Министерство образования Республики Беларусь

Брестский государственный технический университет

Кафедра “ ЭВМ и Систем “

Реферат

На тему: “ История развития прикладного программного

обеспечения персонального компьютера “

По дисциплине: “ Введение в специальность “

Специальности Т.10.01

Выполнил:

ст. I курса гр. АС-14

Смаль Д. А.

Проверил:

Доцент Савицкий Ю.В.

Брест 2001

Оглавление:

1. Введение (общее понятие ПО для ПК)-----------------------------------------стр. 3-4
2. Прикладное программное обеспечение----------------------------------------стр.4-8
3. Пакеты прикладных программ (ППП)-----------------------------------------стр.8-19
   1. Понятие ППП----------------------------------------------------------------стр.8-12
   2. Структура и основные компоненты ППП------------------------------стр.13-16
   3. Этапы развития ППП-------------------------------------------------------стр.16-19
4. Примеры прикладных программ-----------------------------------------------стр.20-22
5. Заключение-------------------------------------------------------------------------стр.22-23
6. Список использованной литературы------------------------------------------стр. 24

**1.Введение. Общее понятие ПО для ПК.**

Персональный компьютер, как известно, является универсальным устройством для обработки информации. Персональные компьютеры могут выполнять любые действия по обработке информации. Для этого необходимо составить для компьютера на понятном ему языке точную и подробную последовательность инструкций – программу, как надо обрабатывать информацию.

Меняя программы для компьютера, можно превратить его в рабочее место бухгалтера или конструктора, дизайнера или ученого, писателя или агронома. Кроме того, тенденция понижения стоимости компьютерной техники при одновременном росте ее производительности привела к тому, что компьютеры становятся предметом домашнего обихода, как, например, телевизор или холодильник, что расширяет сферу применения ПК еще больше. Соответственно, требуется все более разнообразное программное обеспечение для решения задач в новых областях применения ПК. Непрерывное повышение мощности персональных компьютеров, периферийных устройств, а также развитие средств связи дает разработчикам программного обеспечения все больше возможностей для максимально полного удовлетворения запросов конечных потребителей. Это и ставший стандартом графический интерфейс для любого ПО, и внедренные возможности для отправки документов и данных с помощью Интернет непосредственно из прикладной программы (Microsoft Word 2000, Excel 2000, Access 2000 и др.), и возможность использования компьютера как хранилища информации благодаря появлению новых видов накопителей большой емкости и малым временем доступа к данным, а также многие другие возможности и сервисные функции.

При своем выполнении программы могут использовать различные устройства для ввода и вывода данных, подобно тому, как человеческий мозг пользуется органами чувств для получения и передачи информации. Сам по себе ПК не обладает знаниями ни в одной области своего применения, все эти знания сосредоточены в выполняемых на нем программах. Поэтому часто употребляемое выражение «компьютер сделал» означает ровно то, что на ПК была выполнена программа, которая позволила выполнить соответствующее действие.

В настоящее время весь комплекс ПО делится на системные и пользовательские программы. Системное программное обеспечение выполняет функции «организатора» всех частей ПК, а также подключенных к нему внешних устройств. Программы для пользователей служат для выполнения каких – либо конкретных задач во всех сферах человеческой деятельности.

В своем реферате я хочу рассказать про развитие прикладного программного обеспечения персонального компьютера, пакетов прикладных программ (ППП), а также про использование прикладных программ в жизни каждого пользователя.

**2.Прикладное программное обеспечение.**

**1. Основные понятия.**

Прикладное ПО представляет собой распространенный класс программных продуктов, представляющий наибольший интерес для пользователя.

*Прикладное ПО предназначено для решения повседневных задач обработки информации:*

* создания документов, графических объектов, баз данных;
* проведения расчетов;
* ускорения процесса обучения;
* проведения досуга.

Все эти программы пишутся по принципу максимального удобства для пользователя, обладают дружественным интерфейсом (средствами общения “компьютер-человек”, “человек-компьютер”). В настоящее время эти программы требуют высокопроизводительных, обладающих большими ресурсами компьютеров, хотя каждый программист стремится сделать свою программу в первую очередь наиболее доступной.

*Примеры прикладных программ:*

* ТР, ГР, СУБД, ЭТ;
* музыкальные редакторы;
* обучающие программы (помогают изучать различные предметы);
* программы тестирования (используются для проведения контроля по предметам);
* программы статистических расчетов;
* компьютерные игры;
* интегрированные пакеты (программы, объединяющие несколько типов прикладных задач: ТР, СУБД и др.)
* телекоммуникационные и сетевые программы.

***2. Пpогpаммы для работы с текстами.***

*Текстовые редакторы и издательские системы* - это программы для набора, редактирования и подготовки к печати любых документов от маленьких заметок или договора на одну страничку до многотомной энциклопедии или цветного иллюстрированного журнала.

Для повышения грамотности выпускаемых книг, газет и других изданий предназначены *программы проверки правописания*.

Есть *программы-переводчики* - с английского, немецкого, французского и других языков на русский и обратно.

*Программы-словари* дают не только письменный перевод введенных слов, но и устный, что облегчает понимание и усваивание слов написанных на иностранном языке.

Благодаря *программам распознавания образов* можно использовать сканер для ввода не только картинок, но и текстов.

Текстовые редакторы (MS Word, Лексикон, Слово и Дело).

Издательские системы (Corel Ventura, Page Maker).

Переводчики (Stylus).

Словари (Lingvo)

Распознаватели текстов (Fine Reader).

***3. Программы для работы с графикой.***

Программы для работы с графикой предназначены для создания графических объектов, мультфильмов, видеоклипов и прочих анимационных объектов.

Гpафические редакторы (Photoshop, Corel Draw, Paintbrush).

Аниматоpы (Alias Power Animator).

Пpогpаммы для обработки 3d графики (3d Studio).

Пpогpаммы для обработки видео.

***4. 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.

Pедактоpы (Scream Tracker).

Плейеpы (Jet Audio).

***5. Программы для делопроизводства.***

Существует программы, позволяющие создавать БД, редактировать БД, выполнять различные операции в БД. Эти программы называются системы управления базами данных (СУБД).

Программы, которые позволяют автоматизировать вычисления над данными, представленными в форме прямоугольных таблиц, называются электронными таблицами (ЭТ).

Отдельный обширный класс программных продуктов - финансовые, банковские, бухгалтерские программы, программы для ведения офисной документации, программы планирования финансовой, коммерческой и производственной деятельности, предназначенные в своей основной массе для людей специализирующихся в экономической деятельности.

Системы управления базами данных (FoxPro, Clipper, Access).

Электронные таблицы (Lotus 1-2-3, MS Excel).

Бухгалтерские (1C: бухгалтерия).

Математические (MathLab).

Конструкторские (AutoCAD).

***6. Игры.***

Игровые программы позволяют не только развлекаться, но и получать некоторые новые полезные знания.

***7. Телекоммуникационные и сетевые программы.***

Коммуникационные программы предназначены для обслуживания модема (и факс-модема). Всемирная сеть Интернет позволяет получить доступ к компьютерам, расположенным в разных частях света. Программы, работающие с локальной сетью, позволяют объединить все компьютеры класса, института или какой либо организации для совместной работы или для использования одних и тех же ресурсов.

Интеpнет-бpоузеpы (Netscape Navigator).

Теpминалы (TeleMax, Hyper Terminal).

Почтовые редакторы (GoldED).

***8. Интегрированные пакеты программ .***

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.

Пакет MS Works

***9. Обучающие и тестирующие программы.***

Обучающие и тестирующие программы предназначены для получения новых знаний, для тестирования по различным дисциплинам, для приема экзаменов,зачетов и т.д.

Обучающая система (TeachPro Word, TeachPro Windows95,

TeachPro Windows 98, TeachPro Excel)

Тестирующие программы (Test).

**3.Пакеты прикладных программ ( ППП )**

* 1. ПОНЯТИЕ ППП

Многочисленные программные средства для решения различных типов вычислительных задач можно разделить на 4 группы:

- отдельные прикладные программы;

- библиотеки прикладных программ;

- пакеты прикладных программ;

- интегрированные программные системы. Рассмотрим по порядку каждую из этих групп. Отдельная прикладная программа пишется, как правило, на некотором универсальном языке программирования (Паскаль и т.п.) и предназначается для решения конкретной прикладной задачи. Примерами могут служить программа решения системы линейных алгебраических уравнений тем или иным численным методом, прог­рамма вычисления собственных значений матрицы и т. д. Авторами таких программ являются прикладные программисты, специализирую­щиеся: в соответствующих предметных областях. Прикладная прог­рамма может быть реализована в" виде набора модулей, каждый из которых выполняет некоторую самостоятельную функцию. Например, программа вычисления собственник значений матрицы может вклю­чать модули, - реализующие преобразования матрицы из одной, формы представления в другую ввод и вывод данных, обработку аварий­ных ситуаций с выдачей диагностических сообщений пользователю и другие действия.

Библиотека представляет собой набор отдельных программ, каждая из которых решает некоторую прикладную задачу или выпол­няет определенные вспомогательные функции (управление памятью, обмен с внешними устройствами и т.п.). Библиотеки программ за­рекомендовали себя эффективным средством решения вычислительных задач. Они интенсивно используются при решении научных и инже­нерных задач с помощью ЭВМ. Условно их можно разделить на библиотеки широкого применения и специализированные библиотеки.

Программы, входящие в состав библиотеки широкого примене­ния, предназначены для решения задач из различных предметных областей.

Специализированные библиотеки ориентированы на решение от­дельных, порой достаточно узких, классов задач.

Переход от разработки отдельных прикладных программ к соз­данию библиотек программ поставил перед разработчиками ряд проблем как системного, так и прикладного характера. К числу основных проблем, возникающих на различных этапах конструирова­ния библиотеки, относятся проблемы систематизации, документиро­вания, тестирования и переносимости.

Проблема систематизации состоит в разбиении библиотеки на разделы и подразделы в соответствии с классификацией зада предметной области и методов их решения. Сюда входит выбор еди­ных правил наименования программ (с учетом модификации и развития библиотеки), единых форм представления и наименования математически сходных объектов, единой схемы контроля ошибок и т. д.

Проблема документирования заключается в составлении единых правил описания программ библиотеки. Наличие качественной доку­ментации существенно упрощает доступ к отдельным программам, организацию взаимодействия между программами, включение новых программ. Важную роль при решении указанной проблемы играют средства автоматизации документирования, обеспечивающие широкое применение шаблонов (для титульных листов, фрагментов текста и т. п.), использование текстов программ библиотеки для автоматизи­рованного составления документации и т. д.

Тестирование библиотеки состоит в проверке программ на спе­циально подготовленных тестовых данных. Результаты тестирования в большой мере зависят от правильности и полноты набора тестов. Тестирование, в частности, включает проверку соответствия текс­та программы выбранному стандарту языка программирования, опре­деление области применимости программы и качестве диагности­ческого аппарата, выявление разнообразных количественных характеристик, программы (скорость работы, точность получаемых результатов и т. п.) сравнение с другими программами для реше­ния той же задачи.

Проблема переносимости состоит в разработке методов и средств, обеспечивающих возможность использования программ биб­лиотеки в различных вычислительных условиях (на различных.типах ЭВМ, в различных операционных системах и т.д.) Эта проблема включает в себя такие аспекты, лак следование стандарту языка программирования, организация работы с машинно-зависимыми конс­тантами, создание инструментальных средств, позволяющих автоматизировать перенос программ из одной вычислительной среды в другую. Доступ к программам библиотеки осуществляется с использова­нием штатных средств того или иного языка программирования. Разработка библиотек программ обычно осуществляется силами прикладных программистов. При этом нередко в библиотеку включаются программы, написанные в. разное время и разными авторами. Отсутствие в составе библиотеки специализированного системного обеспечения позволяет в большинстве случаев при ее конструиро­вании обойтись без какой-либо существенной помощи системных программистов (они могут привлекаться, например, для написания лишь отдельных служебных программ).

Таким образом, характерной особенностью библиотек программ является отсутствие проблемно-ориентированного входного языка и достаточно развитого системного обеспечения. Как правило, библиотеки программ ориентированы на типовые задачи предметной об­ласти и не содержат средств решения специфических прикладных задач (в первую очередь это относится к библиотекам широкого применение программы которых могут использоваться для решения задач из различных предметных областей).

Перейдем теперь к рассмотрению пакетов прикладных программ (ППП) как самостоятельной формы прикладного программного обес­печения. Для этого прежде всего необходимо уточнить само поня­тие пакета. В настоящее время не существует признанной всеми специалис­тами единой точки зрения по этому вопросу. Отсутствует также единая терминология в пакетной проблематике. Это объясняется прежде всего новизной данного научного направления, которое сложилось, в основном за последние 30 лет (приблизительно с начала 70-х годов). Кроме того, различные определения ППП расс­матривают это понятие с разных точек зрения, выделяя те или иные функциональные или структурные особенности пакетов. ППП определяется и как совокупность прог­рамм для решения определенного класса задач, к которой обраща­ются при помощи простой символики (языка) и как совокупность программ, совместимых по структуре данных, способам управления, объединяемых общностью функционального назначениями представляющих со­бой средство решения класса задач определенным кругом пользова­телей. При этом 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. В данном случае не делается попытки выделить ППП среди других форм программного обеспечения ЭВМ.

Будем считать пакетом программ любой комплекс, ориентированный на решение некоторого класса задач. Формально такое определение не исключает из числа пакетов и библиотеки программ. Однако сложившееся на сегодняшний день представление о ППП как о самостоятельной форме программного обеспечения, позволяет указать на ряд характерных отличительных особенностей пакетов.

Одной из главных особенностей является ориентация ППП не на отдельную задачу, а на некоторый класс задач, включающий и спе­цифические задачи предметной области. Отсюда следует необходи­мость, модульной организации ППП как основного технологического принципа его конструирования. Суть этого принципа состоит в оформлении общих фрагментов используемых алгоритмов в виде са­мостоятельных модулей. Решение сформулированной пользователем задачи осуществляется некоторой "цепочкой" таких модулей.

Другой особенностью ППП является наличие в его составе специализированных языковых средств, обеспечивающих удобную работу пользователя с пакетом. Как правило, развитый пакет обладает несколькими входными языками, ориентированными на выполнение различных функций и различные типы пользователей. Язык может предназначаться для формулировки исходной задачи, описания ал­горитма решения и начальных данных, организации доступа и поддержания базы данных или информационной базы ППП, разработки программных модулей, описания модели предметной области, управ­ления процессом решения в диалоговом режиме и других целей.

Еще одна особенность ППП состоит в наличии специальных сис­темных средств, обеспечивавших принятую в предметной области дисциплину работы. К их числу относятся специализированные бан­ки данных, средства информационного обеспечения, средства взаи­модействия пакета с операционной системой и т. п.

Наконец, 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 кразличным предметным областям.

Следует указать на отсутствие четких и однозначных границ между перечисленными формами прикладного программного обеспече­ния. Так, отдельная прикладная программа, ориентированная на решение класса задач и оформленная в виде совокупности модулей может рассматриваться как библиотека или даже пакет программ несмотря на отсутствие специализированных языковых и системных средств.

Переход от создания библиотек программ к разработке ППП был вызван целым рядом причин. К их числу прежде всего относится резкое увеличение возможностей ЭВМ. Это привело к значительному усложнению системного обеспечения вычислительных машин. Произошли существенные изменения в большинстве областей применения ЭВМ.

3.2. СТРУКТУРА и ОСНОВНЫЕ КОМПОНЕНТЫ ППП

Несмотря на большое разнообразие конкретных пакетных разработок, можно выделить следующие основные компоненты ППП:

- входные языки;

- предметное обеспечение;

- системное обеспечение.

Важно отметить, что такое разбиение на составные элементы отражает в первую очередь функции, выполняемые программами ППП, а не структуру самих программ, которая зависит от индивидуаль­ных особенностей конкретного пакета. В разных пакетах указанные компоненты могут быть развиты в различной степени или вовсе отсутствовать. Однако наиболее развитые ППП, как правило, облада­ют всеми этими компонентами, каждый из которых может иметь до­вольно сложную структуру.

В многочисленных работах, посвященных пакетной проблемати­ке, из-за не устоявшейся терминологии нередко используются другие названия составных элементов ППП. Например, входной язык называют также языком заданий или языком управления. Для обозначения предметного обеспечения применяются термины "функцио­нальное наполнение", "функциональная подсистема" или "тело па­кета". Системное обеспечение часто называют системным наполнением, организующей или управляющей программой, а также процессором пакета.

Рассмотрим функции каждого из компонентов ППП.

Входные языки представляют собой средство общения пользова­теля с пакетом. Как отмечалось в п. 3.1, развитый пакет может обладать несколькими входными языками, предназначенными для выполнения различных функций и ориентированными на различные типы пользователей. Можно выделить следующие основные типы пользователей ППП:

Разработчик ППП, осуществляющий его модификацию и развитие с учетом изменения круга пользователей, класса решаемых задач (появление новых типов задач, развитие численных методов, модификация форм проведения работ и т. д.), а также состава аппаратного и программного обеспечения ЭВМ:

Ответственный за сопровождение, в функции которого входит поддержание пакета в работоспособном состоянии в условиях конкретной вычислительной системы (обеспечение сохранности программ и массивов данных, своевременное дублирование информационных файлов, выявление ошибок в программах пакета).

Администратор, отвечающий за организацию доступа пользо­вателей к пакету, содержимое базы данных, защиту информации от несанкционированного доступа;

Конечный пользователь, применяющий пакет для решения конкретных прикладных задач.

Входные языки отражают объем и качество предоставляемых па­кетом средств, а также удобство их использования. Таким образом, с точки зрения конечного пользователя именно входной язык является основным показателем возможностей ППП.

В качестве входных языков могут использоваться как универсальные, так и специализированные языки программирования. Например, в качестве входного языка разработчика ППП для написания прикладных и системных программ пакета обычно используется тот или иной универсальный язык программирования (Фортран, Паскаль). В то же время входной язык конечного пользователя в развитом пакете, как правило, является языком качественно более высокого уровня по сравнению с универсальными языками. Изобра­зительные средства такого языка учитывают особенности задач предметной области и специфику пользователей. Подобные языки называют проблемно-ориентированными, или предметно-ориентиро­ванными.

Перейдем теперь к рассмотрению других компонентов ППП, конкретная прикладная деятельность характеризуется двумя факторами:

1) классом решаемых задач и используемых для этих целей методов,

2) дисциплиной работы, т.е. совокупностью правил, соглашений и технологических приемов, принятых при разработке, отладке, эксплуатации программ.

Предметное обеспечение представляет собой компонент пакета, отражающий особенности первого из этих факторов, т. е. особенности конкретной предметной области. Предметное обеспечение включает:

- программные модули, реализующие алгоритмы (или их отдель­ные фрагменты) решения прикладных задач;

- средства сборки программ из отдельных модулей,

Определение состава библиотеки модулей и форм их взаимодействия между собой является одной из наиболее трудоемких задач при построении ППП. Ее решение предполагает проведение тщательного и квалифицированного модульного анализа используемых алгоритмов. Удачно проведенный модульный анализ в значительной степени влияет на полноту охвата предметной области, а также на возможность расширения класса решаемых задач. Таким образом, выделение модулей существенно зависит от специфики задач и используемых алгоритмов и, по сути дела, отражает принятый в пакет способ сборки программ.

Наиболее распространено в настоящее время оформление каждого модуля в виде программной единицы на том или ином языке программирования (например, в виде подпрограммы или подпрограммы–функции на языке Фортран). Такой модуль обеспечивает решение некоторой самостоятельной задачи и связан с другими модулями лишь входной и выходной информацией. Организация предметного обеспечения в виде библиотеки программ характерна для большинства существующих ППП.

Помимо рассмотренного подхода к оформлению модулей как программных единиц используются и другие способы.

Системное обеспечение представляет собой совокупность системных средств (программы, файлы, таблицы и т. д.), обеспечивающих определенную дисциплину работы пользователя при решении прикладных задач. По своей роли в составе ППП и выполняемым функциям системное обеспечение по существу является специализированной операционной системой, определяющей операционное окружение пакета. Несмотря на многообразие способов реализации системного обеспечения в рамках конкретных пакетных разработок, можно выделить его следующие основные компоненты:

- монитор, управляющий процессом решения и взаимодействием всех компонентов ППП;

- трансляторы с входных языков;

- средства работы с данными;

- средства информационного обеспечения, реализующие выдачу разнообразной справочной информации как по запросам пользователей (о структуре и возможностях ППП, о допущенных ошибках и т.д.), так и по запросам различных компонентов пакета (например, сведения о свойствах модулей предметного обеспечения, необходимые планировщику вычислений);

- различные служебные программы, в том числе реализующие взаимодействие пакета с операционной системой (работа с внешней памятью, средства ввода/вывода, драйверы специализированных, внешних устройств и др.).

В конкретном ППП, как правило, отсутствует четкое структурное разделение программ на предметное и системное обеспечение. Например, программа планирования вычислений может одновременно выполнять те или иные служебные функции {информационное обеспечение, связь с операционной системой и т. п.) и тем самым носиться как к предметному, так и к системному обеспечению. Кроме того, одни и те же программы в одном пакете могут относиться к предметному обеспечению, а в другом - к системному. Так, программы вывода графиков в рамках специализированного пакета машинной графики естественно отнести к предметному обеспечению. Однако те же программы следует считать служебными и относящимися к системному обеспечению, например, в пакете решения вычислительных задач).

3. 3 ЭТАПЫ РАЗВИТИЯ ППП

Пакетная проблематика в качестве самостоятельного научно направления сложилась в основном за последние 15-20 лет. Первые ППП представляли собой простые тематические подборки программ для решения отдельных задач в той или иной прикладной области. Современный пакет является сложной программной системой, включающей специализированные системные и языковые средства. В относительно короткой истории развития вычислительных ППП можно выделить 4 основных поколения (класса) пакетов. Каждый из этих: классов характеризуется определенными особенностями входящих состав ППП компонентов - входных языков, предметного и системного обеспечения.

В качестве входных языков ППП первого поколения использовались универсальные языки программирования (Фортран, Алгол-60 и т. п.) или языки управления заданиями соответствующих операционных систем Проблемная ориентация входных языков достигалась за счет соответствующей мнемоники в именах переменных, функций процедур, а также в текстовых константах. Составление заданий на таком языке практически не отличалось от написания программ на алгоритмическом языке.

Предметное обеспечение первых ППП, как правило, было организовано в форме библиотек программ, т.е. в виде наборов (пакетов) независимых программ на некотором базовом языке программирования (отсюда впервые возник и сам термин "пакет"). Такие ППП иногда называют пакетами библиотечного типа, или пакетами простой структуры.

В качестве системного обеспечения пакетов первого поколения обычно использовались штатные компоненты программного обеспечения ЭВМ: компиляторы с алгоритмических языков, редакторы текстов, средства организации библиотек программ, архивные системы и т. д.Эти пакеты не требовали сколько-нибудь развитой системной поддержки, и для их функционирования вполне хватало указанных системных средств общего назначения. В большинстве случаев разработчиками таких пакетов были прикладные программисты, которые пытались приспособить универсальные языки программирования к своим нуждам.

Разработка ППП второго поколения осуществлялась уже с участием системных программистов. Это привело к появлению специализированных входных языков (их называют встроенными языками) на базе универсальных языков программирования. Проблемная ориентация таких языков достигалась не только за счет использования определенной мнемоники, но также применением соответствующих языковых конструкций, которые упрощали формулировку задачи и делали ее более наглядной. Транслятор с такого языка представлял собой препроцессор (чаще всего макропроцессор) к транслятору соответствующего алгоритмического языка.

В качестве модулей в пакетах этого класса стали использоваться не только программные единицы (т.е. законченные программы на том или ином языке программирования), но и такие объекты, последовательность операторов языка программирования, совокупность данных, схема счета и др.

Существенные изменения претерпели также принципы организации системного обеспечения ППП. В достаточно развитых пакетах второго поколения уже можно выделить элементы системного обеспечения, характерные для современных пакетов: монитор, трансляторы с входных языков, специализированные банки данных, средства описания модели предметной области и планирования вычислений и др.

Третий этап развития ППП характеризуется появлением самостоятельных

входных языков, ориентированных на пользователей-непрограммистов. Особое внимание в таких ППП уделяется системным компонентам обеспечивающим простоту и удобство. Это достигается главным образом за счет такой специализации входных языков и включения в состав пакета средств автоматизированного планирования вычислений.

Наконец, четвертый этап характеризуется созданием ППП, эксплуатируемых в диалоговом режиме работы. Основным преимуществом диалогового взаимодействия с ЭВМ является возможность активной обратной связи с пользователем в процессе постановки задачи, ее решения и анализа полученных результатов. Появление и интенсивное развитие различных форм диалогового общения обусловлено прежде всего прогрессом в области технических средств обеспечения диалога. Сюда относится создание разнообразной дисплейной техники (растровые дисплеи, средства реализации графических, цветовых и звуковых возможностей, различные технические устройства для ведения диалога и т. д.), а также надежных и скоростных линий связи. Развитие аппаратного обеспечения повлекло за собой создание разнообразных программных средств поддержки диалогового режима работы (диалоговые операционные системы, диалоговые пакеты программ различного назначения и т. д.). Во многих приложениях диалог уже полностью заменил пакетную обработку, а построчный режим диалога уступает место поэкранному режиму и многооконному графическому способу общения.

Прикладная система состоит из диалогового монитора-набора универсальных программ, обеспечивающих ведение диалога и обмен данными, и базы знаний об области. Информация о структуре, целях и форма диалога задает сценарий, в соответствии с который монитор управляет ходом диалога. Носителями процедурных знаний о предметной области являются прикладные модули, реализующие функции собственной системы. Таким образом, создание прикладной системы сводится к настройке диалогового монитора на конкретный диалог, путем заполнения базы знаний. При этом программировать в традиционном смысле этого слова приходится лишь прикладные модули, знания о диалоге вводятся в систему с помощью набора соответствующих средств - редактора сценариев. Логично требовать, чтобы редактор сценариев также представлял собой диалоговую программу, отвечавшую рассмотренным выше требованиям. Благодаря готовому универсальному монитору программист может сосредоточиться на решении чисто прикладных задач, выделение же знаний о диалоге в сценарий обеспечивает в значительной степени необходимая гибкость программного продукта.

Большое внимание в настоящее время уделяется проблеме создания "интеллектуальных" ППП. Такой пакет позволяет конечному пользователю лишь сформулировать свою задачу в содержательных терминах, не указывая алгоритма ее решения. Синтез решения и сборка целевой программы производятся автоматически. При этом детали вычислений скрыты от пользователя, и компьютер становится интеллектуальным партнером человека, способным понимать его задачи. Предметное обеспечение подобного ППП представляет собой некоторую базу знаний, содержащую как, процедурные, так и описательные знания. Такой способ решения иногда называют концептуальным программированием, характерными особенностями которого является программирование в терминах предметной области использование ЭВМ уже на этапе постановки задач, автоматический синтез программ решения задачи, накопление знаний о решаемых задачах в базе знаний.

В заключение данного раздела рассмотрим еще одну современную тенденцию разработки ППП. Она заключается в применении специализированных инструментальных средств и систем, позволяющих ускорить и упростить процесс создания пакета, а также снизить стоимость разработки. При этом особое внимание уделяется созданию системных средств, позволяющих использовать в качестве предметного обеспечения ППП написанные ранее прикладные программы. Кроме того, инструментальные системы обычно реализуются таким образом, что их можно использовать в качестве базы (готовых компонентов) для системного обеспечения разрабатываемых пакетов (поэтому их иногда называют базовые инструментальными системами). Создание инструментальных средств, упрощающих разработку ППП в различных предметных областях, представляет собой одно из актуальных направлений системного программирования в пакетной проблематике.

При выборе метода реализации того или иного ППП следует учитывать особенности конкретной ситуации, в частности, имеющиеся в наличии людские и материальные ресурсы. Так пакет библиотечного типа, не являясь развитой системой с точки зрения рассмотренных требований, обладает, однако, тем преимуществом, что входной язык и системное обеспечение такого пакета могут быть достаточно легко реализованы силами прикладного программиста. Поэтому в случае, когда подобный пакет удовлетворяет конкретных пользователей, его разработка является вполне оправданной.

1. **Примеры прикладных программ**

В качестве примеров прикладных программ я хочу рассмотреть две специализированные программы, поставляемые в пакете Microsoft Office: Microsoft Word и Microsoft Excel.

Microsoft Word представляет собой популярный текстовый процессор, предназначенный для работы под управлением ОС Windows. Он представляет широкие возможности по подготовке документов. В их числе выделим следующее: развитый интерфейс, обширную и удобную в применении систему справочной помощи, широкие возможности по внедрению и связыванию графических объектов, возможности редактирования рисунков средствами самого Word, разнообразные возможности по форматированию абзацев и символов, удобство в построении и редактировании таблиц, наличие развитого формульного редактора, наличие разнообразных конвертеров для связи с другими приложениями, наличие средств контроля грамматической правильности текста, автоматизированное форматирование документов на основе стилей.

Word входит в состав Microsoft Office и обеспечивает возможность интеграции с другими компонентами названной и более ранних версий пакета. В документы Word можно легко встроить данные (таблицы, графику), сформированные в среде табличного процессора Microsoft Exсel, системы подготовки презентаций PowerPoint и СУБД Access.

Табличный процессор Excel поддерживает также общие функциональные возможности текстовых процессоров, такие как использование макросов, построение диаграмм, автозамена и проверка орфографии, использование стилей, шаблонов, автоформатирование данных, обмен данными с другими приложениями, наличие развитой справочной системы, печать с настройкой параметров и другие сервисные возможности.

Табличный процессор Excel целесообразно использовать для создания таблиц в случаях, когда предполагаются сложные расчеты, сортировка, фильтрация, статистический анализ массивов, построение на их основе диаграмм.

Опишем основные ключевые понятия, используемые при работе с табличным процессором Excel.

Рабочая книга является основным документом Excel. Она хранится в файле с произвольным именем и расширением xls. При создании или открытии рабочей книги ее содержимое представлено в отдельном окне. Каждая книга по умолчанию содержит 16 рабочих листов.

Листы предназначены для создания и хранения таблиц, диаграмм и макросов. Лист состоит из 256 столбцов и 16384 строк.

Ячейка является структурной наименьшей единицей для размещения данных внутри рабочего листа. Каждая ячейка может содержать данные в виде текста, числовых значений, формул или параметров форматирования. При вводе данных Excel автоматически распознает тип данных и определяет перечень операций, которые могут с ними производиться. По своему содержимому ячейки делятся на исходные (влияющие) и зависимые. В последних записаны формулы, которые имеют ссылки на другие ячейки таблицы. Следовательно, значения зависимых ячеек определяются содержимым других (влияющих) ячеек таблицы. Ячейка, выбранная с помощью указателя, называется активной или текущей ячейкой.

Адрес ячейки предназначен для определения местонахождения ячейки в таблице. Существует два способа записи адресов ячеек:

1. Указанием буквы столбца и номера строки таблицы, перед которыми может записываться знак $, указывающий на абсолютную адресацию. Этот способ используется по умолчанию и называется стилем А1.
2. Указанием номера строки и номера столбца, следующих после букв R и С, соответственно. Номера строк и столбцов могут заключаться в квадратные скобки, которые указывают на относительную адресацию.

Формула – это математическая запись вычислений, производимых над данными таблицы. Формула начинается со знака равенства или математического оператора и записывается в ячейку таблицы. Результатом выполнения формулы является вычисленное значение. Это значение автоматически записывается в ячейку, в которой находится формула.

Функция – это математическая запись, указывающая на выполнение определенных вычислительных операций. Функция состоит из имени и одно или нескольких аргументов, заключенных в круглые скобки.

Указатель ячейки – это рамка, с помощью которой выделяется активная ячейка таблицы. Указатель перемещается с помощью мыши или клавиш управления курсором.

Ссылка – это запись адреса ячейки в составе формулы. Ссылки могут быть абсолютные, относительные и смешанные.

Список – это специальным образом оформленная таблица, с которой можно работать как с базой данных. В такой таблице каждый столбец представляет собой поле, а каждая строка – запись файла базы данных.

Таким образом, каждый из этих пакетов решает свои функции. Развитие этих программ происходило от самого “архаического” периода до периода современности. Изменение интерфейса, простота в использовании улучшаются с каждой новой версией этих программных продуктов, Microsoft Word и Microsoft Excel являются признанными во всем мире прикладными программами, нам, обычным пользователям, остается ожидать новых версий этих программ, в надежде на их очередное усовершенствование и упрощение в пользовании.

**5. Заключение**

Современный белорусский рынок прикладного программного обеспечения является, в значительной мере, рынком пиратского ПО. Это связано с тем, что белорусский потребитель не в состоянии платить полную стоимость лицензионного ПО западных и американских разработчиков. По оценкам экспертов, до 90% продаж составляет ворованное программное обеспечение. Вместе с тем, покупая взломанное ПО, потребитель должен быть готов, что в любой момент его компьютер может «зависнуть», или возможна потеря данных.

Западные крупнейшие производители программного обеспечения решили двигаться 2-я путями: с одной стороны, они вкупе с правоохранительными органами все чаще привлекают продавцов пиратского ПО к уголовной ответственности, а с другой стараются держать цены на свои продукты минимально низкими, разрабатывая и запуская разного рода партнерские программы. Так, например, в мае 2000г. фирма Microsoft проводила опрос руководителей предприятий, использующих от 50 ПК, на предмет регистрации предприятий в московском офисе Microsoft для того, что бы в будущем они могли приобрести фирменные продукты со значительными скидками. Кроме этого, та же фирма Microsoft предлагает всем желающим «скачать» ее новейший браузер Microsoft Explorer 5.01 с узла в сети Интернет совершенно бесплатно.

Ошибки в прикладном программном обеспечении были и остаются основным путем проникновения злоумышленника как на сервера, так и на рабочие станции. Объективная причина этого – разработка подобного ПО различными группами разработчиков, которые просто не в состоянии уделить должного внимания сетевой и локальной безопасности своего продукта. И если фирмы-разработчики операционных систем тратят огромные суммы на тщательные испытания поведения их программ в нестандартных ситуациях, а также активно учитывают многолетний опыт своих же ошибок, то для небольших фирм это просто не под силу, да и крайне невыгодно экономически.

Ошибки активно ищутся группами "хакеров" практически во всем более или менее распространенном ПО, однако, наибольшую известность приобретают, конечно, исследования программ, установленных почти у каждого пользователя.

Многие атаки используют не только непосредственные ошибки в реализации ПО, но и непродуманные разработчиками аспекты использования стандартных возможностей программы. Так, пожалуй, самым ярким примером этого являются MACRO-вирусы в документах системы MicroSoft Office. Возможность исполнения макросов была встроена в эту систему из самых благих побуждений, но тот факт, что макросы могут запускаться на определенные события (например, открытие документа) и получать доступ на модификацию к другим документам, сразу же был использован создателями вирусов отнюдь не в благих целях.

В заключение я хотел бы отметить то, что разработка и защита прикладного программного обеспечения весьма трудоемкий процесс, требующий определенных навыков и знаний.

Список использованной литературы:

1. Можаров Р.В., Можарова Н.Р., Евтеев В.В., Кузьменко О.А., Шевченко М.О. Программное обеспечение персональных компьютеров//Учебное пособие для вузов. – М.: Финстатинформ, 1999.

2. Хомоненко А.Д. Основы современных компьютерных технологий//Учебное

пособие для вузов.– Ст-Петербург: Корона принт, 1998.

3. Борисов В.А. Разработка пакетов программ вычислительного типа. - М. : Изд-во

МГУ, 1990

www.citforum.ru

www.bankreferatov.ru