от ИТ-проекта – вероятностные: прикладная информационная экономика (Applied Information Economics) и справедливая цена опционов (Real Options Valuation, ROV).

Суть метода прикладной информационной экономики в том, чтобы для каждой из заявленных целей ИТ-проекта определить вероятность ее достижения и далее из нее вывести вероятность улучшений в бизнес-процессах компании. Например, позволяет ли проект по созданию ИС складского учета оптимизировать процессы материально-технического снабжения и сбыта? Насколько увеличится скорость принятия решения? В какой степени это повлияет на себестоимость продукции, оборачиваемость средств? Вероятностные методы нечасто используются для оценки будущего эффекта от ИТ-проекта. Метод

прикладной информационной экономики в большей степени субъективен, чем другие. Другой метод – метод справедливой цены опциона, наоборот, конкретен, но достаточно труден и требует большого времени для анализа.

К вероятностным методам оценки экономического эффекта от ИТ-проекта можно отнести и статистический метод. Оценка эффективности проводится на основании опыта внедрений различных ИТ-технологий, статистические показатели могут дать сформировать поле для принятия решений и сделать качественные выводы. В таблице 6.2 приведены наиболее часто встречающиеся и актуальные показатели.

Таблица 6.2

Среднестатистические мировые показатели эффекта от внедрения ИТ-систем

Показатель	Средние внедрения	Лучшие внедрения
Снижение количества задержек при поставках продукции заказчикам	90 %	97 %
Уменьшение неснижаемых остатков на складах материалов	30 %	45 %
Повышение оборачиваемости запасов	20 %	30 %
Сокращение НЗП	17 %	25 %
Повышение оборачиваемости средств в области реализации готовой продукции	12 %	21 %
Повышение производительности работников и оборудования	10 %	17 %
Снижение затрат на закупку материалов и комплектующих	4 %	6 %

Как правило, компании используют несколько методов оценки экономического эффекта от ИТ-проекта в совокупности. Опыт показывает, что в разных ситуациях ближе к истине оказываются разные методы.

6.2. Информационная система расчета экономической эффективности проектов информатизации

В настоящее время информационные технологии, являясь одним из важнейших элементов деятельности компаний, играют исключительно важную роль в обеспечении их конкурентоспособности. Во всем мире значительную долю в расходах компаний составляют информационные технологии.

Только в США объем расходов на компьютерную технику, телекоммуникации и оплату специалистов превышает 1 трлн. долл. в год. При этом даже, несмотря на значительное разочарование в информационных технологиях, ставшее следствием разорения большинства интернет-компаний в конце прошлого века, ежегодные расходы на информационные технологии продолжают расти из года в год.

Один из авторов учебника Захарова А.А. и выпускники Юргинского технологического института Томского политехнического университета задались целью создать информационную базу для разработки информационной системы расчета экономической эффективности проектов информатизации.

Были поставлены следующие задачи:

1) анализ методов оценки эффективности вложений в информационные технологии;
2) классификация методов оценки эффективности вложений в информационные технологии;
3) выбор методики расчета оценки эффективности вложений в информационные технологии;
4) определение информации для разрабатываемой системы.

Эффективность вложений в ИТ-технологии часто ставится под сомнение руководством компаний. Оценка эффективности вложений в информационные технологии является весьма актуальной проблемой. Многие руководители задают вопросы о том, насколько необходимо оценивать экономическую эффективность и какие методы и подходы при этом использовать.

Для упрощения оценки экономической эффективности была разработана информационная система расчета экономической эффективности проектов информатизации.

Разработанный информационный продукт предназначен для подсчета затрат на проекты информатизации, а самое главное – расчет эффективности данных затрат.

Информационная система выполняет следующие функции:

1) расчет затрат на внедрение информационных технологий как на этапе планирования, так и на этапе оценки фактически полученных результатов;

2) расчет показателей эффекта от внедрения информационных технологий;

3) расчет показателей эффективности внедрения информационных технологий.

Необходимые данные для решения описанных выше функций система получает из входной информации. Входной информацией являются:

- наименование проекта;
- этапы разработки и внедрения проекта;
- исполнители этапа разработки и внедрения проекта;
- трудоемкость каждого исполнителя на определенном этапе;
- количество исполнителей на каждом этапе разработки и внедрения;
- статьи затрат на разработку, внедрение и эксплуатацию;
- суммы затрат на разработку, внедрение и эксплуатацию.
- стоимостная оценка эффекта от внедрения ИС.

В системе имеются справочники и документы, в которые заносятся входные данные. Создан документ «Разработка», данные из которого заполняются из справочников «Проект», «Этапы разработки и внедрения» и «Должность». Поля «Трудоемкость» и «Количество исполнителей» заполняются вручную (рис. 6.2).

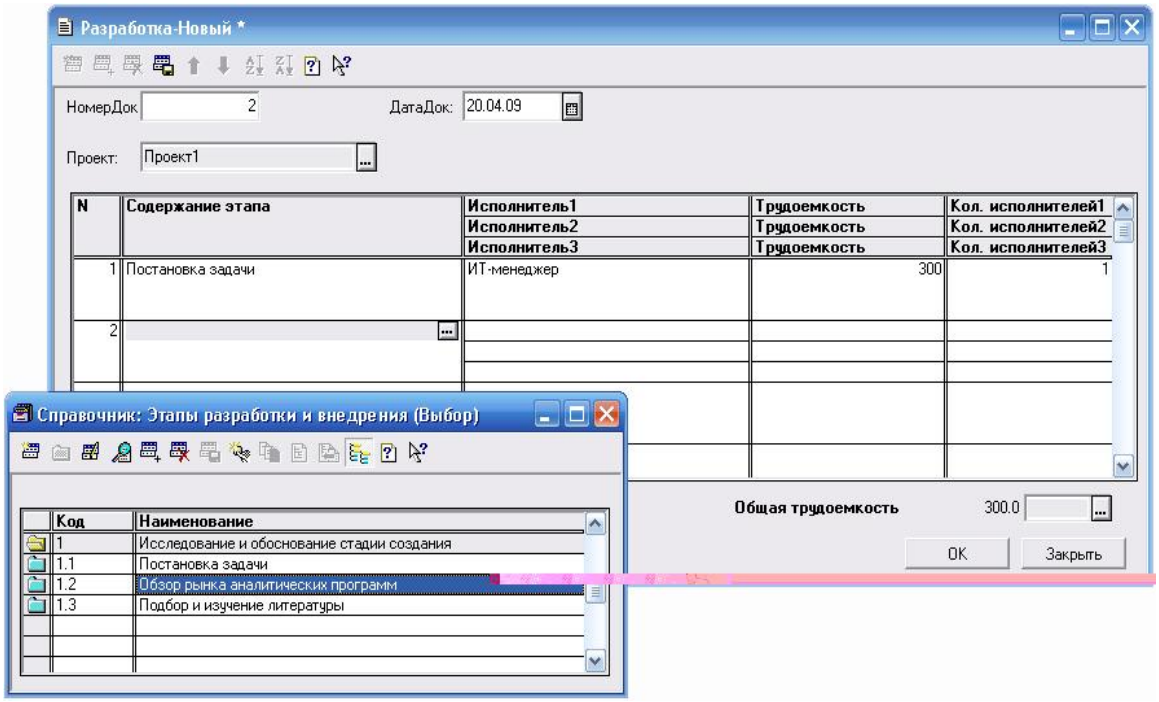


Рис. 6.2. Ввод данных из справочника «Этапы разработки и внедрения» в документ «Разработка»

Выходной информацией служат отчеты, созданные в результате работы системы. В данном случае это конкретные значения, рассчитанные с помощью методов оценки эффективности вложений в информационные технологии.

В системе выходной информацией служат следующие отчеты:

- календарный план проекта на этапе разработки;
- календарный план проекта на этапе внедрения;
- затраты на разработку;
- затраты на внедрение;
- затраты на эксплуатацию;
- эффективность проекта информатизации.

Для корректной работы системы расчета экономической эффективности необходимо воспользоваться существующими методами. В целом, можно выделить три основные группы методов, позволяющих определить экономическую эффективность: финансовые (они же количественные), качественные и вероятностные.

Данные методы применяются как к оценке планируемых результатов, так и фактических (таблица 6.3).

Таблица 6.3

Классификация методов оценки эффективности проектов информатизации

Методы оценки эффективности	Финансовые							Качественные				Вероят-ностные	
	REJ	NVP	IRR	Payback Period	EVA	TCO	TEI	BSC	Information	Portfolio	IT Scorecard	AIE	Real Options
Прогнозирование (на этапе планирования)													
Оценка фактических данных													

Были выбраны 4 метода оценки экономической эффективности для разрабатываемой информационной системы: чистый приведенный доход NPV, внутренняя норма доходности IRR, срок окупаемости проекта Payback period и совокупная стоимость владения TCO.

Первые три метода являются классическими при оценке эффективности и в совокупности хорошо зарекомендовали себя за годы существования. Данные методы предлагают рассматривать проект по внедрению ИТ как инвестиционный проект.

Эти методы дополняют друг друга и поэтому всегда используются в комплексе. ТСО же позволяет оценить совокупные затраты на информационные технологии (оборудование, инструментальные средства, процессы сопровождения информационных систем, а также действия конечных пользователей), анализировать их и соответственно управлять ИТ-затратами (бюджетом) для достижения наилучшей отдачи от ИТ в организации. ТСО представляет собой не просто отдельный интегральный показатель, но целую систему показателей, соответствующих различным статьям расходов.

В настоящее время авторы получили свидетельство о регистрации Информационной системы в отделе регистрации программ для ЭВМ, баз данных и топологий ИМС Федерального института промышленной собственности Федеральной службы по интеллектуальной собственности, патентам и товарным знакам.

Контрольные вопросы и задания.

1. Методы оценки экономической эффективности.
2. Параметры экономической эффективности.
3. Историческая роль информационных технологий и информационных систем в организации.
4. Экономические и процессные методики оценки эффективности вложений в ИТ.
5. Вероятностные методы оценки экономического эффекта от ИТ-проекта.

Тестовые задания для контроля знаний

1. *Операционная система – это:*

a) комплекс программ, который осуществляет управление компьютером, его ресурсами, обеспечивает диалог пользователя с компьютером, осуществляет запуск прикладных программ;
b) программа, обеспечивающая удобный и наглядный способ общения пользователя с ПК;
c) специальное устройство для общения пользователя с ПК.

2. *В функции операционной системы не входит:*

a) обеспечение выполнения системных и прикладных программ;
b) поддержка работы периферии компьютера;
c) организация и поддержка файловой системы;
d) выполнение тригонометрических операций.

3. *Структура программного обеспечения ПК:*

a) ОС и прикладные программы;
b) ОС WINDOWS и программы MS Office;
c) Системное ПО, инструментальное ПО, прикладное ПО.

4. *Файл – это:*

a) устройство для хранения информации;
b) внутренняя команда операционной системы;
c) поименованная совокупность логически связанных данных на магнитном носителе.

5. *Папка – это:*

a) место на диске для хранения программ, документов, объединенных по какому–либо критерию;
b) место хранения файла;
c) совокупность атрибутов файла.

6. *Что такое файловая система?*

a) совокупность программ, обеспечивающих работу внешней памяти;
b) часть операционной системы, управляющая размещением и доступом к файлам и папкам на диске;
c) расположение файлов на диске.

7. *Что такое спецификация файла?*

a) путь к файлу и полное имя файла;
b) тип файла;
c) имя файла с указанием расширения.

8. *Тестирование основных блоков и устройств компьютера – это функция:*

a) загрузки ОС;
b) командного процессора;
c) базовой системы ввода – вывода (BIOS).

9. *Как, используя шаблон имени файла: указать все файлы, имя которых начинается на SYS?*

a) SYS.*;
b) SYS*.*;
c) SYS.???.

10. *Можно ли создать папку в другой папке?*

a) да;
b) нет;
c) можно лишь в том случае, если папка, в которой создается новая, была тоже создана вами.

11. *Для чего служит «Проводник»?*

a) для работы с файлами и объектами;
b) для защиты от несанкционированного доступа в систему;
c) для редактирования текстов.

12. *Назначением значка «Мой компьютер» в Windows является:*

a) тестирование компьютера;
b) просмотр программ по настройке устройств данного компьютера;
c) просмотр технических и прочих параметров компьютера и его устройств, их настройка, а также доступ к папкам и файлам.

13. *Для чего служит «Корзина»?*

a) для хранения файлов пользователя;
b) для хранения системных файлов;
c) для временного хранения удаленных файлов.

14. *Что такое интерфейс пользователя?*

a) экран монитора;
b) комплекс программ, реализующих диалог пользователя с операционной системой;
c) специальное место в памяти компьютера, где хранятся программы пользователя.

15. *Что представляет собой буфер обмена?*

a) устройство, где хранится информация, доступная для всех программ;
b) папка, где хранятся скопированные файлы;
c) специальная область оперативной памяти, для временного размещения информации, доступной для всех прикладных программ Windows.

16. *В зоне заголовка окна не находятся кнопки системного меню:*

a) «Свернуть»;
b) «Переключиться в другое окно»;
c) «Развернуть–восстановить»;
d) «Закрыть».

17. *Что такое ярлык?*

a) значок, обозначающий файл;
b) специальный значок, который обеспечивает быстрый доступ к объектам;
c) файл, запускающий программу на выполнение.

18. *Windows 98/2000 – это:*

a) многозадачная, многопоточная, интегрированная ОС с графическим интерфейсом и с расширенными сетевыми возможностями;
b) ОС, обеспечивающая работу всех устройств ПК;
c) программа–оболочка, обеспечивающая удобный диалог пользователя с ПК.

19. *Что такое форматирование диска?*

a) создание структуры записи информации на поверхности диска: разметка дорожек, секторов, записи маркеров и другой информации;
b) удаление всей информации с диска;
c) запись на диск информации о структуре папок диска.

20. *Что такое драйвер устройства?*

a) устройство для управления принтером;
b) программа ОС для управления работой стандартными внешними устройствами;
c) программа, тестирующая основные блоки и устройства компьютера.

21. Операционная система (ОС) в качестве расширенной машины:

a) выполняет всю рутинную работу по управлению ресурсами компьютера;
b) выполняет всю высокоинтеллектуальную работу по управлению ресурсами компьютера;
c) частично выполняет рутинную работу по управлению ресурсами компьютера;
d) управляет программами.

22. Эволюция операционных систем происходила следующим образом:

a) мультипрограммирование, распределенная ОС, компиляторы, библиотеки служебных подпрограмм;
b) библиотеки служебных подпрограмм, компиляторы, распределенная ОС, мультипрограммирование;
c) библиотеки служебных подпрограмм, компиляторы, мультипрограммирование, распределенная ОС;
d) компиляторы, библиотеки служебных подпрограмм, мультипрограммирование, распределенная ОС.

23. Поддержка многозадачности:

a) наличие средств защиты информации каждого пользователя от несанкционированного доступа других пользователей;
b) распараллеливания вычислений в рамках одной задачи;
c) использование всего пула процессоров, разделяя их между системными и прикладными задачами;
d) число одновременно выполняемых задач ОС.

24. Современная ОС обладает следующими характеристиками:

a) однозадачная и многопользовательская;
b) невытесняющая многозадачность и;
c) поддержка многонитевости и многопользовательская;
d) однопользовательская и однозадачная.

25. Поддержка ОС многонитевости:

a) разделение процессорного времени между их отдельными ветвями (нитями) задачи;
b) разделение процессорного времени между задачами, а между их отдельными ветвями (нитями);
c) управление разделением совместно используемых ресурсов;
d) разделение процессорного времени между ресурсами.

26. Многопроцессорные системы требуют от операционной системы:

a) особой организации, с помощью которой только сама операционная система могла бы выполняться параллельно отдельными процессорами системы;
b) синхронизации доступа к разделяемым ресурсам, обнаружению отказов и динамической реконфигурации системы;
c) легкости переноса с компьютера одного типа на компьютер другого типа;
d) особой организации, с помощью которой сама операционная система, а также поддерживаемые ею приложения могли бы выполняться параллельно отдельными процессорами системы.

27. Многозадачные ОС пакетной обработки предназначены:

a) для управления различными техническими объектами, такими, например, как станок, спутник, научная экспериментальная установка или технологический процесс;
b) решения задач в основном вычислительного характера, не требующих быстрого получения результатов;
c) выбора программы исходя из текущего состояния объекта или в соответствии с расписанием плановых работ;
d) работы в условиях локальной сети.

28. В понятие «файловая система» не входит:

a) совокупность всех файлов на диске;
b) наборы структур данных, используемых для управления файлами;
c) комплекс системных программных средств, реализующих управление файлами;
d) данные, которые имеют жизненно важное значение для успешной работы ОС.

29. К какому из четырёх периодов относится фраза?

1. 1945–1955 гг.
2. 55–65 гг.
3. 65–80 гг.
4. 80 гг. – настоящее время.

a) Появились первые системы пакетной обработки, которые просто автоматизировали запуск одной программ за другой и тем самым увеличивали коэффициент загрузки процессора. Системы пакетной обработки явились прообразом современных операционных систем, они стали первыми системными программами, предназначенными для управления вычислительным процессом.

b) В этот период произошло разделение персонала на программистов и операторов, эксплуатационников и разработчиков вычислительных машин.

c) Для этого периода характерно также создание семейств программно-совместимых машин. Первым семейством программно-совместимых машин, построенных на интегральных микросхемах, явилась серия машин IBM/360. Программная совместимость требовала и совместимости операционных систем.

d) Программирование осуществлялось исключительно на машинном языке, все задачи организации вычислительного процесса решались вручную каждым программистом с пульта управления. Не было никакого другого системного программного обеспечения, кроме библиотек математических и служебных подпрограмм.

e) Компьютеры стали широко использоваться не специалистами, что потребовало разработки «дружественного» программного обеспечения, это положило конец кастовости программистов

f) *Мультипрограммирование* – это способ организации вычислительного процесса, при котором на одном процессоре попеременно выполняются несколько программ.

g) На рынке операционных систем доминировали две системы: MS-DOS и UNIX.

h) Операционные системы, построенные с намерением удовлетворить всем этим противоречивым требованиям, оказались чрезвычайно сложными. Они состояли из многих миллионов ассемблерных строк, написанных тысячами программистов, и содержали тысячи ошибок, вызывающих нескончаемый поток исправлений. В каждой новой версии операционной системы исправлялись одни ошибки и вносились другие.

i) Появился новый тип ОС – системы разделения времени.

j) Сетевая ОС не имеет фундаментальных отличий от ОС однопроцессорного компьютера. Она обязательно содержит программную поддержку для сетевых интерфейсных устройств (драйвер сетевого адаптера), а также средства для удаленного входа в другие компьютеры сети и средства доступа к удаленным файлам, однако эти дополнения существенно не меняют структуру самой операционной системы.

30. Что такое программное обеспечение?

a) программы, управляющие работой ПК;
b) совокупность программ обработки данных и необходимых для их эксплуатации документов;
c) программы, находящиеся в памяти ПК.

31. Системное программное обеспечение (System Software):

a) комплекс программ для решения задач определенного класса в конкретной предметной области;
b) совокупность программ и программных комплексов для обеспечения работы компьютера и сетей ЭВМ;
c) комплекс программ для тестирования компьютера.

32. Какие программы не относятся к антивирусным?

a) программы-фаги;
b) программы сканирования;
c) программы-ревизоры;
d) программы-детекторы.

33. Как вирус может появиться в компьютере?

a) переместиться с диска;
b) при решении математической задачи;
c) при подключении к компьютеру модема;
d) самопроизвольно.

34. Компьютерным вирусом является:

a) программа проверки и лечения дисков;
b) любая программа, созданная на языках низкого уровня;
c) программа, скопированная с плохо отформатированного диска;
d) специальная программа небольшого размера, которая может приписывать себя к другим программам, она обладает способностью «размножаться».

35. Какие программы не относятся к антивирусным?

a) программы-фаги;
b) программы сканирования;
c) программы-ревизоры;
d) программы-детекторы.

36. Интерфейс пользователя:

a) это комплекс программ, реализующий диалог пользователя с компьютером;
b) совокупность программ для решения задач различных предметных областей;
c) комплекс специальных программных средств для управления загрузкой, запуском и выполнением пользовательских программ.

37. Буфер обмена:

a) это пространство оперативной памяти для временного размещения данных;
b) часть ПЗУ для временного размещения данных;
c) специальная программа для временного хранения информации.

38. Расширение файла указывает:

a) на размер файла;
b) тип файла;
c) время создания файла.

39. Объектами Windows являются:

a) «Мой компьютер»;
b) «Проводник»;
c) «Корзина»;
d) «Электронная таблица».

40. Что такое OLE-технология?

a) создание графических документов;
b) технология объединения в одном документе фрагментов из разных приложений Windows;
c) технология создания новых объектов в Windows.

41. Табличный процессор:

a) это специальный микропроцессор для работы с матрицами чисел;

b) любая таблица с пронумерованными строками и столбцами;
c) электронная таблица, представляющая собой матрицу ячеек;
d) электронная база данных с таблицами.

42. Что такое мультимедиа?
a) возможность решать на ПК много задач одновременно;
b) соединение в единый комплекс различной информации: текстовой, графической, видео, звуковой, управляемой в интерактивном режиме;
c) возможность передавать информацию по телефонным каналам.

43. Поле базы данных должно иметь:
a) один тип данных;
b) разные типы данных;
c) только числовой.

44. Ключ в БД:
a) поле, однозначно характеризующее запись;
b) числовой тип данных;
c) критерий сортировки БП.

45. Сортировка в Excel имеет:
a) 1 уровень;
b) произвольное количество уровней;
c) 3 уровня.

46. Фильтрация в БД:
a) просмотр отдельных полей;
b) просмотр записей БД, удовлетворяющим каким-либо условиям;
c) просмотр записей БД.

47. Критерий (диапазон условий) применяется:
a) в справочнике;
b) только в расширенном фильтре;
c) в расширенном фильтре и функциях БД.

48. Критерий (диапазон условий) в БД:
a) это формула для выбора информации;
b) отдельная таблица с именами полей и правилом выбора информации;

c) отдельная таблица с правилом выбора информации.

49. Для формирования промежуточных итогов в БД необходимо:
a) ключевой реквизит разместить в 1 колонке;
b) создать критерий;
c) предварительно отсортировать БД.

50. Основным элементом БД в Excel является:
a) запись;
b) поле;
c) таблица.

51. Структура БД изменится:
a) если добавить, удалить запись;
b) отредактировать запись;
c) добавить, удалить поле.

52. Типы структур данных:
a) линейная, иерархическая, сетевая, реляционная;
b) линейная и табличная;
c) произвольные.

53. База данных:
a) это информационные структуры, хранящиеся во внешней памяти;
b) программные средства, позволяющие организовывать информацию в виде таблиц;
c) программные средства, обрабатывающие табличные данные;
d) программные средства, осуществляющие поиск информации;
e) информационные структуры, хранящиеся в оперативной памяти.

54. В реляционной базе данных информация организована в виде:
a) сети;
b) иерархической структуры;
c) файла;
d) дерева;
e) двумерных таблиц.

55. Алгоритм:

a) это последовательность действий по преобразованию исходной информации в конечный результат за конечное число шагов;
b) перечень команд, понимаемых компьютером;
c) техническое задание на программный продукт.

56. Программа:

a) это система правил, описывающая последовательность действий, которые необходимо выполнить для решения задачи;
b) указание на выполнение действий из заданного набора;
c) область внешней памяти для хранения текстовых, числовых данных и другой информации;
d) последовательность команд, реализующая алгоритм решения задачи.

57. Программа-интерпретатор выполняет:

a) поиск файлов на диске;
b) пооператорное выполнение программы;
c) полное выполнение программы.

58. Программа-компилятор:

a) переводит исходный текст в машинный код;
b) формирует текстовый файл;
c) записывает машинный код в форме загрузочного файла.

59. Информационная технологи:

a) это процесс, использующий совокупность средств и методов сбора, обработки и передачи данных для получения информации нового качества о состоянии объекта, процесса или явления;
b) алгоритм, реализующий задачу о состоянии объекта, процесса или явления;
c) программное средство обработки и передачи данных для получения информации нового качества о состоянии объекта, процесса или явления.

60. Структурное программирование – это стиль программирования, основанный на принципах использования:

a) компиляции, интерпретации, транслитерации;
b) парадигм – наследование, полиморфизм, инкапсуляция;
c) базовых структур – последовательность, ветвление, цикл.

61. В состав интегрированной системы программирования входят:

a) редактор связей;
b) калькулятор;
c) текстовый редактор;
d) графический редактор.

62. Очень короткая программа, которая находится в первом секторе системного диска:

a) это ядро операционной системы;
b) BIOS;
c) модуль оперативной системы;
d) загрузчик операционной системы.

63. Служебные (сервисные) программы предназначены:

a) для согласования работы внешних устройств и компьютера;
b) выполнение ввода, редактирования и форматирования текстов;
c) автоматизации проектно-конструкторских работ;
d) диагностики состояния и настройки вычислительной системы.

64. В некоторой папке хранятся файлы, созданные в MS Word, MS Excel, MS Access, MS PowerPoint:

a) tab.doc
b) acc.xls
c) xls.doc
d) doc.ppt
e) present.mdb
f) abc.itf

Количество файлов, созданных в *Word, Excel, Access, PowerPoint,* соответственно равно:

a) 3,1,0,2;
b) 4,1,1,0;
c) 2,2,1,1;
d) 3,1,1,1.

65. Дан фрагмент электронной таблицы.

В ячейку C3 введена формула

=ЕСЛИ (A2+B2<12;0;МАКС (A2:D2)).

Сравните значения в ячейках C3 и B5.

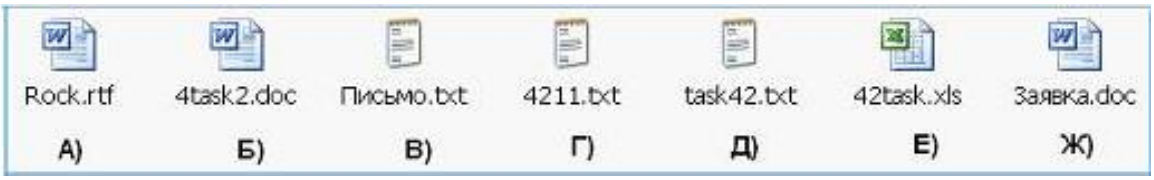
	A	B	C	D	E
1	1		2		ДА
2	3	9		24	ДА
3	0,5				ДА
4				НЕТ	НЕТ
5	4	0			

a) значение в ячейке С3 больше значения в ячейке В5;
b) сравнение недопустимо, так как полученные данные имеют разный тип;
c) значение в ячейке С5 равно значению в ячейке В5;
d) значение в ячейке С5 меньше значения в ячейке В5.

66. В функции служебных приложений ОС Windows не входит:

a) нахождение и устранение дефектов файловой системы;
b) согласование работы внешних устройств и компьютера;
c) архивация данных;
d) очистка дисков.

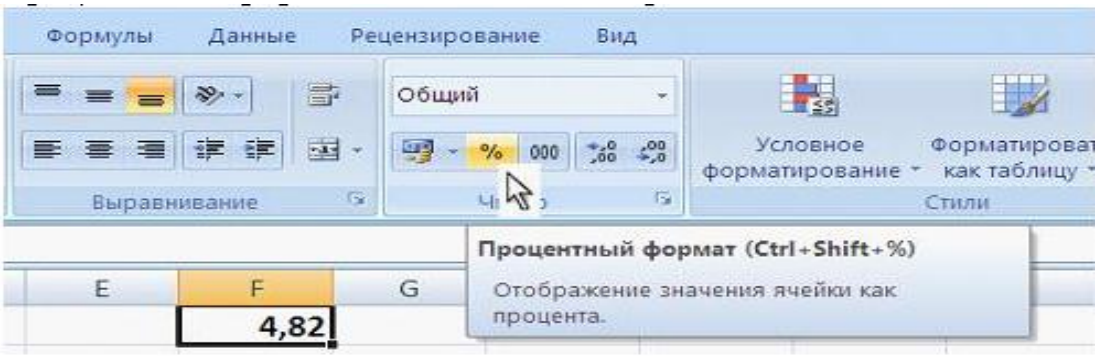
67. В одной из папок жесткого диска имеются файлы.



После проведения сортировки «по имени» в порядке убывания файлы расположатся в порядке:

a) В), Ж), Д), А), Г), Е), Б);
b) Б), Е), Г), А), Д), Ж), В);
c) Д), А), Г), Е), Б), В), Ж);
d) Б), Ж), А), Г), Д), В), Е).

67. В ячейку F1 введено число 4,82. Если нажать на кнопку Процентный формат, то это число примет вид:



a) 482,00 %;
b) 48,2 %;
c) 482 %;
d) 4,82 %.

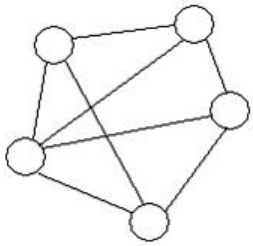
68. В некоторой папке содержатся файлы.

После проведения сортировки по типу последним окажется файл:
a) primer6.ppt;
b) primer1.pptx;
c) primer3.pas;
d) primer2.pdf.

69. Языками программирования для экспертных систем не являются:
a) C и C++;
b) Lisp и Prolog;
c) HTML и XML;
d) Фортран и Паскаль.

70. *Сетевая база данных представляет собой такую организацию данных, при которой:*
a) связи между данными распределяются по уровням, причем элементы нижнего уровня входят в состав элементов более высокого уровня;
b) связи между данными описываются в виде совокупности нескольких двумерных таблиц;

c) связи между данными описываются в виде двумерной таблицы;
d) связи между данными носят произвольный характер.



71. Перемещаясь из одной папки в другую, пользователь последовательно посетил папки:

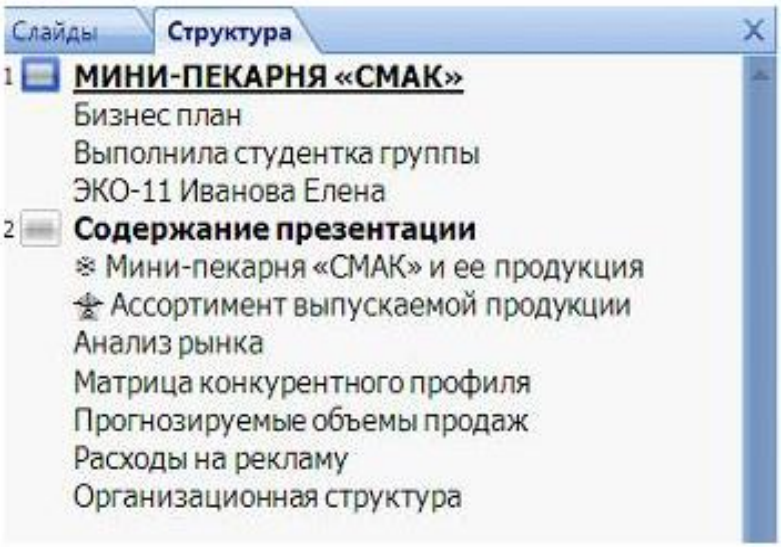
DOC, USER, SCHOOL, D:\, LETTER, INBOX.

При каждом перемещении пользователь либо спускался в папку на уровень ниже, либо поднимался на уровень выше.

Укажите полное имя папки, из которой начал перемещение пользователь:

a) D:\LETTER\INBOX;
b) D:\SCHOOL\USER\DOC;
c) D:\DOC\USER\SCHOOL;
d) D:\DOC.

72. Режим структуры работы с презентацией позволяет:



a) осуществлять показ презентации;
b) настраивать эффекты анимации;
c) добавлять новый текст на слайд или редактировать существующий;

d) производить хронометраж слайд-фильма.

73. *Представлена база данных «Волшебные страны». После проведения сортировки сведения о НАРНИИ переместятся на одну строку вниз. Это возможно, если сортировка будет проведена в порядке:*

a) возрастания по полю НАСЕЛЕНИЕ;
b) возрастания по полю СТРАНА;
c) убывания по полю ПЛОЩАДЬ;
d) возрастания по полю ПЛОЩАДЬ.

Волшебные страны : таблица

Страна	Население	Площадь
НАРНИЯ	148	46,9
ОЗ	155	95,3
ШВАМБРАНИЯ	132	53,5
ЛУКОМОРЬЕ	199	47,7
ЗАЗЕРКАЛЬЕ	211	76,2
ЗЕМНОМОРЬЕ	325	108,4
	0	0

74. Выделенная часть Панели задач имеет название:

a) панель инструментов;
b) область уведомлений;
c) панель быстрого запуска;
d) панель состояния.

75. В среде СУБД Access создать новую форму можно на основе:

a) только таблицы;
b) только запроса;
c) отчета;
d) таблицы или запроса.

Список сокращений

АРМ – автоматизированное рабочее место.
АСНИ – автоматизированные системы научных исследований.
АСУ – автоматизированные системы управления.
АСУП ТП – автоматизированные системы управления производством и технологическими процессами.
БД – база данных.
ВР – виртуальная реальность.
ВС – вычислительная система.
ДО – дистанционное обучение (образование).
ИАД – интеллектуальный анализ данных.
ИИ – искусственный интеллект.
ИС – информационная система.
ИТ – информационная технология.
КДИ – компьютерные деловые игры.
КОББИ – компания Компьютерное обучение бизнесу.
ООП – объектно-ориентированный подход.
ОС – операционная система.
ОСРВ – операционные системы реального времени.
ПЗУ – постоянное запоминающее устройство.
ПК – персональный компьютер.
ПО – программное обеспечение.
ПП – программный продукт.
ППО – прикладное программное обеспечение.
ППП – пакеты прикладных программ.
ПС – программные средства.
ПЭВМ – персональная электронная вычислительная машина.
САПР – системы автоматизированного проектирования.
СОД – система обработки данных.
СМИ – средства массовой информации.
СПО – системное программное обеспечение.
СППР – системы поддержки принятия решений.
СУБД – система управления базами данных.
ТЭП – Технико-экономическое планирование.
ТСО – технические средства обучения.
ФСА – функциональная структура алгоритма.
ЭВМ – электронная вычислительная машина.
ЭИС – экономические информационные системы.
ЭП – электронная почта.
ЭС – экспертная система.

Алфавитный указатель

Рекомендуемая и использованная литература

1. Автоматизированные информационные технологии в экономике: Учебник/ М.И.Семенов, И.Т.Трубилин В.И. Лойко, Т.П.Барановская; Под общ. ред. И.Т.Трубилина. – М.: Финансы и статистика, 2001.– С.13–45, 60–64.
2. Информатика для юристов и экономистов/ Под редакцией С.В. Симиновича – Спб.: Питер, 2006.–688 с.: ил.
3. Информатика: Учебник – 3-е переработ.изд./ Под ред.проф. Н.В.Макаровой. – М.: Финансы и статистика, 2001. – 768 с.
4. Информационные технологии управления: Учебное пособие/ Под ред. Ю.М.Черкасова. – М.: ИНФРА–М, 2001.– С. 5–33.
5. Леонтьев В.П. Новейшая энциклопедия. 1000 лучших программ. Настольная книга пользователя. – М.:ОЛМА-ПРЕСС образование, 2004. – 752 с.: ил.
6. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 4-е изд. – СПб.: Питер, 2010. – 944 с.: ил.
7. Олифер В.Г., Олифер Н.А. Сетевые операционные системы: Учебник для вузов. 2-е изд. – СПБ.: Питер, 2008. – 669 с.: ил.
8. Теория экономических информационных систем: Учебное пособие/ Исакова А.И. – Томск, 2001. – 124 с.
9. Учеб. пособие для студентов пед. вузов/ Под ред. Е.К. Хеннера, А.В. Могилев, Н.Н. Пак, Е.К.Хеннер – М.: Изд.центр «Академия», 2000 г. – 816 с.
10. Экономическая информатика/ Под ред. П.В. Конюховского и Д.Н.Колесова. – СПб.: Питер, 2000. – С.14–30.
11. Каймин В.А. Информатика: Учебник. – 4-е изд. – М.: ИНФРА-М, 2003. – 285 с. – (Высшее образование).
12. Кадушин А., Михайлова Н. Оценить нельзя верить // IT-форум, 2003. – № 5. – С. 30–37.
13. Лейн Д. Просвещенный ИТ-директор: Лучшие примеры из практики Кремниевой долины / Дин Лейн; Пер. с англ. – М.: Альпина Бизнес Букс, 2005.
14. Коптелов А. BSC для ИТ // CIO, 2008. – № 2. – С. 66–69.

Учебное издание

ЗАХАРОВА Александра Александровна,
МОЛНИНА Елена Владимировна,
ЧЕРНЫШЕВА Татьяна Юрьевна

ИНФОРМАТИКА И ПРОГРАММИРОВАНИЕ: ПРОГРАММНЫЕ СРЕДСТВА РЕАЛИЗАЦИИ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ

Учебник

Научный редактор
доктор технических наук,
профессор *А.А. Мицель*
Редактор *Т.В.Казанцева*
Компьютерная верстка *Е.В. Молнина*
Дизайн обложки *С.В. Сахаров*

Подписано к печати 07.02.12. Формат 60x84/16. Бумага «Снегурочка.
Печать XEROX. Усл.-печ. л. 18,94. Уч.–изд. л. 17,16
Заказ 1465.Тираж 100 экз.

Томский политехнический университет
Система менеджмента качества
Издательства Томского политехнического университета сертифицирована
NATIONAL QUALITY ASSURANCE по стандарту BS EN ISO 9001:2008

ИЗДАТЕЛЬСТВО ТПУ. 634050, г. Томск, пр. Ленина, 30.
Тел./факс:8(3822) 56-35-35, www.tpu.ru