МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Индустриальный институт (филиал)
федерального государственного бюджетного образовательного учреждения
высшего образования «Югорский государственный университет»
(ИндИ (филиал) ФГБОУ ВО «ЮГУ»)

УТВЕРЖДАЮ
Заместитель директора по УВР
_____ Гарбар О.В.
«09» сентября 2021 г.

Методические указания

по выполнению практических работ

ПМ.04. СОПРОВОЖДЕНИЕ И ОБСЛУЖИВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ КОМПЬЮТЕРНЫХ СИСТЕМ

09.02.07 Информационные системы и программирование

Нефтеюганск
2021

РАССМОТРЕНО:
Предметной цикловой
Комиссией специальных технических
дисциплин
Протокол №1 от 09.09.2021
_____Шарипова И.А.

СОГЛАСОВАНО:
заседанием Методсовета
протокол №1 от 16.09.2021
Председатель Методсовета
_____Н.И. Савватеева

Методические указания по выполнению практических и лабораторных работ по ПМ.04. Сопровождение и обслуживание программного обеспечения компьютерных систем

Разработчик: И.В. Чупракова, преподаватель ИндИ (филиала) ФГБОУ ВО «ЮГУ».

Содержание

Пояснительная записка

Методические указания по ПМ.04. Сопровождение и обслуживание программного обеспечения компьютерных систем для специальности 09.02.07 Информационные системы и программирование разработаны на основе рабочей программы ПМ.04. Сопровождение и обслуживание программного обеспечения компьютерных систем по специальности 09.02.07 Информационные системы и программирование.

Реализация профессионального модуля предусматривает проведение лабораторных и практических работ в форме практической подготовке обучающихся.

Практическая подготовка при реализации профессионального модуля организуется путем проведения практических занятий, практикумов, лабораторных работ и иных аналогичных видов учебной деятельности, предусматривающих участие обучающихся в выполнении отдельных элементов работ, связанных с будущей профессиональной деятельностью, а также демонстрацию практических навыков, выполнение, моделирование обучающимися определенных видов работ для решения практических задач, связанных с будущей профессиональной деятельностью в условиях, приближенных к реальным производственным.

С целью овладения указанным видом профессиональной деятельности и соответствующими профессиональными компетенциями обучающийся в ходе освоения профессионального модуля **должен:**

Иметь практический опыт	В настройке отдельных компонентов программного обеспечения компьютерных систем; выполнении отдельных видов работ на этапе поддержки программного обеспечения компьютерной системы
уметь	подбирать и настраивать конфигурацию программного обеспечения компьютерных систем; использовать методы защиты программного обеспечения компьютерных систем; проводить инсталляцию программного обеспечения компьютерных систем; производить настройку отдельных компонентов программного обеспечения компьютерных систем; анализировать риски и характеристики качества программного обеспечения
знать	основные методы и средства эффективного анализа функционирования программного обеспечения; основные виды работ на этапе сопровождения программного обеспечения; основные принципы контроля конфигурации и поддержки целостности конфигурации программного обеспечения; средства защиты программного обеспечения в компьютерных системах

Результатом освоения профессионального модуля является овладение обучающимися видом профессиональной деятельности (ВПД) Разработка модулей программного обеспечения для компьютерных систем, в том числе профессиональными (ПК) и общими (ОК) компетенциями:

Код	**Наименование результата обучения**
ВД 4	Сопровождение и обслуживание программного обеспечения компьютерных систем
ПК 4.1.	Осуществлять инсталляцию, настройку и обслуживание программного обеспечения компьютерных систем.
ПК 4.2	Осуществлять измерения эксплуатационных характеристик программного обеспечения компьютерных систем
ПК 4.3	Выполнять работы по модификации отдельных компонент программного обеспечения в соответствии с потребностями заказчика

ПК 4.4	Обеспечивать защиту программного обеспечения компьютерных систем программными средствами.
ОК 1.	Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам
ОК 2.	Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности.
ОК 3	Планировать и реализовывать собственное профессиональное и личностное развитие.
ОК 4	Планировать и реализовывать собственное профессиональное и личностное развитие.
ОК 5	Планировать и реализовывать собственное профессиональное и личностное развитие.
ОК 6	Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей, применять стандарта антикоррупционного поведения.
ОК 7	Содействовать сохранению окружающей среды, ресурсосбережению, эффективно действовать в чрезвычайных ситуациях.
ОК 8	Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности
ОК 9	Использовать информационные технологии в профессиональной деятельности.
ОК 10	Пользоваться профессиональной документацией на государственном и иностранном языках

Правила выполнения практических работ

Подготовка к выполнению практических работ. Практические работы в группах проводятся в соответствии с расписанием учебных занятий. Для выполнения практических работ обучающийся должен руководствоваться следующими положениями:

– Предварительно ознакомиться с графиком выполнения практических работ;

– Внимательно ознакомиться с описанием соответствующей практической работе и установить, в чем состоит основная цель и задача этой работы;

– По лекционному курсу и соответствующим литературным источникам изучить теоретическую часть, относящуюся к данной практической работе;

– Неподготовленные к работе обучающиеся к выполнению практической работы не допускаются.

После окончания работы рабочее место должно быть приведено в порядок. В течение всего времени занятий обучающиеся обязаны находиться на своих рабочих местах. Выходить из помещения во время занятий можно только с разрешения преподавателя.

Оформление отчета по практическим работам. Составление отчета о проведенных исследованиях является важнейшим этапом выполнения практической работы. По каждой выполненной работе в рабочей тетради составляют отчет, руководствуясь следующими положениями:

– Указать название и порядковый номер лабораторной работы, а так же краткое сформулировать цель работы;

– Схемы и графики чертить с соблюдением принятых стандартных условий обозначений;

– Отчет по каждой практической работе должен содержать основные выводы. В заголовке отчета указывают номер работы и ее полное наименование. При составлении отчета

нужно кратко описать цель работы, ее содержание, указать использованные аппаратуру и оборудование.

– При выполнении практической работ необходимо строго следовать правилам техники безопасности.

Критерии оценки работ

Оценка «отлично» ставится, если обучающийся выполнил работу в полном объеме с соблюдением необходимой последовательности действий; в «Отчете к практическим работам» правильно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» ставится, если обучающийся выполнил требования к оценке "5", но допущены 2-3 недочета.

Оценка «удовлетворительно» ставится, если обучающийся выполнил работу не полностью, но объем выполненной части таков, что позволяет получить правильные результаты и выводы; в ходе проведения работы были допущены ошибки.

Оценка «неудовлетворительно» ставится, если обучающийся выполнил работу не полностью или объем выполненной части работы не позволяет сделать правильных выводов.

Перечень практических работ

Наименование разделов профессионального модуля (ПМ), междисциплинарных курсов (МДК) и тем	Содержание учебного материала, лабораторные работы и практические занятия, самостоятельная работа обучающихся, курсовая работа (проект), практическая подготовка	Объем часов Всего
ПМ.04. Сопровождение и обслуживание программного обеспечения компьютерных систем		
МДК 04.01. Внедрение и поддержка компьютерных систем		
Раздел 1. Внедрение и поддержка компьютерных систем		
Тема 4.1. Основные методы внедрения и анализа функционирования программного обеспечения	**Содержание учебного материала (практическая подготовка)**	
	Практическая работа № 4.1. «Разработка сценария внедрения программного продукта для рабочего места»	4
	Практическая работа № 4.2. «Разработка руководства оператора»	4
	Практическая работа № 4.3. «Разработка (подготовка) документации и отчетных форм для внедрения программных средств»	4
Тема 4.2. Загрузка и установка программного обеспечения	**Содержание учебного материала (практическая подготовка)**	
	Практическая работа № 4.4. «Измерение и анализ эксплуатационных характеристик качества программного обеспечения».	4
	Практическая работа № 4.5. «Выявление и документирование проблем установки программного обеспечения»	4
	Практическая работа № 4.6. «Устранение проблем совместимости программного обеспечения»	4
	Практическая работа № 4.7. «Конфигурирование программных и аппаратных средств»	4
	Практическая работа № 4.8. «Настройки системы и обновлений»	4
	Практическая работа № 4.9. «Создание образа системы. Восстановление системы»	4
	Практическая работа № 4.10. «Разработка модулей программного средства»	4
	Практическая работа № 4.11. «Настройка сетевого доступа»	4
МДК 04.02. Обеспечение качества функционирования компьютерных систем		
Раздел 2. Обеспечение качества функционирования компьютерных систем		
Тема 4.2. Основные методы обеспечения качества функционирования	**Содержание учебного материала (практическая подготовка)**	
	Практическая работа № 4.12. «Тестирование программных продуктов»	4
	Практическая работа № 4.13. «Сравнение результатов тестирования с требованиями технического задания и/или спецификацией»	4
	Практическая работа № 4.14. «Анализ рисков»	4
	Практическая работа № 4.15. «Выявление первичных и вторичных ошибок»	4
Тема 4.2. Методы и средства защиты компьютерных систем	**Содержание учебного материала (практическая подготовка)**	
	Практическая работа №4.16. «Обнаружение вируса и устранение последствий его влияния»	4
	Практическая работа № 4.17. «Установка и настройка антивируса. Настройка обновлений с помощью зеркала»	4
	Практическая работа № 4.18. «Настройка политики безопасности»	4
	Практическая работа № 4.19. «Настройка браузера»	4
	Практическая работа № 4.20. «Работа с реестром»	6
	Практическая работа № 4.21. «Работа с программой восстановления файлов и очистки дисков»	4

Практическая работа № 4.1. Разработка проекта внедрения программного продукта. Управление внедрением

Цель работы: научиться управлять внедрение программных продуктов
Теоретический материал

Методологии внедрения представляют собой глубоко проработанные, проверенные, многократно апробированные рабочие инструкции и шаблоны проектных документов. Такие стандарты обычно далеки от теоретических абстракций, ориентированы на особенности конкретных систем, содержат наилучший опыт.

Управление проектами разделяется на управление — по стоимости, срокам и содержанию.

Управления сроками проекта (time management) — это процесс, используемый для обеспечения своевременного завершения проекта. Он состоит из шести процессов:

Определение состава операций

— процесс определения конкретных плановых операций, которые необходимо выполнить для внедрения ИС.

Определение взаимосвязей операций — процесс выявления и документирования последовательности выполнения плановых операций.

Определение ресурсов операции — процесс определения необходимых для выполнения каждой плановой операции ресурсов и их количества.

Определение длительности операций — процесс определения продолжительности выполнения каждой плановой операции.

Разработка расписания — процесс составления расписания проекта с учетом последовательностей операций, их длительности, требований к ресурсам и ограничений на сроки выполнения проекта в целом.

Управление расписанием — процесс управления изменениями расписания проекта.

Проект считается успешным, если он завершен в установленные сроки, выполнен в рамках бюджета и в соответствии с ожиданиями заказчика.

Управление стоимостью проекта объединяет процессы, выполняемые в ходе планирования, разработки бюджета и контролирования затрат и обеспечивающие завершение проекта в рамках утвержденного бюджета. К процессам управления стоимостью относятся:

Стоимостная оценка

— определение примерной стоимости ресурсов, необходимых для выполнения операций проекта;

Разработка бюджета расходов

— суммирование оценок стоимости отдельных операций или пакетов работ с целью формирования базового плана по стоимости;

Управление стоимостью

— воздействие на факторы, вызывающие отклонения по стоимости, и управление изменениями бюджета проекта.

Управление рисками тесно связано с общим жизненным циклом проекта. На ранних этапах преобладают риски, связанные с бизнесом, рамками проекта, требованиями к конечному продукту и проектированием этого продукта. На стадии реализации доминируют технологические риски, далее возрастает роль рисков, связанных с поддержкой и сопровождением системы. На протяжении всего жизненного цикла проекта возникают новые риски, что требует проведения дополнительных операций анализа и планирования.

Целью управления рисками проекта является повышение вероятности реализации и значимости позитивных событий и снижение вероятности реализации событий, негативных для целей проекта.

Ход работы

Для выполнения практической работы вам понадобиться вспомнить основные управляющие внедрением функции в команде.

Задание:

- В ранее созданных вами группах распределите функции управления
- Определите ответственных и исполнителей управления
- Продумайте и опишите все процессы управления
- Создайте отчеты о проделанной работе

Контрольные вопросы

- Что такое методологии внедрения?
- Назовите известные вам методологии управления
- Что такое управление сроками проекта? Для чего оно нужно?
- Из каких процессов оно состоит?
- В чем заключается управление стоимостью?
- Из каких процессов оно состоит?
- В чем заключается управление рисками?
- Форма отчёта

Конспект, с отчётом о проделанной работе. Ответы на вопросы.

Практическая работа № 4.2. Разработка руководства оператора

Цель работы: изучение нормативно правовой документации, регламентирующей разработку документации на программные средства.

Основу отечественной нормативной базы в области документирования ПС составляет комплекс стандартов Единой системы программной документации (ЕСПД). Основная и большая часть комплекса ЕСПД была разработана в 70-е и 80-е годы. Стандарты ЕСПД в основном охватывают ту часть документации, которая создается в процессе разработки ПС, и связаны, по большей части, с документированием функциональных характеристик ПС.

Согласно ЕСПД программный документ – это документ, содержащий сведения, необходимые для разработки, изготовления, эксплуатации и сопровождения программного изделия. Номенклатуру программных документов определяет

ГОСТ 19.101-77 «ЕСПД. Виды программ и программных документов». В качестве основных видов программ стандартом определяются:

§ ***компоненты*** – программы, рассматриваемые как единое целое, выполняющие законченную функцию и применяемые самостоятельно или в составе комплекса;

§ ***комплексы*** – программы, состоящие из двух или более компонентов, выполняющие взаимосвязанные функции и применяемые самостоятельно или в составе другого комплекса.

Виды программных документов и их краткое содержание представлены в стандарте описаниями, приведенными в таблице 1.

Таблица 1. Виды программных документов

Вид документа	Содержание документа
Спецификация	Состав программы и документация на ее
Ведомость держателей подлинников	Перечень предприятий, на которых хранятся подлинники программных документов
Текст программы	Запись программы с комментариями

Описание программы	Сведения о логической структуре и функционировании программы
Программа и методика испытаний	Требования, подлежащие проверке при испытании программы, а также порядок и методы их контроля
Техническое задание	Назначение и область применения программы; технические, технико-экономические и специальные требования, предъявляемые к программе; необходимые стадии и сроки разработки; виды испытаний
Пояснительная записка	Схема алгоритма, общее описание алгоритма и (или) функционирования программы, а также обоснование принятых технических и технико-экономических решений
Эксплуатационные документы	Сведения для обеспечения функционирования и эксплуатации программы

Перечень эксплуатационных документов, рекомендуемых ЕСПД, представлен в табл. 2.
Таблица 2. Виды эксплуатационных документов

вид документа	Содержание документа
Ведомость эксплуатационных документов	Перечень эксплуатационных документов на программу
Формуляр	Основные характеристики программы, комплектность и сведения об эксплуатации программы
Описание применения	Сведения о назначении программы, области применения, применяемых методах, классе решаемых задач, ограничениях для применения, минимальной конфигурации технических средств
Руководство системного программиста	Сведения для проверки, обеспечения функционирования и настройки программы на условия конкретного применения
Руководство программиста	Сведения для эксплуатации программы
Руководство оператора (пользователя)	Сведения для обеспечения процедуры общения оператора с вычислительной системой в процессе выполнения программы
Описание языка	Описание синтаксиса и семантики языка
Руководство по техническому обслуживанию	Сведения для применения тестовых и диагностических программ при обслуживании технических средств

Допускается объединение отдельных видов эксплуатационных документов (за исключением ведомости эксплуатационных документов и формуляра), необходимость объединения указывается в техническом задании. Объединенному документу присваивают наименование и

обозначение одного из объединяемых документов. В объединенных документах должны быть приведены сведения, которые необходимо включать в каждый объединяемый документ.

ГОСТ 19.701-90 (ИСО 5807-85) "Единая система программной документации. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения". Стандарт распространяется на условные обозначения (символы) в схемах алгоритмов, программ, данных и систем и устанавливает правила выполнения схем, используемых для отображения различных видов задач обработки данных и средств их решения.

В РФ действует ряд стандартов в части документирования ПС, разработанных на основе прямого применения международных стандартов ИСО.

ГОСТ Р ИСО/МЭК 9294-93 «Информационнаятехнология. Руководство по управлению документированием программного обеспечения». Стандарт устанавливает рекомендации по эффективному управлению документированием ПС для руководителей, отвечающих за их создание. Целью стандарта является оказание помощи в определении стратегии документирования ПС; выборе стандартов по документированию; выборе процедур документирования; определении необходимых ресурсов; составлении планов документирования.

ГОСТ Р ИСО 9127-94 «Системы обработки информации. Документация пользователя и информация на упаковке для потребительских программных пакетов». В контексте настоящего стандарта под потребительским программным пакетом (ПП) понимается «программная продукция, спроектированная и продаваемая для выполнения определенных функций; программа и соответствующая ей документация, упакованные для продажи как единое целое». Под документацией пользователя понимается документация, которая обеспечивает конечного пользователя информацией по установке и эксплуатации ПП. Под информацией на упаковке понимают информацию, воспроизводимую на внешней упаковке ПП. Ее целью является предоставление потенциальным покупателям первичных сведений о ПП.

Содержание документа "Руководство пользователя" Документ **"Руководство пользователя",** разрабатывается на основании методических указаний РД 50-34.698-90. Данный документ формируется IT-специалистом, или функциональным специалистом, или техническим писателем в ходе разработки рабочей документации на систему и её части на стадии «Рабочая документация».

Состав руководства оператора в соответствии со стандартом

1. Введение.
2. Назначение и условия применения.
3. Подготовка к работе.
4. Описание операций.
5. Аварийные ситуации.
6. Рекомендации по освоению.

Введение

В разделе "Введение" указывают:

область применения;

краткое описание возможностей;

уровень подготовки пользователя;

перечень эксплуатационной документации, с которой необходимо ознакомиться пользователю.

1.1. Область применения

Требования настоящего документа применяются при:

предварительных комплексных испытаниях;

опытной эксплуатации;

приемочных испытаниях;

промышленной эксплуатации.

1.2. Краткое описание возможностей

Например:

Информационно-аналитическая система Корпоративное Хранилище Данных (ИАС КХД) предназначена для оптимизации технологии принятия тактических и стратегических управленческих решений конечными бизнес-пользователями на основе информации о всех аспектах финансово-хозяйственной деятельности Компании.

ИАС КХД предоставляет возможность работы с регламентированной и нерегламентированной отчетностью.

1.3. Уровень подготовки оператора Например:

Оператор ИАС КХД должен иметь опыт работы с ОС MS Windows (95/98/NT/2000/XP), навык работы с ПО Internet Explorer, Oracle Discoverer, а также обладать следующими знаниями:

- *знать соответствующую предметную область;*
- *знать основы многомерного анализа;*
- *понимать многомерную модель соответствующей предметной области;*
- *знать и иметь навыки работы с аналитическими приложениями.*

Квалификация Оператор должна позволять:

- *формировать отчеты в Oracle Discoverer Plus;*
- *осуществлять анализ данных.*

1.4. Перечень эксплуатационной документации, с которой необходимо ознакомиться оператору

Например:

информационно-аналитическая система «Корпоративное хранилище данных».

ПАСПОРТ;

Информационно-аналитическая система «Корпоративное хранилище данных». ОБЩЕЕ ОПИСАНИЕ СИСТЕМЫ.

2. Назначение и условия применения

В разделе "Назначение и условия применения" указывают: - виды деятельности, функции, для автоматизации которых предназначено данное средство автоматизации;

условия, при соблюдении (выполнении, наступлении) которых обеспечивается применение средства автоматизации в соответствии с назначением (например, вид ЭВМ и конфигурация технических средств, операционная среда и общесистемные программные средства, входная информация, носители данных, база данных, требования к подготовке специалистов и т. п.).

Например:

Oracle Discoverer Plus в составе ИАС КХД предназначен для автоматизации подготовки, настройки отчетных форм по показателям деятельности, а также для углубленного исследования данных на основе корпоративной информации хранилища данных.

Работа с Oracle Discoverer Plus в составе ИАС КХД возможна всегда, когда есть необходимость в получении информации для анализа, контроля, мониторинга и принятия решений на ее основе.

Работа с Oracle Discoverer Plus в составе ИАС КХД доступна всем пользователям с установленными правами доступа.

3. Подготовка к работе

В разделе "Подготовка к работе" указывают:

- *состав и содержание дистрибутивного носителя данных;*
- *порядок загрузки данных и программ;*
- *порядок* проверки работоспособности.

3.1. Состав и содержание дистрибутивного носителя данных

Например:

Для работы с ИАС КХД необходимо следующее программное обеспечение:

Internet Explorer (входит в состав операционной системы Windows);

Oracle JInitiator устанавливается автоматически при первом обращении пользователя к ИАС КХД.

3.2. Порядок загрузки данных и программ Например:

Перед началом работы с ИАС КХД на рабочем месте оператора необходимо выполнить следующие действия:

Необходимо зайти на сайт ИАС КХД ias-dwh.ru.

Во время загрузки в появившемся окне "Предупреждение

о *безопасности", которое будет содержать следующее: 'Хотите установить и выполнить "Oracle JInitiator" ...' Нажимаем на кнопку "Да".*

После чего запуститься установка Oracle JInitiator на Ваш компьютер. Выбираем кнопку Next и затем OK.

3.3. Порядок проверки работоспособности

Например:

Для проверки доступности ИАС КХД с рабочего места оператора необходимо выполнить следующие действия:

Открыть Internet Explorer, для этого необходимо кликнуть по ярлыку «Internet Explorer» на рабочем столе или вызвать из меню «Пуск».

Ввести в адресную строку Internet Explorer адрес: ias-dwh.ru и нажать «Переход».

В *форме аутентификации ввести пользовательский логин и пароль. Нажать кнопку «Далее».*

Убедиться, что в окне открылось приложение Oracle Discoverer Plus.

В *случае если приложение Oracle Discoverer Plus не запускается, то следует обратиться в службу поддержки.*

4. Описание операций

В разделе "Описание операций" указывают:

- *описание всех выполняемых функций, задач, комплексов задач, процедур;*
- *описание операций технологического процесса обработки данных, необходимых для выполнения* функций, комплексов задач (задач), процедур.

Для каждой операции обработки данных указывают:

- *наименование;*
- *условия, при соблюдении которых возможно выполнение операции;*
- *подготовительные действия;*
- *основные действия в требуемой последовательности;*
- *заключительные действия;*
- *ресурсы*, расходуемые на операцию.

В описании действий допускаются ссылки на файлы подсказок, размещенные на магнитных носителях.

4.1. Выполняемые функции и задачи

Например:

Oracle Discoverer Plus в составе ИАС КХД выполняет функции и задачи

4.2. Описание операций технологического процесса обработки данных, необходимых для выполнения задач

Например:

Задача: «Визуализация отчетности»

Операция 1: *Регистрация на портале ИАС КХД*

Условия*, при соблюдении которых возможно выполнение операции:*

- *Компьютер пользователя подключен к корпоративной сети.*
- *Портал ИАС КХД доступен.*
- *ИАС КХД функционирует в штатном режиме.*

***Подготовительные действия**:*

· На компьютере оператора необходимо выполнить дополнительные настройки, приведенные в п. 3.2 настоящего документа.

***Основные действия** в требуемой последовательности:*

На иконке «ИАС КХД» рабочего стола произвести двойной щелчок левой кнопкой мышки.

В открывшемся окне в поле «Логин» ввести имя пользователя, в поле «Пароль» ввести пароль пользователя. Нажать кнопку «Далее».

Заключительные действия:

Не требуются.

***Ресурсы, расходуемые на операцию**:*

15-30 секунд.

5. Аварийные ситуации

В разделе "Аварийные ситуации" указывают:

1. действия в случае несоблюдения условий выполнения технологического процесса, в том числе при длительных отказах технических средств;

2. действия по восстановлению программ и/или данных при отказе магнитных носителей или обнаружении ошибок в данных;

3. действия в случаях обнаружении несанкционированного вмешательства в данные;

4. действия в других аварийных ситуациях. Например:

В случае возникновения ошибок при работе ИАС КХД, не описанных ниже в данном разделе, необходимо обращаться к сотруднику подразделения технической поддержки ДИТ (HelpDesk) либо к ответственному Администратору ИАС КХД.

6. Рекомендации по освоению

В разделе "Рекомендации по освоению" указывают рекомендации по освоению и эксплуатации, включая описание контрольного примера, правила его запуска и выполнения.

Например:

Рекомендуемая литература:

Oracle® Business Intelligence Discoverer Viewer User's Guide Oracle® Business Intelligence Discoverer Plus User's Guide

Рекомендуемые курсы обучения:

Discoverer 10g: Создание запросов и отчетов

Ход работы

1. Подготовить документ (*.doc), содержащий структуру основных разделов руководства оператора стандартного форматирования: шрифт TimesNewRoman, 12 пт, поля, межстрочный интервал - стандартные, как в техническом задании, имя файла - <ФИО студента. Руководство пользователя>.

2. На основании технического задания на разработку (практическая работа МДК 03.01), заполнить разделы руководства оператора "Введение", "Назначение и условия применения", "Подготовка к работе".

3. Сохранить документ с именем (Фамилия, инициалы студента. Наименование работы).

4. Прикрепить файл руководства оператора в разделе Руководство оператора (практическая работа 2) учебного сервера stud.scc

5. Используя почтовый-клиент Mozilla Thunderbird отослать письмо-отчет преподавателю с указанием гиперссылки на стартовую страницу по адресу vlr@prep.scc или 90@192.168.5.90

6. Ответить на контрольные вопросы (или выполнить тест на ПК).

1. Перечислить состав разделов руководства пользователя.
2. Пояснить состав раздела «Введение».
3. Пояснить состав раздела «Назначение и условия применения2 применения».
4. Пояснить состав раздела «Подготовка к работе»
5. Пояснить состав раздела «Описание операций»
6. Пояснить состав раздела «Аварийные ситуации»
7. Пояснить состав подраздела «Рекомендации по освоению»

Практическая работа 4.3. Разработка (подготовка) документации и отчетных форм для внедрения программных средств

Цель работы: научиться составлять техническую документацию программного продукта.

Теоретический материал

Программная документация, включает:

Техническое задание (назначение, область применения программы, требования, предъявляемые к программе);

Текст программы (запись программы с необходимыми комментариями);

Описание программы (сведения о логической структуре и функционировании программы);

Пояснительная записка (схема алгоритма, общее описание алгоритма и/или функционирования программы, обоснование принятых решений);

Эксплуатационные документы.

К эксплуатационным документам относят:

- описание применения (сведения о назначении программы, области применения, применяемых методах, классе решаемых задач, ограничениях для применения, минимальной конфигурации технических средств);
- руководство системного программиста (сведения для проверки, обеспечения функционирования и настройки программы на условия конкретного применения);
- руководство программиста (сведения для эксплуатации программы);
- руководство оператора (сведения для обеспечения общения оператора с вычислительной системой в процессе выполнения программы);
- описание языка (описание синтаксиса и семантики языка);
- руководство по техническому обслуживанию (сведения для применения тестовых и диагностических программ при обслуживании технических средств)

Основная часть программной документации составляется на стадии рабочего проекта. Необходимость того или иного документа определяется на этапе составления технического задания. Допускается объединять отдельные виды документов.

Эксплуатационный документ "Описание языка" включается в программную документацию, если разработанный программный продукт реализует некий язык программирования, управления заданиями, организации вычислительного процесса и т. п.

Эксплуатационный документ "Руководство по техническому обслуживанию" включается в программную документацию, если разработанный программный продукт требует использования тестовых или диагностических программ.

Описание применения

Документ "Описание применения" относится к эксплуатационным документам и состоит из следующих разделов:

- назначение программы (возможности, основные характеристики, ограничения области применения);
- условия применения (требования к техническим и программным средствам, общие характеристики входной и выходной информации, а также требования и условия организационного, технического и технологического характера);
- описание задачи (указываются определения задачи и методы её решения);
- входные и выходные данные.

Руководство программиста

Документ "Руководство программиста" относится к эксплуатационным документам и включается в программную документацию, если разработанный программный продукт требует обслуживания программистом. Документ состоит из следующих разделов:

- назначение и условия применения программы (назначение и функции программы, сведения о технических и программных средствах, обеспечивающих выполнение данной программы);
- характеристики программы (временные характеристики, режимы работы, средства контроля правильности выполнения и т. п.);
- обращение к программе (способы передачи управления и параметров данных);
- входные и выходные данные (формат и кодирование);
- сообщения (тексты сообщений, выдаваемых программисту или оператору в ходе выполнения программы и описание действий, которые необходимо предпринять по этим сообщениям).

Руководство оператора

Документ "Руководство оператора" относится к эксплуатационным документам и состоит из следующих разделов:

- назначение программы (информация, достаточная для понимания функций программы и её эксплуатации);
- условия выполнения программы (минимальный и/или максимальный набор технических и программных средств и т. п.);
- выполнение программы (последовательность действий оператора, обеспечивающих загрузку, запуск, выполнение и завершение программы; описываются функции, форматы и возможные варианты команд, с помощью которых оператор осуществляет загрузку и управляет выполнением программы, а также ответы программы на эти команды);
- сообщения оператору (тексты сообщений, выдаваемых оператору в ходе выполнения программы и описание действий, которые необходимо предпринять по этим сообщениям).

Ход работы

Для готового программного модуля, создать руководство пользователя программного продукта.

Документация должна содержать необходимые сведения по установке, обеспечению надёжной работы продукта, справочное пособие для пользователя, демонстрационные версии, примеры документов, создаваемых при помощи данного программного продукта, обучающие программы.

а). Оформлять документацию на программные средства.

Текстовый документ, выполненный согласно общим положениям о стандартах документирования программных средств

б) . Использовать инструментальные средства для автоматизации оформления документации.

Использование текстового редактора.

в). Методы и средства разработки технической документации
при работе использовать ссылки на следующие документы:
Техническое задание
Программа и методика испытаний
Пояснительная записка
Ведомость эксплуатационных документов
Руководство программиста

Практическая работа № 4.4. Измерение и анализ эксплуатационных характеристик качества программного обеспечения

Цель работы: изучить принципы измерения и анализа эксплуатационных характеристик ПО

Теоретические сведения

Качество программного обеспечения

Как проверить, что требования определены достаточно полно и описывают все, что ожидается от будущей программной системы? Это можно сделать, проследив, все ли необходимые аспекты *качества ПО* отражены в них. Именно понятие *качественного ПО* соответствует представлению о том, что *программа* достаточно успешно справляется со всеми возложенными на нее задачами и не приносит проблем ни конечным пользователям, ни их начальству, ни службе поддержки, ни специалистам *по* продажам. Да и самим разработчикам создание качественной программы приносит гораздо больше удовольствия.

Если попросить группу людей высказать свое мнение *по* поводу того, что такое *качественное ПО*, можно получить следующие варианты ответов:

- Его легко использовать.
- Оно демонстрирует хорошую *производительность*.
- В нем нет ошибок.
- Оно не портит пользовательские данные при сбоях.
- Его можно использовать на разных платформах.
- Оно может работать 24 часа в сутки и 7 дней в неделю.
- В него легко добавлять новые возможности.
- Оно удовлетворяет потребности пользователей.
- Оно хорошо документировано.

Все это действительно имеет непосредственное *отношение* к *качеству ПО*. Но эти ответы выделяют характеристики, важные для конкретного пользователя, разработчика или группы таких лиц. Для того чтобы удовлетворить потребности всех сторон (конечных пользователей, заказчиков, разработчиков, администраторов систем, в которых оно будет работать, регулирующих организаций и пр.), для достижения прочного положения разрабатываемого *ПО* на рынке и повышения потенциала его развития необходим учет всей совокупности характеристик *ПО*, важных для всех заинтересованных лиц.

Приведенные выше ответы показывают, что *качество ПО* может быть описано большим набором разнородных характеристик. Такой подход к описанию сложных понятий называется **холистическим** (от греческого слова ????, целое). Он не дает единой концептуальной основы для рассмотрения затрагиваемых вопросов, какую дает *целостная система* представлений (например, Ньтоновская механика в физике или классическая теория вычислимости на основе машин Тьюринга), но позволяет, *по* крайней мере, не упустить ничего существенного.

Качество программного обеспечения определяется в стандарте *ISO* 9126 [1] как вся совокупность его характеристик, относящихся к возможности удовлетворять высказанные или подразумеваемые потребности всех заинтересованных лиц.

Тот же стандарт *ISO* 9126 [1,2,3,4] дает следующее *представление* качества.

Различаются понятия **внутреннего качества**, связанного с характеристиками *ПО* самого *по* себе, без учета его поведения; **внешнего качества**, характеризующего *ПО* с точки зрения его поведения; и *качества ПО* **при использовании** в различных контекстах — того качества, которое ощущается пользователями при конкретных сценариях работы *ПО*. Для всех этих аспектов качества введены метрики, позволяющие оценить их. Кроме того, для создания добротного *ПО* существенно качество технологических процессов его разработки.

Общие принципы обеспечения качества процессов производства во всех отраслях экономики регулируются набором стандартов *ISO 9000*. Наиболее важные для разработки *ПО* стандарты в его составе следующие:

- ISO 9000:2000 Quality management systems — Fundamentals and vocabulary [5].
- Системы управления качеством — Основы и словарь. (Аналог — ГОСТ Р-2001).
- ISO 9001:2000 Quality management systems — Requirements. Models for quality assurance in design, development, production, installation, and servicing [6].

Системы управления качеством — Требования. Модели для обеспечения качества при проектировании, разработке, коммерциализации, установке и обслуживании.

Определяет общие правила обеспечения качества результатов во всех процессах жизненного цикла. (Аналог — ГОСТ Р-2001).

- Этот стандарт выделяет следующие процессы:
  - Управление качеством.
  - Управление ресурсами.
  - Развитие системы управления.
  - Исследования рынка.
  - Проектирование продуктов.
  - Приобретения.
  - Производство.
  - Оказание услуг.
  - Защита продуктов.
  - Оценка потребностей заказчиков.
  - Поддержка коммуникаций с заказчиками.
  - Поддержка внутренних коммуникаций.
  - Управление документацией.
  - Ведение записей о деятельности.
  - Планирование.
  - Обучение персонала.
  - Внутренние аудиты.
  - Оценки управления.
  - Мониторинг и измерения.
  - Управление несоответствиями.
  - Постоянное совершенствование.
  - Управление и развитие системы в целом.
- Для каждого процесса требуется иметь планы развития процесса, состоящие как минимум из следующих разделов:
  - Проектирование процесса.
  - Документирование процесса.
  - Реализация процесса.
  - Поддержка процесса.
  - Мониторинг процесса.
  - Управление процессом.
  - Усовершенствование процесса.
- Помимо поддержки и развития системы процессов, нацеленных на удовлетворение нужд заказчиков и пользователей, *ISO 9001* требует:

- Определить, документировать и развивать собственную систему качества на основе измеримых показателей.
- Использовать эту систему качества как средство управления процессами, нацеливая их на большее удовлетворение нужд заказчиков, планируя и постоянно отслеживая качество результатов всех видов деятельности, в том числе и самого управления.
- Обеспечить использование качественных ресурсов, качественного (компетентного, профессионального) персонала, качественной инфраструктуры и качественного окружения.
- Постоянно контролировать соблюдение требований к качеству на практике, во всех процессах проектирования, производства, предоставления услуг и при приобретениях.
- Предусмотреть процесс устранения дефектов, определить и контролировать качество результатов этого процесса.

Ранее использовавшиеся стандарты *ISO 9002*:1994 Quality systems — Model for *quality assurance* in production, installation and servicing и *ISO 9003*:1994 Quality systems — Model for *quality assurance* in final *inspection* and test в 2000 году были заменены соответствующими им частями *ISO 9001*.

- ISO 9004:2000 Quality management systems — Guidelines for performance improvements [7].
- Системы управления качеством. Руководство по улучшению деятельности. (Аналог — ГОСТ Р-2001).
- ISO/IEC 90003:2004 Software engineering — Guidelines for the application of ISO 9001:2000 to computer software .

Руководящие положения по применению стандарта *ISO 9001* при разработке, поставке и обслуживании программного обеспечения

Этот стандарт конкретизирует положения *ISO 9001* для разработки программных систем, с упором на обеспечение качества при процессе проектирования. Он также определяет некоторый набор техник и процедур, которые рекомендуется применять для контроля и обеспечения качества разрабатываемых программ.

Стандарт *ISO* 9126 [1,2,3,4] предлагает использовать для описания внутреннего и внешнего *качества ПО* многоуровневую модель. На верхнем уровне выделено 6 основных характеристик *качества ПО*. Каждая характеристика описывается при помощи нескольких входящих в нее **атрибутов**.

Ниже приведены определения характеристик и атрибутов *по* стандарту *ISO* 9126:2001:

- ***Функциональность (functionality)***

Способность ПО в определенных условиях решать задачи, нужные пользователям. Определяет, что именно делает ПО, какие задачи оно решает.

- **Функциональная пригодность (suitability)**.

Способность решать нужный набор задач.

- **Точность (accuracy)**.

Способность выдавать нужные результаты.

- **Способность к взаимодействию (interoperability)**.

Способность взаимодействовать с нужным набором других систем.

- **Соответствие стандартам и правилам (compliance)**.

Соответствие ПО имеющимся индустриальным стандартам, нормативным и законодательным актам, другим регулирующим нормам.

- **Защищенность (security)**.

Способность предотвращать неавторизированный, т.е. без указания лица, пытающегося его осуществить, и неразрешенный доступ к данным и программам.

- ***Надежность (reliability)***.

Способность ПО поддерживать определенную работоспособность в заданных условиях.

- **Зрелость, завершенность (maturity)**.

Величина, обратная частоте отказов ПО. Обычно измеряется средним временем работы без сбоев и величиной, обратной вероятности возникновения отказа за данный период времени.

- **Устойчивость к отказам (fault tolerance)**.

Способность поддерживать заданный уровень работоспособности при отказах и нарушениях правил взаимодействия с окружением.

- **Способность к восстановлению (recoverability)**.

Способность восстанавливать определенный уровень работоспособности и целостность данных после отказа, необходимые для этого время и ресурсы.

- **Соответствие стандартам** *надежности* **(reliability compliance)**.

Этот атрибут добавлен в 2001 году.

- ***Удобство использования*** **(usability) или практичность**.

Способность ПО быть удобным в обучении и использовании, а также привлекательным для пользователей.

- **Понятность (understandability)**.

Показатель, обратный к усилиям, которые затрачиваются пользователями на восприятие основных понятий ПО и осознание их применимости для решения своих задач.

- **Удобство обучения (learnability)**.

Показатель, обратный усилиям, затрачиваемым пользователями на обучение работе с ПО.

- **Удобство работы (operability)**.

Показатель, обратный усилиям, предпринимаемым пользователями для решения своих задач с помощью ПО.

- **Привлекательность (attractiveness)**.

Способность ПО быть привлекательным для пользователей. Этот атрибут добавлен в 2001 году.

- **Соответствие стандартам** *удобства использования* **(usability compliance)**.

Этот атрибут добавлен в 2001 году.

- ***Производительность*** **(efficiency) или эффективность**.

Способность ПО при заданных условиях обеспечивать необходимую работоспособность по отношению к выделяемым для этого ресурсам. Можно определить ее и как отношение получаемых с помощью ПО результатов к затрачиваемым на это ресурсам всех типов.

- **Временная эффективность (time behaviour)**.

Способность ПО выдавать ожидаемые результаты, а также обеспечивать передачу необходимого объема данных за отведенное время.

- **Эффективность использования ресурсов (resource utilisation)**.

Способность решать нужные задачи с использованием определенных объемов ресурсов определенных видов. Имеются в виду такие ресурсы, как оперативная и долговременная память, сетевые соединения, устройства ввода и вывода и пр.

- **Соответствие стандартам** *производительности* **(efficiency compliance)**.

Этот атрибут добавлен в 2001 году.

- ***Удобство сопровождения (maintainability)***.

Удобство проведения всех видов деятельности, связанных с сопровождение программ.

- **Анализируемость (analyzability) или удобство проведения анализа**.

Удобство проведения анализа ошибок, дефектов и недостатков, а также удобство анализа необходимости изменений и их возможных последствий.

- **Удобство внесения изменений (changeability)**.

Показатель, обратный трудозатратам на выполнение необходимых изменений.

- **Стабильность (stability)**.

Показатель, обратный риску возникновения неожиданных эффектов при внесении необходимых изменений.

- **Удобство проверки (testability)**.

Показатель, обратный трудозатратам на проведение *тестирования* и других видов проверки того, что внесенные изменения привели к нужным результатам.

- **Соответствие стандартам** *удобства сопровождения* **(maintainability compliance)**.

Этот атрибут добавлен в 2001 году.

- ***Переносимость (portability)***.

Способность ПО сохранять работоспособность при переносе из одного окружения в другое, включая организационные, аппаратные и программные аспекты окружения.

Иногда эта характеристика называется в русскоязычной литературе мобильностью. Однако термин "мобильность" стоит зарезервировать для перевода "mobility" — способности ПО и компьютерной системы в целом сохранять работоспособность при ее физическом перемещении в пространстве.

- **Адаптируемость (adaptability)**.

Способность ПО приспосабливаться различным окружениям без проведения для этого действий, помимо заранее предусмотренных.

- **Удобство установки (installability)**.

Способность ПО быть установленным или развернутым в определенном окружении.

- **Способность к сосуществованию (coexistence)**.

Способность ПО сосуществовать с другими программами в общем окружении, деля с ними ресурсы.

- **Удобство замены (replaceability) другого ПО данным**.

Возможность применения данного ПО вместо других программных систем для решения тех же задач в определенном окружении.

- **Соответствие стандартам** *переносимости* **(portability compliance)**.

Этот атрибут добавлен в 2001 году.

Перечисленные атрибуты относятся к внутреннему и внешнему *качеству ПО* согласно ISO 9126. Для описания *качества ПО* при использовании стандарт ISO 9126-4 [4] предлагает другой, более узкий набор характеристик.

- **Эффективность (effectiveness)**.

Способность ПО предоставлять пользователям возможность решать их задачи с необходимой точностью при использовании в заданном контексте.

- **Продуктивность (productivity)**.

Способность ПО предоставлять пользователям определенные результаты в рамках ожидаемых затрат ресурсов.

- **Безопасность (safety)**.

Способность ПО обеспечивать необходимо низкий уровень риска нанесения ущерба жизни и здоровью людей, бизнесу, собственности или окружающей среде.

- **Удовлетворение пользователей (satisfaction)**.

Способность ПО приносить удовлетворение пользователям при использовании в заданном контексте.

Помимо перечисленных характеристик и атрибутов качества, стандарт *ISO* 9126:2001 определяет наборы метрик для оценки каждого атрибута. Приведем следующие примеры таких метрик.

- **Полнота реализации функций** — процент реализованных функций по отношению к перечисленным в требованиях. Используется для измерения функциональной пригодности.
- **Корректность реализации функций** — правильность их реализации по отношению к требованиям. Используется для измерения функциональной пригодности.

- **Отношение числа обнаруженных дефектов к прогнозируемому**. Используется для определения зрелости.
- **Отношение числа проведенных тестов к общему их числу**. Используется для определения зрелости.
- **Отношение числа доступных проектных документов к указанному в их списке**. Используется для измерения удобства проведения анализа.
- **Наглядность и полнота документации**. Используется для оценки понятности.

Перечисленные характеристики и атрибуты *качества ПО* позволяют систематически описывать требования к нему, определяя, какие свойства *ПО по* данной характеристике хотят видеть заинтересованные стороны. Таким образом, требования должны определять следующее.

- Что ПО должно делать, например:
  - позволять клиенту оформить заказы и обеспечить их доставку;
  - обеспечивать контроль качества строительства и отслеживать проблемные места;
  - поддерживать нужные характеристики автоматизированного процесса производства, предотвращая аварии и оптимальным образом используя имеющиеся ресурсы.
- Насколько оно должно быть надежно, например:
  - работать 7 дней в неделю и 24 часа в сутки;
  - допускается неработоспособность в течение не более 3 часов в год;
  - никакие введенные пользователями данные при отказе не должны теряться.
- Насколько им должно быть удобно пользоваться, например:
  - покупатель должен, зная название товара и имея средние навыки работы в Интернет, находить нужный ему товар за не более чем 2 минуты;
  - инженер по специальности "строительство мостов" должен в течение одного дня уметь разобраться в 80% функций системы.
- Насколько оно должно быть эффективно, например:
  - поддерживать обслуживание до 10000 запросов в секунду;
  - время отклика на запрос при максимальной загрузке не должно превышать 3 с;
  - время реакции на изменение параметров процесса производства не должно превышать 0.1 с;
  - на обработку одного запроса не должно тратиться более 1 MB оперативной памяти.
- Насколько удобно должно быть его сопровождение, например:
  - добавление в систему нового вида запросов не должно требовать более 3 человеко-дней;
  - добавление поддержки нового этапа процесса производства не должно стоить более $20000.
- Насколько оно должно быть переносимо, например:
  - ПО должно работать на операционных системах Linux, Windows XP и MacOS X;
  - ПО должно работать с документами в форматах MS Word 97 и HTML;
  - ПО должно сохранять файлы отчетов в форматах MS Word 2000, MS Excel 2000, HTML, RTF и в виде обычного текста;
  - ПО должно сопрягаться с существующей системой записи данных о заказах.

Приведенные атрибуты качества закреплены в стандартах, но это не значит, что они вполне исчерпывают понятие *качества ПО*. Так, в стандарте *ISO* 9126 отсутствуют характеристики, связанные с **мобильностью ПО (mobility)**, т.е. способностью программы работать при физических перемещениях машины, на которой она работает. Вместо *надежности* многие исследователи предпочитают рассматривать более общее понятие **добротности (dependability)**, описывающее способность *ПО* поддерживать определенные *показатели качества по* основным характеристикам (*функциональности*, *производительности*, *удобству использования*) с заданными вероятностями выхода за их рамки и определенным максимальным ущербом от воз-

можных нарушений. Кроме того, активно исследуются понятия *удобства использования*, безопасности и защищенности *ПО,* — они кажутся большинству специалистов гораздо более сложными, чем это описывается данным стандартом.

ХОД РАБОТЫ:

Основные составляющие тестирования перечислены. Они использованы для оценки функционального качества информационной системы.

1. Функциональные возможности удовлетворяют сформулированные потребности заказчиков и пользователей при применении информационной системы, но они малы. Таким образом, функциональные возможности средние.
2. Информационная система имеет все необходимые функции, исходя из технического задания. Таким образом, функциональная пригодность высокая.
3. Программное средство обеспечивает правильные или приемлемые результаты и внешние эффекты. Таким образом, правильность высокая.
4. Так как система пока не сетевая и ее используют только сотрудники магазина, она лишена защиты паролем. Таким образом, защищенность низкая.
5. В программном продукте используется мало функций, что отрицательно сказывается на надежности. Таким образом, надёжность низкая.
6. Так как система разработана в простом средстве разработки Delphi, можно без проблем модифицировать ее или добавить какие-нибудь новые функции. Таким образом, сопровождаемость высокая.
7. Система имеет очень простой и понятный интерфейс. Из-за этого она будет привлекательна для квалифицированных пользователей при применении. Таким образом, практичность высокая.
8. Информационная система использует мало вычислительных ресурсов при выполнении своих задач и функций. Таким образом, эффективность высокая.
9. База данных находится в папке с информационной системой и связана с ней, поэтому появилась возможность беспроблемного переноса из одного компьютера в другой. При этом программа не требует никаких записей в реестре операционной системы. Таким образом, мобильность высокая.

Тестирование программного средства выполнено. Теперь можно соотнести все характеристики и оценки на них в таблицу

Задание 1. Заполните таблицу на основе тестирования ПО

Таблица 3.7. Тестирование программного средства.

Характеристика	Оценка	
Функциональные возможности		
Функциональная пригодность		
Правильность		
Защищенность		
Надежность		
Сопровождаемость		
Практичность		

Эффективность		
Мобильность		

Задание 2. Опишите выявленные недостатки разработанной программы
Задание 3. Опишите перспективы развития

Практическая работа № 4.5. Выявление и документирование проблем установки программного обеспечения

Цель работы: научиться устанавливать ПО и выявлять проблемы установки

Ход работы

От версии к версии Microsoft старается улучшить стабильность и надежность операционной системы Windows, правда иногда что-то начинает идти не так и возникают проблемы. Начиная с самых ранних версий ОС Windows NT, в них существовали инструменты поиска и устранения неисправностей, но они были запрятаны глубоко в недра системы и были довольно сложны в применении, практически недоступны начинающему пользователю. Все изменилось в Windows 7.

В этой операционной системе появился новый компонент для устранения проблем - этот компонент Устранение неполадок(Windows Troubleshooting Platform), который является расширяемой инфраструктурой для автоматизированной диагностики проблем аппаратных средств и программного обеспечения и попытки автоматически устранять некоторые распространенные проблемы, такие как проблемы, возникающие при работе с сетью, аппаратным обеспечением и устройствами, связанные с использованием Интернета, а также проблемы совместимости программ.

Компонент обзавелся графическим интерфейсом и теперь пользователю, выполняющему поиск неисправностей поможет Мастер, который попытается идентифицировать источник проблемы, предоставит инструкции для решения проблемы или решит ее автоматически. Несмотря на то, что компонент Устранения неполадок не рассчитаны на решение всех возможных проблем, рекомендуется использовать его в качестве первого этапа работ по устранению неполадок, так как это может сэкономить время и избавить пользователя от лишних действий.

1. Открыть компонент устранение неполадок можно из нескольких мест:

- Панель управления -Устранение неполадок
- Панель управления -Восстановление
- Устранение неполадок- Центр поддержки -Устранение неполадок

Кроме того, запустить компонент можно и из некоторых работающих приложений.

Например, если IE не может открыть веб сайт, щелкните кнопку Диагностика проблем подключения. Запустится мастер Диагностики сетей, который входит в пакет поиска неисправностей компонента Устранение неполадок.

2. Для знакомства запустим компонент Устранение неполадок из Панели управления: Щелкните Пуск -Панель управления -Крупные значки -Устранение неполадок. Откроется окно компонента Устранение неполадок, если пользователь открыл это окно впервые, то будет предложено получить доступ к Windows Online Troubleshooting Service (WOTS) -это бесплатный онлайн сервис, позволяющий Windows загружать новые или обновленные пакеты поиска неисправностей. Нажмите кнопку Да, если хотите связаться с WOTS или Нет, если хотите пользоваться только встроенными средствами поиска неисправностей.
3. Для того, чтобы получать из интернета сведения или новые средства устранения неполадок внизу окна должен быть установлен флажок Получить самые последние средства устранения неполадок через интернет-службу устранения неполадок Windows.
4. Кроме того проверьте параметры настройки компонента Устранения неполадок. Для этого нажмите ссылку в левой части окна Настройка, Для того, чтобы получать из интернета сведения или новые средства устранения неполадок внизу окна должен быть установлен флажок Получить самые последние средства устранения неполадок через интернет-службу устранения неполадок Windows.

5. Убедитесь в том, что флажок Разрешить пользователям просматривать средства устранения неполадок, доступные через интернет -службы устранения неполадок Windows поставлен.
6. Пакеты поиска неисправностей разбиты по категориям:
    a. Программы;
    b. Оборудование и звук;
    c. Сеть и интернет;
    d. Оформление и персонализация;
    e. Система и безопасность.

Для просмотра полного списка пакетов, нажмите ссылку в левой части окна Просмотр всех категорий.

7. Откроется окно, содержащее полный список пакетов устранения неполадок. При этом Windows подключается к сети и проверяет наличие новых пакетов устранения неполадок. Подведите указатель мыши к интересующему пакету и увидите параметры пакета, включающие и его описание.
8. По умолчанию, в случае найденных ошибок, мастер устранения неполадок применяет изменения автоматически. Если щелкнуть ссылку Дополнительн ов первом окне мастера и снять флажок Автоматически исправлять ошибки, то при обнаружении неполадки будет предложен список возможных путей ее устранения
9. В любом случае по окончании диагностического теста выводится отчет. Для демонстрации работы компонента Устранения неполадок вручную остановим службу Диспетчер сеансов диспетчера окон рабочего стола. В целом эффект AERO работает, но исчезла прозрачность окон. Для устранения этой неполадки воспользуемся пакетом устранения неполадок Aero. Щелчок по ссылке откроет первое окно мастера. Оставим все без изменений и нажмем кнопку Далее.
10. Мастер продолжит свою работу и начнет диагностику неполадок.
11. Если бы мы воспользовались ссылкой Дополнительнои сняли флажок Автоматически применять исправления-нам был бы предложен список возможных путей устранения этой неполадки. Установив флажки и щелкнув кнопку Далее мы бы применили предложенные исправления.
12. Все произведенные тесты сохраняются в журнале просмотреть который можно щелкнув ссылку Просмотр журнала в левой части окна компонента Устранения неполадок. Подробный отчет можно увидеть, дважды щелкнув по его названию в списке или нажав кнопку Подробности(одноименный пункт есть и в контекстном меню).
13. Если компонент Устранения неполадок удалось решить проблему, можно закрыть его. В противном случае воспользуйтесь ссылкой Просмотреть дополнительные параметры, на экране будет отображен запрос с несколькими вариантами поиска решения по устранению неполадки в Интернете.
14. В левой части окна компонента Устранения неполадок присутствует еще одна ссылка Обратиться за помощью к другу. Если у вас есть друзья, которые хорошо разбираются в компьютерах, предоставьте другу доступ через Интернет к своему компьютеру с помощью Удаленного помощника Windows, чтобы друг помог решить проблему. При этом вы можете следить за его действиями и принимать участие в этом процессе. Так же можно использовать Средство записи действий по воспроизведению неполадок, которое может помочь в выявлении и устранении проблем

Практическая работа № 4.6. Устранение проблем совместимости программного обеспечения

Цель работы: научиться определять совместимость программного обеспечения и устранять проблемы совместимости.

Теоретический материал

Совместимость – способность аппаратных или программных компонентов работать с заданной компьютерной системой, или способность двух устройств работать при соединении друг с другом.

При отсутствии совместимости могут возникать различные виды конфликтов, мешающие или делающие невозможной нормальную работу компьютерной системы. Чаще всего конфликты возникают при установке нового оборудования или программного обеспечения.

Конфликты делятся на аппаратные, программные и программно-аппаратные.

Аппаратные конфликты – это конфликты чаще всего возникающие при сборке оборудования или при его установке в сети и приводящие к частичной или полной неработоспособности устройства. Чтобы избежать таких конфликтов, при сборке ПК необходимо соблюдать следующие правила.

1 Материнская плата и корпус должны быть одного формата (например ATX). Сокеты материнской платы и процессора также должны совпадать (например, у процессора – Socket LGA775, а у материнской платы – Socket 775).

2 Материнская плата должна поддерживать частоту шины процессора. Например, если процессор поддерживает частоту 1333 МГц, то и материнская плата должна поддерживать частоту шины 1333 МГц.

3 Необходимо обратить внимание на звуковую, сетевую и видеокарту, если они не встроенные. Они должны плотно входить в разъемы на материнской плате.

4. Оперативная память также должна быть совместима с материнской платой (они должны поддерживать одинаковую частоту).

При установке компьютера в локальной сети при возникновении конфликтов нужно проверить не только правильность установки сетевой карты, но и правильность обжима кабеля; кроме того, кабель может быть просто поврежден.

Также при установке нескольких карт расширения может возникнуть конфликт адресов BIOS, номеров прерываний или каналов прямого доступа к памяти.

Программные конфликты чаще всего возникают при установке драйверов устройств или другого программного обеспечения и приводят к частичной или полной неработоспособности устройства либо сети.

Программные неисправности при сборке или установке оборудования встречаются намного чаще, чем аппаратные, и возникают не только из-за неправильно установленных драйверов устройств, но и из-за нестабильности работы программного обеспечения.

Основные причины возникновения программных ошибок:

1 Несовершенство программного обеспечения.

2 Несовершенство операционной системы. Какими бы совершенными ни были операционные системы, они не могут создать нормальные условия для работы всего существующего программного обеспечения. Кроме того, совместимость операционных систем с выпуском каждой новой их версии только ухудшается. Поэтому разработчики ПО вынуждены писать программы, ориентированные на конкретную операционную систему. Пользователю же остается либо обновлять прикладное ПО вместе с операционной системой, либо мириться со сложившейся ситуацией. А иногда и выбирать не приходится, – ведь многие программы распространяются бесплатно (можно догадаться, какое у них в таком случае качество).

3 Отсутствие ресурсов.

4 Ошибки в реестре. Реестр — это «мозг» операционной системы Windows, и ошибки в нем негативно сказываются на всех процессах, происходящих в компьютере. Причиной возникновения сбоев в реестре являются все те же программы, «прописывающие» свои файлы и ссылки в самых различных местах. Не стоит также забывать и о «троянских конях» и «червях».

Для «лечения» реестра существуют специальные утилиты, умеющие анализировать его записи и удалять из реестра ошибочные и не используемые данные.

Довольно часто возникает проблема с драйверами, когда пользователь устанавливает новое оборудование. Это может происходить из-за частичной несовместимости англоязычной и русскоязычной версий Windows, в результате чего возникает повреждение базы драйверов.

Решить эту проблему можно, создав такую ситуацию, когда операционная система сама восстановит поврежденную базу, так как база драйверов – это не окончательно сформированный файл, операционная система создает его в процессе своей установки. После установки Windows закрывает доступ к этой базе для предупреждения ошибочного воздействия пользователя на нее. Однако во время установки или удаления различного оборудования операционная система временно открывает доступ к этой базе для внесения туда новых драйверов. Например, если при установке новой видеокарты ПК ее просто «не видит», то для устранения этой проблемы необходимо отключить компьютер, вынуть видеокарту, снова включить систему без видеокарты, дождаться звукового сигнала, который оповещает об отсутствии видеокарты, вновь выключить компьютер, снова вставить видеокарту и затем опять включить компьютер.

В ряде случаев такие действия помогают. После этого необходимо удалить старый драйвер и поставить новый. Если же система не отреагировала на ваши действия, то придется обнулить CMOS.

Другой пример. При установке драйвера новой видеокарты компьютер перестает ее «видеть». Это означает, скорее всего, что для современной видеокарты была поставлена старая версия драйвера, которая не может поддерживать слишком современное оборудование. И наоборот, если видеокарта еле-еле работает, но определить ее ПК не может, то причина данного конфликта – в том, что на старую видеокарту поставили самый новый драйвер (хотя такое бывает редко). В этом случае в драйвере просто нет поддержки данной видеокарты, и система не может ее определить.

Программно-аппаратные конфликты совмещают в себе конфликты и программного, и аппаратного характера, причем для их разрешения зачастую достаточно программно изменить ряд параметров. Рассмотрим несколько таких примеров.

Как известно, прежде операционной системы в компьютере запускается встроенная в чип материнской платы программа BIOS (Base Input/Output System – основная система ввода-вывода). Назначение этого небольшого программного кода – свести к «общему знаменателю» аппаратные различия компьютерного оборудования. Надежная и эффективная работа ПК невозможна без правильно сконфигурированного BIOS. Конфликт же между новейшим оборудованием и устаревшим кодом BIOS — вещь довольно частая. В таком случае выход один: перепрошивка BIOS.

Другим источником конфликтов данного вида является механизм Plug and Play операционной системы Windows, который автоматически выделяет ресурсы в ходе установки всех устройств, поддерживающих данный механизм. Если два устройства обращаются к одним и тем же ресурсам, то возникает аппаратный конфликт. В этом случае необходимо вручную изменить установки ресурсов для обеспечения их уникальности для каждого устройства. Сделать это можно двумя способами, в зависимости от того, насколько имеющийся конфликт мешает загрузке операционной системы.

Если Windows загружается, но при этом не работают (или работают некорректно) некоторые устройства, то достаточно изменить указанные выше ресурсы в оснастке Диспетчер устройств. Если же процесс загрузки Windows прерывается, потому что не могут быть обнаружены жесткие диски или устройства, установленные в PCI-слот, то необходимо просмотреть таблицу прерываний, которую выводит BIOS после процедуры POST, найти устройства с одинаковым номером прерывания и вручную задать одному из них свободное прерывание в таблице свойств PCI системной BIOS.

Таким образом, тестирование совместимости аппаратного и программного обеспечения проводится по минимальным системным требованиям и дополнительным ресурсам, необходимым тому или иному программному обеспечению.

ХОД РАБОТЫ

Задание. Составьте таблицу, содержащую минимальные системные требования для программ, необходимые для тестирования на совместимость**.**

Программа	Частота процессора	Объем оперативной памяти	Свободный объем жесткого диска	Дополнительные требования
Windows 7 Максимальная x64				
Microsoft Office 2013				
Photoshop CS4				
КОМПАС-3D V13				

Практическая работа № 4.7. Конфигурирование программных и аппаратных средств

Цель работы: приобрести практические навыки анализа конфигурации ПК

Теоретические сведения

Под конфигурацией вычислительной машины понимают набор аппаратных и программных средств, входящих в ее состав. Минимальный набор аппаратных средств, без которых невозможен запуск, и работа вычислительной машины определяет ее базовую конфигурацию.

Анализ конфигурации вычислительной машины (рассмотрим на примере персонального компьютера) целесообразно проводить в следующей последовательности:

- внешний визуальный осмотр компьютера;
- анализ аппаратной конфигурации компьютера встроенными средствами операционной системы;
- анализ программной конфигурации компьютера;
- анализ конфигурации вычислительной сети, в случае если компьютер к ней подключен.
- В результате внешнего визуального осмотра компьютера определяются следующие данные по его конфигурации:
- тип корпуса системного блока (форм-фактор);
- виды и количество интерфейсов для подключения периферийных устройств, размещенные на задней стенке и лицевой панели системного блока;
- тип клавиатуры и способ ее подключения к компьютеру (количество клавиш, наличие специальных клавиш);
- тип ручного манипулятора (мыши) и способ ее подключения к компьютеру (манипулятор с механической или оптической системой позиционирования, проводной или беспроводный интерфейс подключения);
- тип монитора (ЭЛТ или жидкокристаллический).

Анализ аппаратной конфигурации компьютера, т.е. состава подключенных аппаратных средств, можно проанализировать специальными тестовыми программами, либо встроенными средствами операционной системы, включающей такое понятие как диспетчер устройств.

Для просмотра содержимого диспетчера устройств найдите на рабочем столе ярлык ***Компьютер***, далее выделите его и нажмите правую клавишу мыши. В открывшемся контекстном меню выберите пункт ***Свойства*** (рис. 1). В результате этого действия откроется окно ***Свойства системы*** (рис.2).

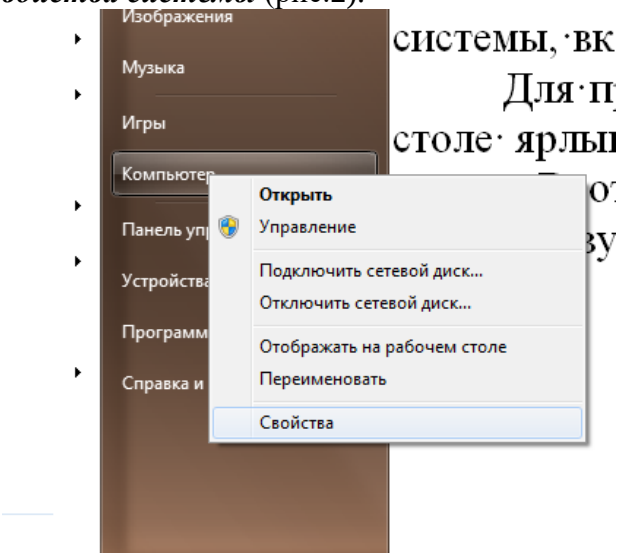

Рисунок 1.

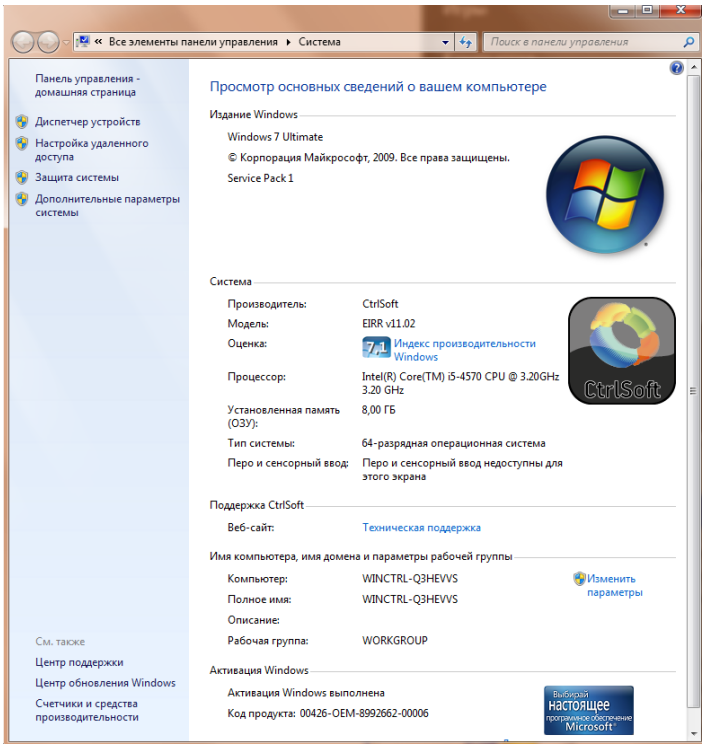

Рисунок 2.

В окне ***Система*** просмотрите и зафиксируйте версию операционной системы, тип процессора и его тактовую частоту, а также объем оперативной памяти (ОЗУ). Далее перейдите к закладке ***Диспетчер устройств*** (рис. 3.3).

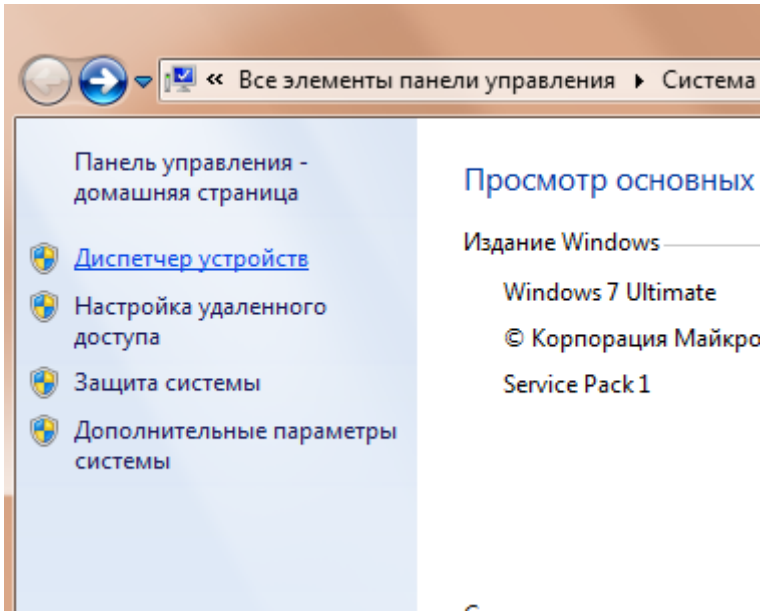

Рисунок 3.

В открывшемся окне ***диспетчера устройств*** (рис. 4) представлено графическое отображение перечня оборудования компьютера.

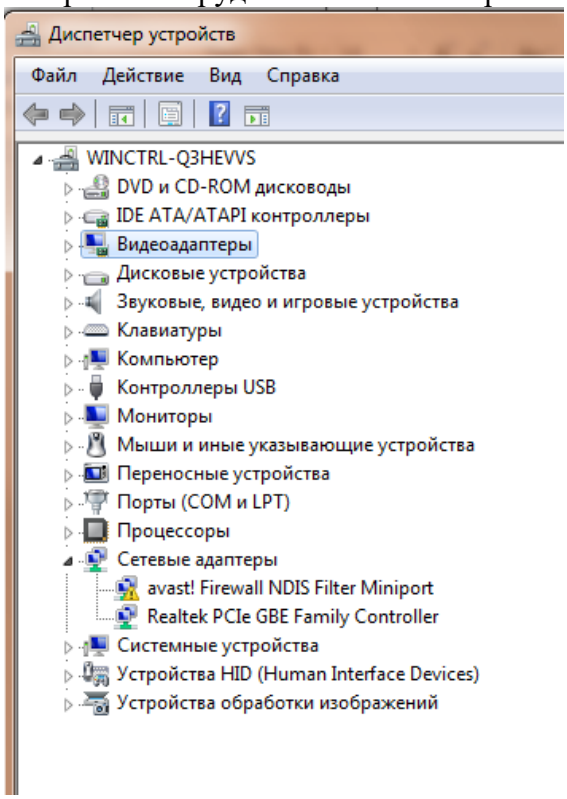

Рисунок 4.

Диспетчер устройств можно использовать для обновления драйверов (или программного обеспечения) оборудования, изменения настроек оборудования, а также для устранения неполадок и даже выключения оборудования из конфигурации компьютера.

Для получения доступа к указанным возможностям необходимо выделить из перечня оборудования требуемое устройство и щелкнуть дважды мышью (рис. 5). Для просмотра содержимого каждого пункта перечня оборудования необходимо дважды нажать на названии соответствующей группы оборудования.

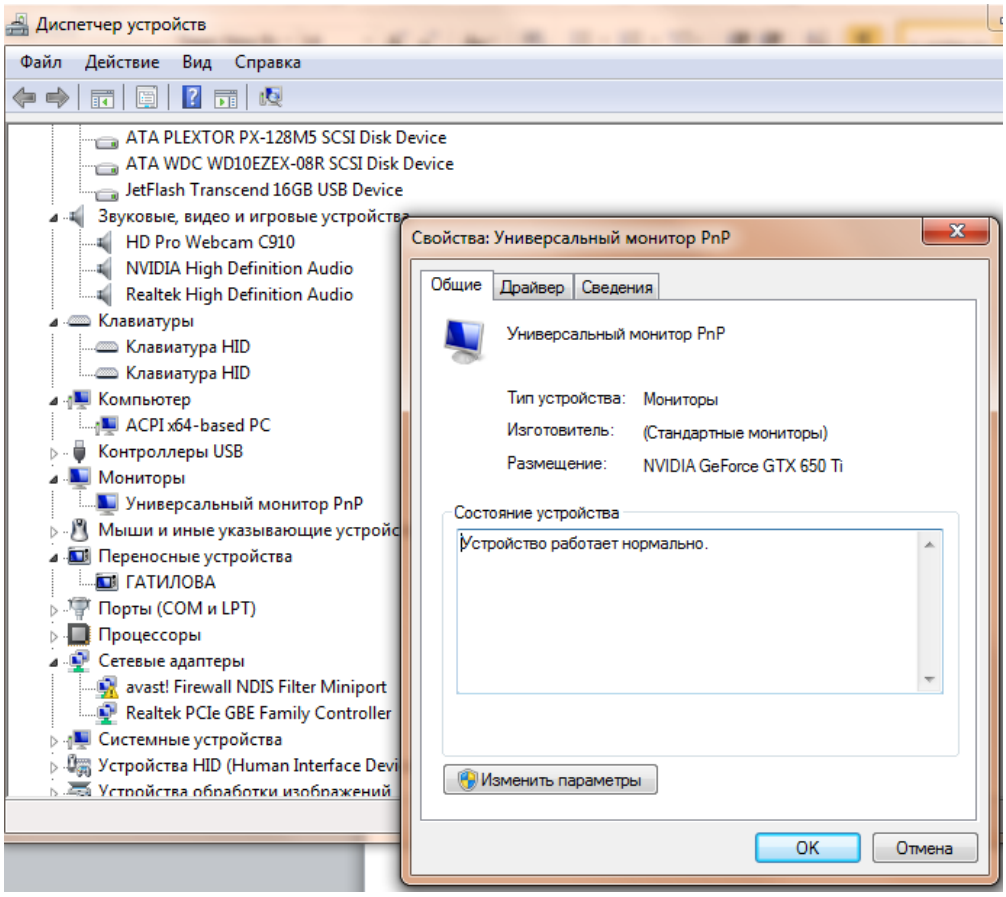

Рисунок 5.

Диспетчер устройств также позволяет:

- определять правильность работы оборудования компьютера;
- изменять параметры конфигурации оборудования;
- определять драйверы устройств, загружаемые для каждого устройства, и получать сведения о каждом драйвере;
- изменять дополнительные параметры и свойства устройств;
- устанавливать обновленные драйверы устройств;
- отключать, включать и удалять устройства;
- осуществлять возврат к предыдущей версии драйвера;
- распечатывать список устройств, установленных на компьютер.

Современные ***Операционные системы*** предоставляют пользователю возможность настройки и загрузки различных конфигураций аппаратных средств в рамках одного компьютера. С этой целью введено понятие ***Профиль оборудования***.

Профиль оборудования - это набор инструкций, используемых Windows для определения устройств, которые должны загружаться при запуске компьютера, или параметров для каждого устройства. При первой установке Windows создается профиль оборудования "Profile 1". По умолчанию все устройства, присутствующие на компьютере на момент установки Windows, включены в "Profile 1".

Вновь создаваемый пользователем профиль оборудования может не включать какое-то из устройств, например, модем или сетевой адаптер, или накопитель гибких магнитных дисков и др.

Если в системе имеется несколько профилей оборудования, можно указать среди них тот, который будет использоваться по умолчанию при каждом запуске компьютера. Windows позволяет также отображать при запуске вопрос, какой профиль следует использовать. После создания профиля оборудования устройства, входящие в него, можно отключать и включать с помощью диспетчера устройств. ***При отключении устройства в профиле оборудования драйверы устройства не загружаются при запуске компьютера.***

Более широкие возможности по анализу конфигурации компьютера, в том числе и программной среды, предоставляет модуль ***Сведения о системе***.

Для доступа к указанному модулю выберите последовательно команды: ***Пуск\Все программы\Стандартные\Служебные\Сведения о системе.***

В результате этого действия откроется окно *Сведения о системе* (рис. 6).

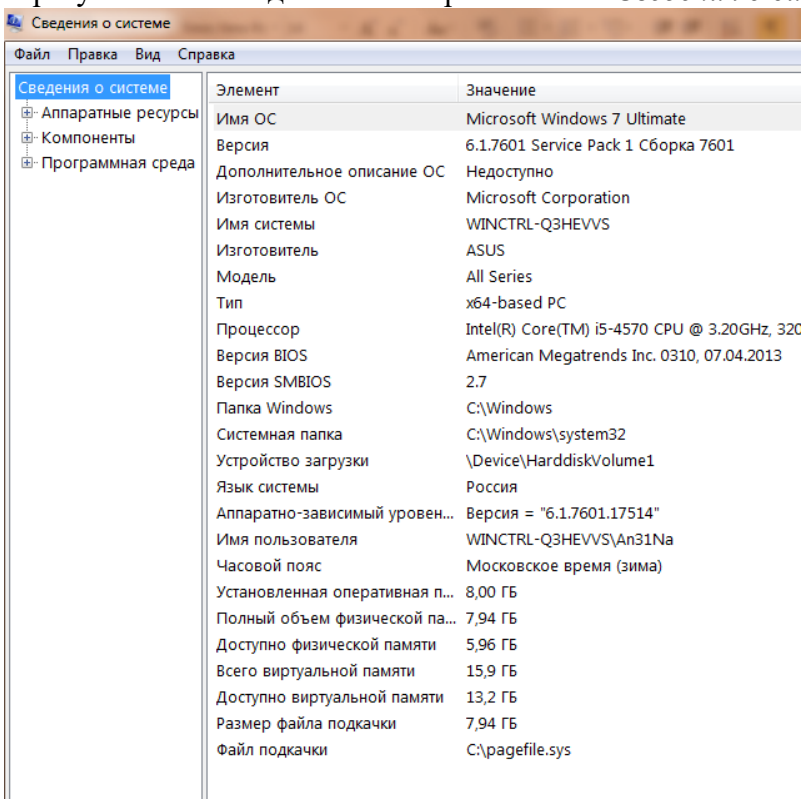

Рисунок 6.

Пример использования модуля ***Сведения о системе*** иллюстрируется на рис. 7, где показаны свойства из подпункта ***Дисплей*** группы ***Компоненты***.

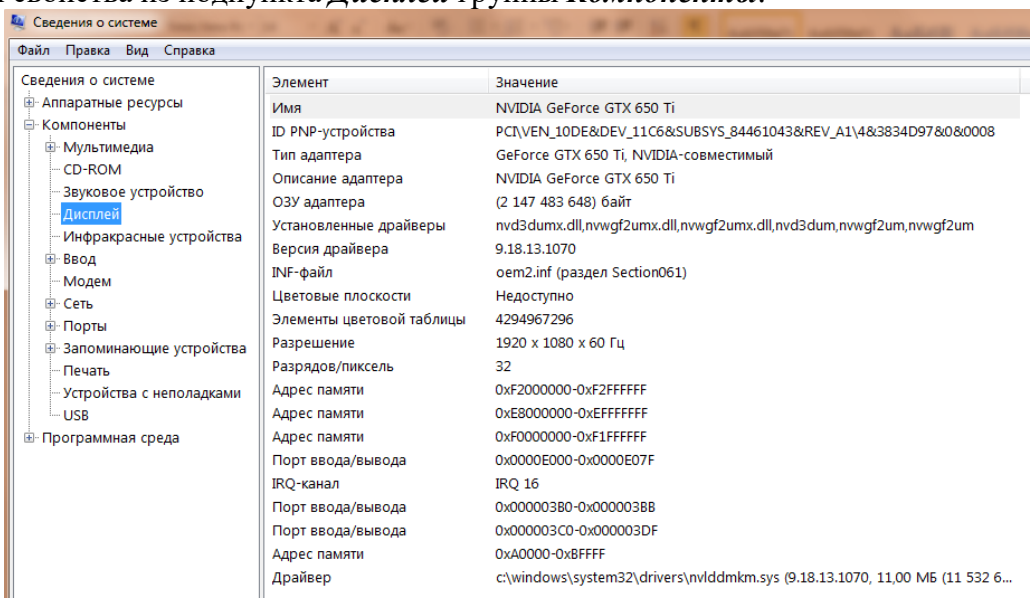

Рисунок 7.

В данном случае можно получить полную информацию о видеоадаптере, что отображается в правой части открытого окна. Аналогично может быть получена информация о других устройствах, а также о программной среде компьютера. Для этого необходимо выбрать соответствующие пункты в левой части окна ***Сведения о системе***.

Для анализа программной среды вычислительной машины помимо модуля ***Сведения о системе*** можно непосредственно просмотреть полный перечень установленного программного обеспечения, который вызывается последовательным выбором команд ***Пуск*** и далее ***Все программы***.

Для анализа конфигурации вычислительной сети необходимо выбрать на рабочем столе ярлык ***Сетевое окружение*** или команду ***Сетевое окружение*** после выбора команды ***Пуск***.

В открывшемся окне в случае подключения компьютера к локальной сети можно проанализировать конфигурацию сети.

Ход работы

Задание 1. Заполните таблицу (в таблицу следует заносить только реальные данные по конфигурации Вашего компьютера, в случае отсутствия какого-либо устройства ставится прочерк).

п/п	**Наименование параметра**	**Значение параметра**
1.	Тип и модель монитора	
2.	Форм-фактор корпуса системного блока	
3.	Клавиатура, интерфейс подключения	
4.	Вид манипулятора "мыши", интерфейс ее подключения	
5.	Интерфейсы подключения периферийных устройств на задней панели системного блока (наименование и количество)	
6.	Интерфейсы подключения периферийных устройств на лицевой панели системного блока (наименование и количество)	
7.	Процессор, модель и тактовая частота	
8.	Объем оперативной памяти	
9.	Тип модема и сетевого интерфейса	
10.	Наименование и скорость привода для чтения оптических дисков	
11.	Модель и объем памяти накопителя на жестких магнитных дисках	
12.	Видеоадаптер, модель и объем видеопамяти	
13.	Модель звукового адаптера	
14.	Версия операционной системы	
15.	Другие периферийные устройства (принтер, сканер и т.д.)	

Задание 2. Создайте иллюстрацию, аналогичную рис. 4. Для этого откройте соответствующее окно и скопируйте содержимое экрана в буфер нажатием на клавиатуре клавиши Print Screen. После этого вставьте содержимое буфера в документ Microsoft Word, сохраните документ.

Вопросы для самоконтроля

1. Что понимается под конфигурацией вычислительной машины?
2. Какова последовательность анализа конфигурации вычислительной машины?
3. Что понимается под профилем оборудования? Каковы преимущества системы с настраиваемым профилем оборудования?
4. Какие инструменты операционной системы Windows используются для анализа конфигурации компьютера.

Практическая работа № 4.8. Настройки системы и обновлений

Цель работы: возможности системы и обновлений

Ход работы

Microsoft регулярно выпускает обновления для улучшения операционной системы Windows 10, исправления ошибок и устранения проблем безопасности. У компании не всегда все складывается удачно: обновления могут быть навязчивыми, запутанными и содержащими большое количество проблем. Пользователи же хотят, чтобы обновления не беспокоили их в момент продуктивной работы, чтобы устанавливались только необходимые обновления, и чтобы обновления не создавали новые проблемы.

1. Как отключить автоматическое обновление Windows 10

В первую очередь, нужно настроить временные интервалы для установки обновлений. Вы можете запланировать период активности, чтобы предотвратить перезагрузку ПК после установки обновления. Кроме того, вы можете посмотреть историю обновлений, чтобы проверить, были ли установлены только нужные обновления. Можно перейти в расширенные настройки, чтобы определить, какие обновления вы получите и когда. Давайте остановимся подробнее на возможностях работы с обновлениями в Windows 10.

Предыдущие версии Windows предлагали апплет «Центр обновления Windows» для панели управления, который использовался для просмотра и управления обновлениями. В Windows 10 разработчики отказались от апплета в пользу раздела Обновления и безопасность приложения Параметры. Откройте приложение Параметры и перейдите в раздел «Обновление и безопасность». Убедитесь, что экран обновления Windows остается активным.

2. Проверить наличие обновлений

Используйте кнопку Проверить наличие обновлений, чтобы посмотреть, нужно ли устанавливать какие-либо обновления. Дождитесь установки обновлений, чтобы ваша система пришла в актуальное состояние.

3. Обратите внимание

Чтобы избежать установки предварительных обновлений в стабильных версиях Windows 10, стоит отказаться от ручного обновления системы – использования кнопки «Проверить наличие обновлений»

4. Дополнительные обновления

Дополнительные обновления Windows 10

Начиная с Windows 10, версия 1903 предварительные обновления не устанавливаются автоматически при использовании кнопки «Проверить наличие обновлений», а доступны для установки вручную в отдельном блоке Доступны дополнительные обновления в Центре обновления Windows.

Чтобы установить предварительное обновление необходимо нажать кнопку Загрузить и установить сейчас. Прежде чем установить такое обновление, подробнее об изменениях и улучшениях можно узнавать в нашем разделе Обновления Windows 10.

5. Обновление функций

Обновление функций до Windows 10, версия 1903. Загрузить и установить сейчас

Microsoft выпускает крупные обновления функций дважды в год. Когда ваше устройство будет готово к установке очередного обновления функций, вы увидите соответствующее уведомление в Центре обновления Windows.

Обновление функций будет установлено только тогда, когда администратор устройства одобрит установку, выбрав «Загрузить и установить сейчас» для соответствующего обновления. Единственное исключение из правила — это приближение срока окончания поддержки конкретной версии Windows 10. В этом случае, обновление будет установлено принудительно, если ваш компьютер совместим с ним.

6. Обновление функций до Windows 10, версия 1903

Если на вашем устройстве по какой-либо причине была заблокирована установка обновления функций, то в Центре обновления Windows вы получите уведомление, что новая версия доступна, но компьютер еще не готов ее получить.

7. Планируем перезагрузку

   Планируем перезагрузку

   После установки обновления Windows спросит пользователя, можно ли выполнить перезагрузку прямо сейчас или нужно запланировать обновление. Если вы работаете над важным документом или выполняете другую важную активность, то перезагружать компьютер не нужно. Вместо этого нажмите ссылку Запланировать перезапуск и выберите время и дату для перезагрузки ПК, чтобы применить обновления.

8. Планируем перезагрузку

   Изменяем период активности

   Изменяем период активности

   Следующим шагом вы можете сообщить Windows, когда вы обычно используете компьютер, чтобы предотвратить прерывания из-за перезагрузки. На странице обновления Windows нажмите ссылку Изменить период активности. Установите интервал времени, в течение которого Windows не будет выполнить перезагрузку после установки обновления. Нажмите кнопку «Сохранить».

9. История обновления

   В любое время вы можете посмотреть, какие обновления были установлены в систему, чтобы проверить отдельные обновления и убедиться, чтобы были установлены только нужные обновления. Нажмите ссылку Просмотр журнала обновления. Windows показывает список всех недавних обновлений. Чтобы узнать больше об определенном обновлении, нажмите на соответствующую ссылку. Откроется страница обновления из Центра поддержки Microsoft, которая предоставит подробную информацию об обновлении, включая известные ошибки.

10. Удаляем обновления

    Действительно, Microsoft иногда выпускает обновления с серьезными ошибками, которые приносят больше вреда, чем пользы. Обычно Редмонд исправляет ошибки уже в следующем корректирующем обновлении. Если вы не хотите ждать исправления, то удаление обновления может стать эффективным способом, чтобы избавиться от проблем. На странице Просмотр журнала обновлений нажмите ссылку Удалить обновления. Windows откроет апплет Панели управления со списком установленных обновлений. Дважды щелкните по проблемному обновления для его удаления.

11. Приостановить обновления на 7 дней

    Приостановить обновления на 7 дней

    Начиная с Windows 10, версия 1903 в Центре обновления Windows стала доступно опция Приостановить обновление на 7 дн., которая позволяет откладывать все обновления, включая обновления безопасности. Если вы включите данную функцию, Windows не будет выполнять проверку и автоматическую установку всех обновлений в течение 7 дней. Опция доступна в том числе и для пользователей Windows 10 Домашняя, которые могут воспользоваться ей до 5 раз, и таким образом откладывать обновления на срок до 35 дней.

12. Как временно отключить обновления драйвера

    Как временно отключить обновления драйвера в Windows 10

    Microsoft также предлагает средство устранения неполадок Show or hide updates (Показать или скрыть обновления), которое позволяет скрывать обновления драйверов и предотвращать повторную установку до того, как станет доступна исправная версия.

    Скачайте пакет средства устранения неполадок с официального сайта Microsoft – wushowhide.diagcab. Это портативное приложение, запустите его – установка не требуется. Чтобы скрыть обновление проблемного драйвера воспользуйтесь опцией Hide updates, чтобы восстановить обновление – Show hidden updates.

13. Восстанавливаем Windows

    Восстанавливаем Windows

Иногда обновления серьезно нарушают работоспособность системы. В этом случае вам подойдет восстановление Windows 10 до более раннего состояния, которое было до установки обновления. На странице обновления Windows нажмите ссылку Просмотр журнала обновлений, а затем кликните по ссылке Параметры восстановления и выберите опцию Вернуть компьютер в исходное состояние.

14. Как переустановить Windows 10 и избавиться от проблем

Имейте в виду, что при сбросе системы будут удалены все установленные приложения и настроенные параметры. Прежде чем, выполнять полный сброс, попытайтесь восстановить систему с помощью точек восстановления. Откройте Панель управления (включите режим отображения Мелкие значки), затем перейдите в Система, нажмите ссылку Защита системы, а затем выберите кнопку «Восстановить». На первом экране выберите опцию «Выбрать другую точку восстановления». На следующем экране выберите подходящую точку восстановления.

15. Поиск затрагиваемых программ

Выберите точку восстановления, которая предшествовала установке обновления. Нажмите кнопку «Поиск затрагиваемых программ», чтобы увидеть, какое влияние окажет процесс восстановления на установленные приложения. Затем нажмите «Далее», чтобы перейти непосредственно к процессу восстановления.

16. Настраиваем расширенные параметры

Настраиваем расширенные параметры

На странице Центр обновления Windows выберите ссылку Дополнительные параметры, чтобы настроить различные функции обновления системы.

При обновлении Windows предоставить обновления для других продуктов Майкрософт – данный параметр гарантирует, что при установке системных обновлений Windows вы также получите обновления для Microsoft Office и других продуктов от Microsoft. Рекомендуется включить опцию.

Автоматически скачивать обновления даже через лимитные подключения данных (может взиматься плата) – данный параметр позволяет загружать обновления через мобильные подключения к Интернету. Если объем трафика на вашем тарифе ограничен, рекомендуем оставить данную опцию отключенной.

Перед перезапуском на экране появится напоминание. Чтобы получать больше уведомлений о перезапуске, включите параметр – данная опция позволяет показывать уведомление о перезагрузке ПК поверх стандартного оповещения. Если вы обычно не отвлекаетесь на стандартные уведомления, то лучше включить опцию. В противном случае, оставьте ее отключенной.

Приостановить обновления – данный параметр позволяет отложить установку обновлений на определенное количество дней. Если вы хотите своевременно получать последние обновления, оставьте параметр отключенным.

Выберите, когда устанавливать обновления

Параметры Выберите, когда устанавливать обновления больше подходят для организаций, чем обычным пользователям. Рекомендуется оставить настройки как есть. Если же вы хотите отложить установку крупных функциональных обновлений Windows 10 до 365 дней, то можете воспользоваться следующей инструкцией:

17. Как отложить установку обновления Windows 10

Нажмите ссылку Оптимизация доставки. На открывшейся странице вы может разрешить скачивание обновления из других компьютеров в вашей сети или из компьютеров в Интернете. При включении данной опции рекомендуется выбрать вариант Компьютеры в локальной сети При желании можно настроить дополнительные параметры оптимизации доставки, но большинству пользователей следует оставить параметры по умолчанию.

В после «Имя сервера» указываем локальную машину и название сервера, с которым будем работать. Вводим установленный в п.15 пароль для учётной записи «sa».

Далее видим типичный для Visual Studio интерфейс:

Практическая работа № 4.9. Создание образа системы. Восстановление системы

Цель работы: изучить технологию импорта данных пользователя в базу данных

Ход работы

Задание 1Создаем диск с образом системы

1. Кликаем по кнопке “Пуск” в левом нижнем углу рабочего стола системы. В открывшемся меню выбираем “Панель управления” ОС Windows 7. В открывшемся окне находим раздел “Система и безопасность” и щелкаем по подразделу “Архивирование данных компьютера”.
2. В открывшемся окне выбираем пункт “Создание образа системы”.
3. Запустится “Мастер создания образа системы”. Который предложит выбрать место сохранения архива. По-умолчанию будет указан пункт “На DVD-дисках”. Оставляем этот вариант и нажимаем кнопку “Далее”.
4. Появится окно “Подтверждения параметров архивации”. После проверки указанных параметров нажимаем кнопку “Активировать”. После чего начнется процесс подготовки и создания архива. Ход процесса будет отображаться в виде заполняющейся цветной полосы.
5. В ходе процесса Мастер предложит Вам вставить в привод чистый носитель информации размером более 1 Гб. Диск должен быть уже отформатирован. Сделать это можно в окне “Мой компьютер”, кликнув правой кнопкой мыши по приводу с DVD –диском. И в открывшемся меню выбрав пункт “Форматировать…”, либо “Стереть этот диск”. А мастер потом его сам отформатирует.
6. Далее Мастер приступит к записи, созданного архива на DVD-диск. Этот процесс может занять значительный промежуток времени.
7. По завершении этой операции появится окошко с предложением добавить на создаваемый диск информацию для аварийного восстановления системы. Как мы помним из статьи “Диск аварийного восстановления”, с помощью записанной на диск информации возможно вернуть систему в одно из состояний, записанных в виде контрольной точки восстановления.
8. После того, как Мастер закончит работу по созданию образа системы, откроется окошко с сообщением об успешном выполнении архивации системы.
9. Теперь диск с созданным образом системы можно извлечь из привода и закрыть окно Мастера, нажав кнопку “Закрыть”. Итак, диск с образом системы успешно создан. Соломенку мы подстелили. Себя подстраховали.

Восстановление системы из образа системы

Давайте рассмотрим ситуацию, когда у нас в результате действия вируса компьютер отказывается загружаться. Под рукой соответственно нет ни диска аварийного восстановления, ни установочного диска с ОС. Но мы заранее подготовили образ диска с системой.

1. Перезагружаем компьютер, нажав на кнопку “Reset”, либо путем выключения и повторного включения питания компьютера. При загрузке жмем клавишу “F8” и заходим в меню “Дополнительные варианты загрузки”.
2. Выбираем пункт “Устранение неполадок компьютера” и жмем клавишу “Enter”.
3. После этого отобразится окно “Параметров восстановления системы” с выбором необходимо языка меню восстановления и параметрами клавиатуры. Как правило, здесь менять ничего не надо, просто жмем кнопку “Далее”.
4. Далее появится окно “Параметры восстановления системы”, в котором необходимо указать Имя пользователя и пароль. Укажите необходимое Имя пользователя, введите пароль и нажмите кнопку “ОК”.
5. Откроется окошко, в котором необходимо будет выбрать вариант восстановления системы.

6. Далее вставьте в привод записанный ранее DVD-диск с образом системы и кликните по пункту “Восстановление образа системы”. Через некоторое время система найдет образ на DVD-диске и запустит “Мастера восстановления компьютера из образа”. На следующих двух окнах Мастера нажмите последовательно кнопку “Далее”, оставив предложенные варианты настройки. И на последнем окне нажмите кнопку “Готово”.
7. Нажав кнопку “Готово”, появится последнее окошко с предупреждением о том, что все данные на системном диске будут заменены данными из образа системы. Подтвердите свое намерения произвести восстановление системы с образа нажатием кнопки “Да”.
8. После этого Мастер приступит к процессу восстановления системы, который займет немного времени.
9. По завершении процесса восстановления ПК автоматически перезагрузится. Операционная система восстановлена из образа, все пользовательские данные сохранены, мы их не трогали.

Вот мы и завершили изучение вопроса восстановление работы компьютера из образа системы.

Практическая работа № 4.10. Разработка модулей программного средства

Цель работы: получить практические навыки разработки модулей программной системы и интеграции этих модулей

Теоретический материал

Термин «интеграция» относится к такой операции в процессе разработки ПО, при которой вы объединяете отдельные программные компоненты в единую систему. В небольших проектах интеграция может занять одно утро и заключаться в объединении горстки классов. В больших — могут потребоваться недели или месяцы, чтобы связать воедино весь набор программ. Независимо от размера задач в них применяются одни и те же принципы.

Тема интеграции тесно переплетается с вопросом последовательности конструирования. Порядок, в котором вы создаете классы или компоненты, влияет на порядок их интеграции: вы не можете интегрировать то, что еще не было создано. Последовательности интеграции и конструирования имеют большое значение.

Поскольку интеграция выполняется после того, как разработчик завершил модульное тестирование, и одновременно с системным тестированием, ее иногда считают операцией, относящейся к тестированию. Однако она достаточно сложна, и поэтому ее следует рассматривать как независимый вид деятельности.

Аккуратная интеграция обеспечивает:

- упрощенную диагностику дефектов;
- меньшее число ошибок;
- меньшее количество «лесов»;
- раннее создание первой работающей версии продукта;
- уменьшение общего времени разработки;
- лучшие отношения с заказчиком;
- улучшение морального климата;
- увеличение шансов завершения проекта;
- более надежные оценки графика проекта;
- более аккуратные отчеты о состоянии;
- лучшее качество кода;
- меньшее количество документации.

При разработке программного модуля целесообразно придерживаться следующего порядка

- изучение и проверка спецификации модуля, выбор языка программирования;
- выбор алгоритма и структуры данных;
- программирование (кодирование) модуля;
- шлифовка текста модуля;
- проверка модуля;
- компиляция модуля.

Первый шаг разработки программного модуля в значительной степени представляет собой смежный контроль структуры программы снизу: изучая спецификацию модуля, разработчик должен убедиться, что она ему понятна и достаточна для разработки этого модуля. В завершении этого шага выбирается язык программирования: хотя язык программирования может быть уже предопределен для всего ПС, все же в ряде случаев (если система программирования это допускает) может быть выбран другой язык, более подходящий для реализации данного модуля (например, язык ассемблера).

На втором шаге разработки программного модуля необходимо выяснить, не известны ли уже какие-либо алгоритмы для решения поставленной и или близкой к ней задачи. И если найдется подходящий алгоритм, то целесообразно им воспользоваться. Выбор подходящих структур данных, которые будут использоваться при выполнении модулем своих функций, в значительной степени предопределяет логику и качественные показатели разрабатываемого модуля, поэтому его следует рассматривать как весьма ответственное решение.

На третьем шаге осуществляется построение текста модуля на выбранном языке программирования. Обилие всевозможных деталей, которые должны быть учтены при реализации функций, указанных в спецификации модуля, легко могут привести к созданию весьма запутанного текста, содержащего массу ошибок и неточностей. Искать ошибки в таком модуле и вносить в него требуемые изменения может оказаться весьма трудоемкой задачей. Поэтому весьма важно для построения текста модуля пользоваться технологически обоснованной и практически проверенной дисциплиной программирования. Впервые на это обратил внимание Дейкстра сформулировав и обосновав основные принципы структурного программирования. На этих принципах базируются многие дисциплины программирования, широко применяемые на практике.

Следующий шаг разработки модуля связан с приведением текста модуля к завершенному виду в соответствии со спецификацией качества ПС. При программировании модуля разработчик основное внимание уделяет правильности реализации функций модуля, оставляя недоработанными комментарии и допуская некоторые нарушения требований к стилю программы. При шлифовке текста модуля он должен отредактировать имеющиеся в тексте комментарии и, возможно, включить в него дополнительные комментарии с целью обеспечить требуемые примитивы качества. С этой же целью производится редактирование текста программы для выполнения стилистических требований.

Шаг проверки модуля представляет собой ручную проверку внутренней логики модуля до начала его отладки (использующей выполнение его на компьютере), реализует общий принцип, сформулированный для обсуждаемой технологии программирования, о необходимости контроля принимаемых решений на каждом этапе разработки ПС.

И наконец, последний шаг разработки модуля означает завершение проверки модуля (с помощью компилятора) и переход к процессу отладки модуля.

Структурное программирование

При программировании модуля следует иметь в виду, что программа должна быть понятной не только компьютеру, но и человеку: и разработчик модуля, и лица, проверяющие модуль, и тестовики, готовящие тесты для отладки модуля, и сопроводители ПС, осуществляющие требуемые изменения модуля, вынуждены будут многократно разбирать логику работы модуля. В современных языках программирования достаточно средств, чтобы запутать эту логику

сколь угодно сильно, тем самым, сделать модуль трудно понимаемым для человека и, как следствие этого, сделать его ненадежным или трудно сопровождаемым. Поэтому необходимо принимать меры для выбора подходящих языковых средств и следовать определенной дисциплине программирования. В связи с этим Дейкстра и предложил строить программу как композицию из нескольких типов управляющих конструкций (структур), которые позволяют сильно повысить понимаемость логики работы программы. Программирование с использованием только таких конструкций назвали *структурным*.

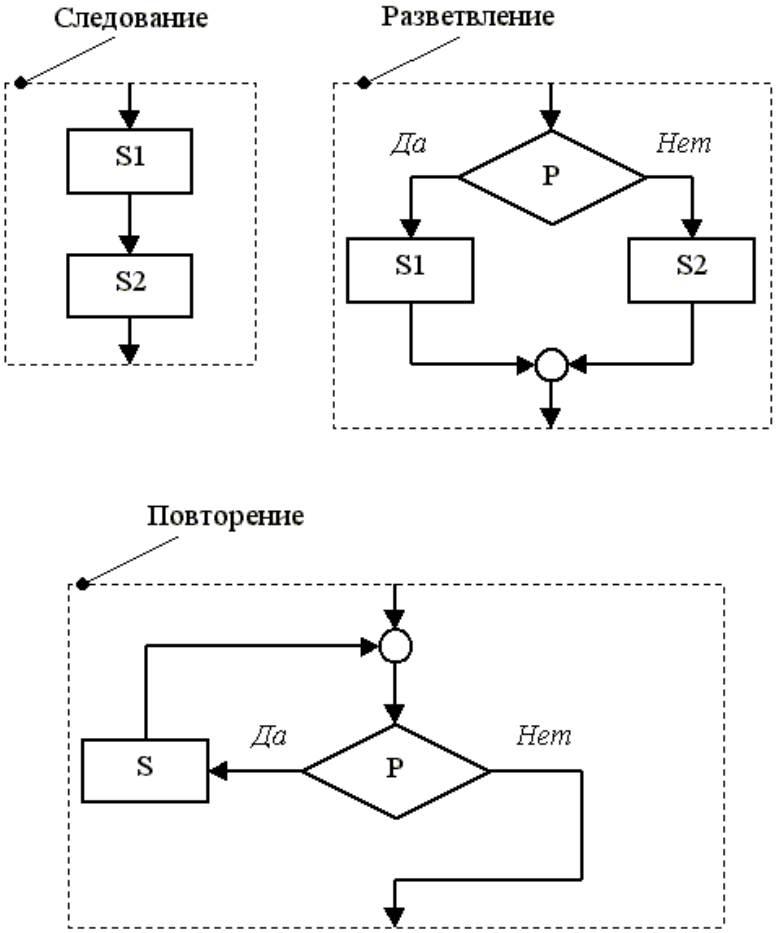

Рис. 1. Основные управляющие конструкции структурного программирования.

Основными конструкциями структурного программирования являются: следование, разветвление и повторение (см. Рис. 1). Компонентами этих конструкций являются обобщенные операторы (узлы обработки) S, S1, S2 и условие (предикат) P. В качестве обобщенного оператора может быть либо простой оператор используемого языка программирования (операторы присваивания, ввода, вывода, обращения к процедуре), либо фрагмент программы, являющийся композицией основных управляющих конструкций структурного программирования. Существенно, что каждая из этих конструкций имеет по управлению только один вход и один выход. Тем самым, и обобщенный оператор имеет только один вход и один выход.

Весьма важно также, что эти конструкции являются уже математическими объектами (что, по существу, и объясняет причину успеха структурного программирования). Доказано, что для каждой неструктурированной программы можно построить функционально эквивалентную (т.е. решающую ту же задачу) структурированную программу. Для структурированных программ можно математически доказывать некоторые свойства, что позволяет обнаруживать в программе некоторые ошибки. Этому вопросу будет посвящена отдельная лекция.

Структурное программирование иногда называют еще "программированием без GO TO". Однако дело здесь не в операторе GO TO, а в его беспорядочном использовании. Очень

часто при воплощении структурного программирования на некоторых языках программирования (например, на ФОРТРАНе) оператор перехода (GO TO) используется для реализации структурных конструкций, что не нарушает принципов структурного программирования. Запутывают программу как раз "неструктурные" операторы перехода, особенно переход к оператору, расположенному в тексте модуля выше (раньше) выполняемого оператора перехода. Тем не менее, попытка избежать оператора перехода в некоторых простых случаях может привести к слишком громоздким структурированным программам, что не улучшает их ясность и содержит опасность появления в тексте модуля дополнительных ошибок. Поэтому можно рекомендовать избегать употребления оператора перехода всюду, где это возможно, но не ценой ясности программы.

К полезным случаям использования оператора перехода можно отнести выход из цикла или процедуры по особому условию, "досрочно" прекращающего работу данного цикла или данной процедуры, т.е. завершающего работу некоторой структурной единицы (обобщенного оператора) и тем самым лишь локально нарушающего структурированность программы. Большие трудности (и усложнение структуры) вызывает структурная реализация реакции на возникающие исключительные (часто ошибочные) ситуации, так как при этом требуется не только осуществить досрочный выход из структурной единицы, но и произвести необходимую обработку (исключение) этой ситуации (например, выдачу подходящей диагностической информации). Обработчик исключительной ситуации может находиться на любом уровне структуры программы, а обращение к нему может производиться с разных нижних уровней. Вполне приемлемой с технологической точки зрения является следующая "неструктурная" реализация реакции на исключительные ситуации [8.7]. Обработчики исключительных ситуаций помещаются в конце той или иной структурной единицы и каждый такой обработчик программируется таким образом, что после окончания своей работы производит выход из той структурной единицы, в конце которой он помещен. Обращение к такому обработчику производится оператором перехода из данной структурной единицы (включая любую вложенную в нее структурную единицу).

Пошаговая детализация и понятие о псевдокоде

Структурное программирование дает рекомендации о том, каким должен быть текст модуля. Возникает вопрос, как должен действовать программист, чтобы построить такой текст. Часто программирование модуля начинают с построения его блок-схемы, описывающей в общих чертах логику его работы. Однако современная технология программирования не рекомендует этого делать без подходящей компьютерной поддержки. Хотя блок-схемы позволяют весьма наглядно представить логику работы модуля, при их ручном кодировании на языке программирования возникает весьма специфический источник ошибок: отображение существенно двумерных структур, какими являются блок-схемы, на линейный текст, представляющий модуль, содержит опасность искажения логики работы модуля, тем более, что психологически довольно трудно сохранить высокий уровень внимания при повторном ее рассмотрении. Исключением может быть случай, когда для построения блок-схем используется графический редактор и они формализованы настолько, что по ним автоматически генерируется текст на языке программирования (как, например, это делается в Р-технологии).

В качестве основного метода построения текста модуля современная технология программирования рекомендует *пошаговую детализацию*. Сущность этого метода заключается в разбиении процесса разработки текста модуля на ряд шагов. На первом

шаге описывается общая схема работы модуля в обозримой линейной текстовой форме (т.е. с использованием очень крупных понятий), причем это описание не является полностью формализованным и ориентировано на восприятие его человеком. На каждом следующем шаге производится уточнение и детализация одного из понятий (будем называть его *уточняемым*), в каком либо описании, разработанном на одном из предыдущих шагов. В результате такого шага создается описание выбранного уточняемого понятия либо в терминах базового языка программирования (т.е. выбранного для представления модуля), либо в такой же форме, что и на первом шаге с использованием новых уточняемых понятий. Этот процесс завершается, когда

все уточняемые понятия будут *уточнения* (т.е. в конечном счете будут выражены на базовом языке программирования). Последним шагом является получение текста модуля на базовом языке программирования путем замены всех вхождений уточняемых понятий заданными их описаниями и выражение всех вхождений конструкций структурного программирования средствами этого языка программирования.

Пошаговая детализация связана с использованием частично формализованного языка для представления указанных описаний, который получил название *псевдокода*. Этот язык позволяет использовать все конструкции структурного программирования, которые оформляются формализованно, вместе с неформальными фрагментами на естественном языке для представления обобщенных операторов и условий. В качестве обобщенных операторов и условий могут задаваться и соответствующие фрагменты на базовом языке программирования.

Головным описанием на псевдокоде можно считать внешнее оформление модуля на базовом языке программирования, которое

должно содержать:

- начало модуля на базовом языке, т.е. первое предложение или заголовок (спецификацию) этого модуля;
- раздел (совокупность) описаний на базовом языке, причем вместо описаний процедур и функций - только их внешнее оформление;
- неформальное обозначение последовательности операторов тела модуля как одного обобщенного оператора (см. ниже), а также неформальное обозначение тела каждого описания процедуры или функции как одного обобщенного оператора;
- последнее предложение (конец) модуля на базовом языке.

Внешнее оформление описания процедуры или функции представляется аналогично. Впрочем, если следовать Дейкстре, раздел описаний лучше также представить здесь неформальным обозначением, произведя его детализацию в виде отдельного описания.

Неформальное обозначение обобщенного оператора на псевдокоде производится на естественном языке произвольным предложением, раскрывающим в общих чертах его содержание. Единственным формальным требованием к оформлению такого обозначения является следующее: это предложение должно занимать целиком одно или несколько графических (печатных) строк и завершаться точкой (или каким-либо другим знаком, специально выделенным для этого).

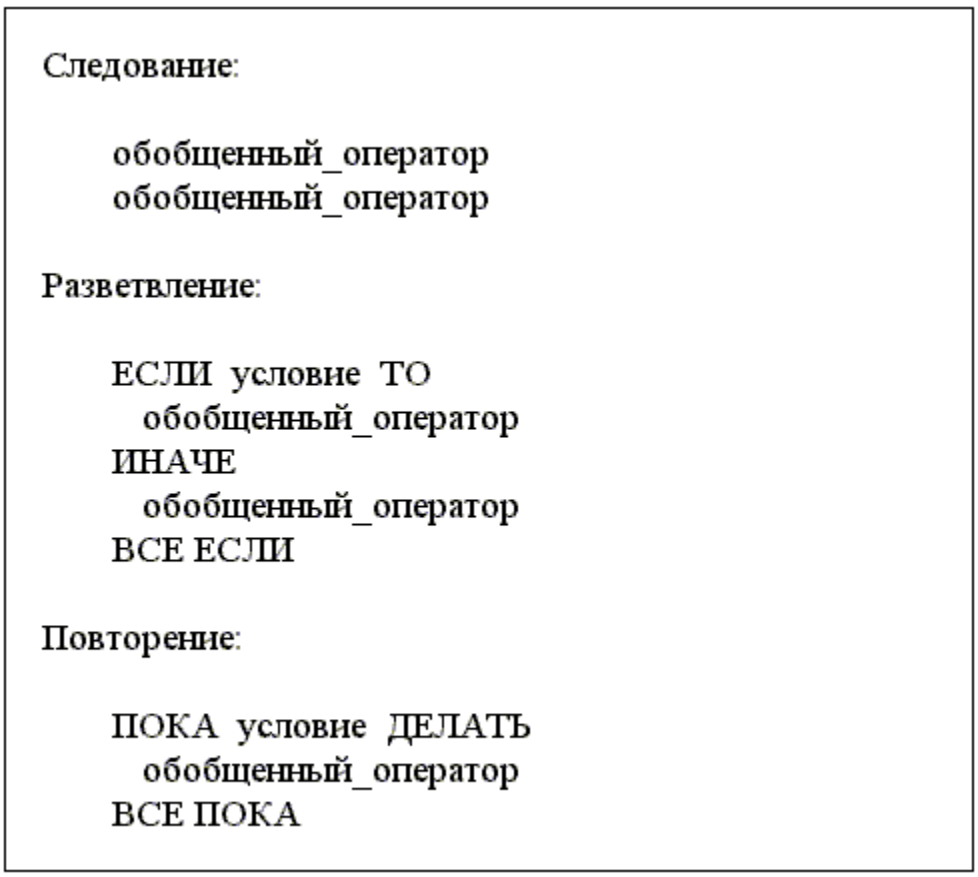

Рис. 2. Основные конструкции структурного программирования на псевдокоде.

Для каждого неформального обобщенного оператора должно быть создано отдельное описание, выражающее логику его работы (детализирующее его содержание) с помощью композиции основных конструкций структурного программирования и других обобщенных операторов. В качестве заголовка такого описания должно быть неформальное обозначение детализируемого обобщенного оператора. Основные конструкции структурного программирования могут быть представлены в следующем виде (см. рис. 2). Здесь условие может быть либо явно задано на базовом языке программирования в качестве булевского выражения, либо неформально представлено на естественном языке некоторым фрагментом, раскрывающим в общих чертах смысл этого условия. В последнем случае должно быть создано отдельное описание, детализирующее это условие, с указанием в качестве заголовка обозначения этого условия (фрагмента на естественном языке).

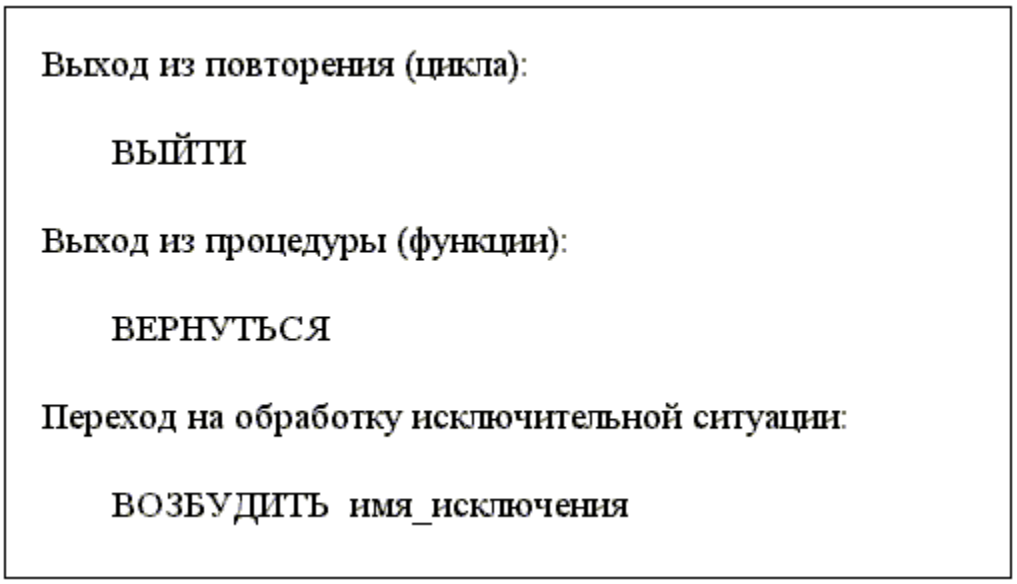

Рис. 3. Частные случаи оператора перехода в качестве обобщенного оператора.

В качестве обобщенного оператора на псевдокоде можно использовать указанные выше частные случаи оператора перехода (см. рис. 8.3). Последовательность обработчиков исключительных ситуаций (исключений) задается в конце модуля или описания процедуры (функции). Каждый такой обработчик имеет вид:

ИСКЛЮЧЕНИЕ имя_исключения
обобщенный_оператор
ВСЕ ИСКЛЮЧЕНИЕ

Отличие обработчика исключительной ситуации от процедуры без параметров заключается в следующем: после выполнения процедуры управление возвращается к оператору, следующему за обращением к ней, а после выполнения исключения управление возвращается к оператору, следующему за обращением к модулю или процедуре (функции), в конце которого (которой) помещено данное исключение.

Рекомендуется на каждом шаге детализации создавать достаточно содержательное описание, но легко обозримое (наглядное), так чтобы оно размещалось на одной странице текста. Как правило, это означает, что такое описание должно быть композицией пяти-шести конструкций структурного программирования. Рекомендуется также вложенные конструкции располагать со смещением вправо на несколько позиций (см. рис. 4). В результате можно получить описание логики работы по наглядности вполне конкурентное с блок-схемами, но обладающее существенным преимуществом - сохраняется линейность описания.

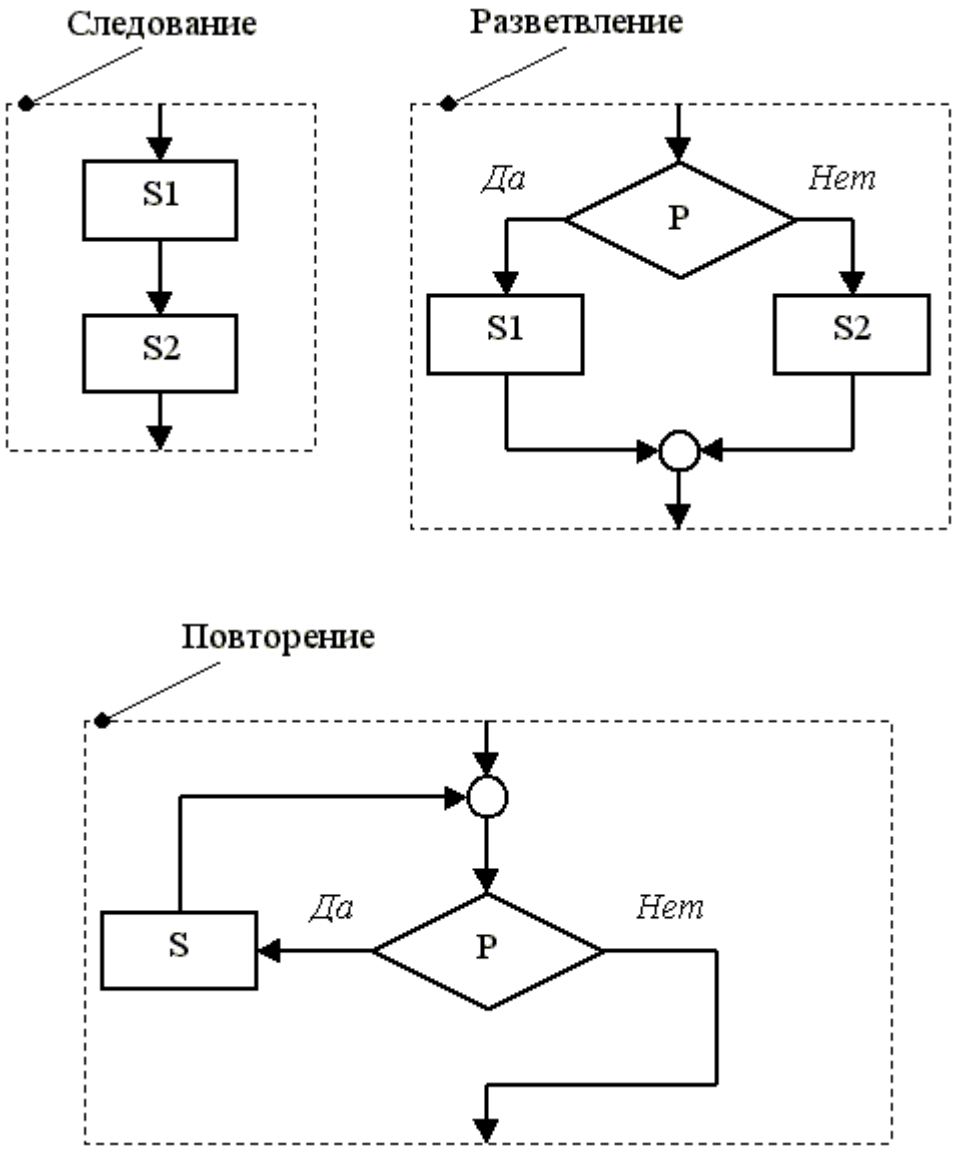

Рис. 4. Пример одного шага детализации на псевдокоде.

Идею пошаговой детализации приписывают иногда Дейкстре. Однако Дейкстра предлагал принципиально отличающийся метод построения текста модуля, который нам представляется более глубоким и перспективным. Во-первых, вместе с уточнением операторов он предлагал постепенно (по шагам) уточнять (детализировать) и используемые структуры данных. Во-вторых, на каждом шаге он предлагал создавать некоторую виртуальную машину для детализации и в ее терминах производить детализацию всех уточняемых понятий, для которых эта машина позволяет это сделать. Таким образом, Дейкстра предлагал, по существу, детализировать по горизонтальным слоям, что является перенесением его идеи о слоистых системах (см. лекцию 6) на уровень разработки модуля. Такой метод разработки модуля поддерживается в настоящее время пакетами языка АДА и средствами объектно-ориентированного программирования.

Контроль программного модуля

Применяются следующие методы контроля программного модуля:

- статическая проверка текста модуля;
- сквозное прослеживание;
- доказательство свойств программного модуля.

При статической проверке текста модуля этот текст просматривается с начала до конца с целью найти ошибки в модуле. Обычно для такой проверки привлекают, кроме разработчика

модуля, еще одного или даже нескольких программистов. Рекомендуется ошибки, обнаруживаемые при такой проверке исправлять не сразу, а по завершению чтения текста модуля.

Сквозное прослеживание представляет собой один из видов динамического контроля модуля. В нем также участвуют несколько программистов, которые вручную прокручивают выполнение модуля (оператор за оператором в той последовательности, какая вытекает из логики работы модуля) на некотором наборе тестов.

Доказательству свойств программ посвящена следующая лекция. Здесь следует лишь отметить, что этот метод применяется пока очень редко.

Ход работы

Задание 1. Разработать модули будущей информационной системы. Оформить внешнюю спецификацию модулей. В спецификацию включить внешнее описание модуля, как подключается модуль, какие данные на входе/выходе модуля, структура модуля и средства защиты информации.

Задание 2. Составить в виде функциональной и (или) структурной схемы общий алгоритм работы ПО.

Практическая работа № 4.11. Настройка сетевого доступа

Цель работы: изучение настройки сетевого доступа

Ход работы

1. Создать у себя на компьютере, на диске Dпапку с названием группы.
2. Настроить к ней общий доступ с полными правами.
3. Организовать доступ к сетевым принтерам.
4. В ней создать текстовый файл со следующими характеристиками: имя файла – фамилия (или фамилии студентов, работающих за этим компьютером), содержимое – IPадрес компьютера, его имя в сети, имя рабочей группы, перечислить все компьютеры в этой рабочей группе, указать сетевое имя принтера и его спецификацию.
5. Передать свой файл по сети всем студентам на занятии.
6. Забрать такой же файл с компьютера справа, добавив к его имени знак «+».
7. Создать папку с ограниченными правами (только для чтения). Протестируйте свою папку с чужого компьютера на возможность записи в ней.
8. Построить схему ЛВС, которую вы исследовали.

Настройка сетевого доступа к дискам

Вы можете открыть пользователям локальной сети доступ к дискам вашего компьютера, что позволит им просматривать, редактировать и сохранять файлы на этих дисках, создавать и удалять папки, прослушивать хранящиеся на вашем компьютере аудиозаписи, устанавливать с вашего винчестера различные программы. Совместное использование дисковых ресурсов может быть необходимо, например, в случае, если только ваш компьютер во всей сети оснащен приводом CD-ROM или DVD.

Чтобы открыть пользователям локальной сети доступ к дисковым ресурсам вашего компьютера, необходимо проделать следующее:

1. откройте системное окно Мой компьютер;
2. щелкните правой кнопкой мыши на изображении диска, к которому вы хотите открыть доступ по сети, и выберите в появившемся меню пункт Свойства;
3. в открывшемся окне Свойства: локальный диск перейдите ко вкладке Доступ и выберите пункт Если вы хотите открыть доступ к корневой папке диска, щелкните здесь (для MS Windows XP), в другой операционной системе семейства Windows достаточно установить переключатель в положение Общий ресурс;
4. в разделе Сетевой совместный доступ и безопасность установите флажок рядом с пунктом Открыть общий доступ к этой папке и введите в поле Общий ресурс сетевое имя

своего диска — оно будет отображаться в папке Сетевое окружение других пользователей локальной сети (рис. 1);

5. если вы хотите открыть пользователям сети полный доступ к своему диску, то есть разрешить им создавать, удалять, перемещать и переименовывать файловые объекты на вашем винчестере, установите флажок рядом с пунктом Разрешить изменение файлов по сети. Если флажок сброшен, пользователи смогут обращаться к диску в режиме «только чтение»;
6. щелкните на кнопке ОК, чтобы сохранить внесенные вами изменения. Диск, к которому открыт доступ из локальной сети, будет показан в папке Мой компьютер с помощью специальной метки в виде изображения открытой ладони.

ПРИМЕЧАНИЕ

В целях безопасности не рекомендуется открывать доступ к диску или логическому дисковому разделу, на котором установлена Microsoft Windows. Кто-либо из пользователей локальной сети может случайно или намеренно внести изменения в системные файлы, в результате чего Windows придет в неработоспособное состояние.

Управление сетевым доступом к папкам

Открытие сетевого доступа к дискам и дисковым разделам является потенциально опасным для хранящихся на винчестере данных, поскольку пользователь локальной сети может случайно или намерено уничтожить, переименовать или изменить файлы, предназначенные только для вашего личного пользования. С точки зрения безопасности лучше открыть доступ не к диску в целом, а к одной дисковой директории, предназначенной для совместного использования в локальной сети. Вы можете назначить такой папке произвольное сетевое имя, например, аналогичное системному имени дискового раздела, благодаря чему пользователям будет казаться, что они работают непосредственно с диском вашего компьютера, в то время как доступ к каким-либо ресурсам за пределами данной директории будет для них закрыт. Чтобы настроить сетевой доступ к какой-либо папке на жестком диске компьютера, необходимо проделать описанные ниже шаги.

1. Перейдите на один из дисков своего компьютера и создайте папку с произвольным именем, которую вы хотите сделать доступной из локальной сети.
2. Щелкните на значке папки правой кнопкой мыши и в появившемся меню выберите пункт Свойства.
3. В открывшемся окне Свойства папки перейдите к вкладке Доступ.
4. В разделе Сетевой совместный доступ и безопасность установите флажок рядом с пунктом Открыть общий доступ к этой папке и введите в поле Сетевой ресурс сетевое имя вашей папки. Оно может совпадать с именем вашего диска, например C, D, E или F, либо быть произвольным, например, Netfolder. Папка, сетевое имя которой совпадает с именем одного из дисковых разделов, фактически может находится на любом диске. Например, папка с сетевым именем C может храниться на диске D. Локальное и сетевое имя папки могут быть различными.
5. Если вы хотите открыть пользователям сети полный доступ к данной папке, установите флажок рядом с пунктом Разрешить изменение файлов по сети. Если флажок сброшен, пользователи смогут обращаться к папке в режиме «только чтение».
6. Щелкните на кнопке О К, чтобы сохранить внесенные вами изменения. Папка, к которой открыт сетевой доступ, будет отображаться в окне Проводника с помощью специальной метки в виде изображения открытой ладони.
7. Управление доступом к локальному принтеру

Вы можете открыть пользователям локальной сети доступ к принтеру, подключенному к вашему компьютеру, чтобы они могли печатать свои документы по сети. Для этого:

1. перейдите в системную папку Принтеры и факсы, выполнив команды Пуск→
2. Панель управления →Принтеры и другое оборудование→Принтеры и факсы; а щелкните на значке установленного в вашей системе принтера правой кнопкой мыши и выберите в появившемся меню пункт Свойства;

3. перейдите к вкладке Доступ диалогового окна Свойства: Принтер, установите переключатель в положение Общий доступ к данному принтеру и введите в поле Сетевое имя произвольное сетевое имя принтера;
4. щелкните на кнопке ОК, чтобы сохранить внесенные изменения. Принтер, к которому открыт сетевой доступ, будет отображаться в окне Принтеры и факсы с помощью специальной метки в виде изображения открытой ладони.

Подключение сетевого принтера

Если принтер подключен не к вашему, а к другому компьютеру локальной сети, вы можете использовать его для распечатки своих документов. Для этого:

1. перейдите в системную папку Принтеры и факсы, выполнив команды Пуск → Панель управления → Принтеры и другое оборудование → Принтеры и факсы;
2. щелкните на пункте Установка принтера в командном меню Задачи печати;
3. в появившемся окне Мастера установки принтеров нажмите на кнопку Далее;
4. в следующем окне Мастера установки принтеров выберите пункт Сетевой принтер, подключенный к другому компьютеру и снова нажмите Далее;
5. в следующем окне установите переключатель в положение Обзор принтеров и щелкните на кнопке Далее;
6. в предложенном списке принтеров, доступных в локальной сети, выберите нужный и снова нажмите Далее (рис. 2);
7. если вы хотите сделать этот принтер используемым в вашей системе по умолчанию, установите в следующем окне переключатель в положение Да и щелкните на кнопке Далее;
8. настройка сетевого принтера завершена. Нажмите на кнопку Готово, чтобы покинуть окно Мастера установки принтеров. Теперь все документы, распечатываемые вами из приложений Windows, будут направляться на этот принтер.

Подключение сетевого диска

Некоторые программы MS Windows, работающие с файловыми ресурсами других сетевых компьютеров (например, сетевая версия бухгалтерского пакета «1С») требуют, чтобы физический диск или дисковый раздел удаленного компьютера был подключен к вашей системе как сетевой диск. Сетевые диски отображаются в системном окне Мой компьютер наравне с вашими локальными дисками, вы можете обращаться к ним и работать с их содержимым так же, как с содержимым собственного винчестера. Для того чтобы подключить к системе сетевой диск, необходимо выполнить следующие операции:

1. щелкните правой кнопкой мыши на расположенном на Рабочем столе Windows значке Мой компьютер и выберите в появившемся меню пункт Подключить сетевой диск. На экране появится окно одноименного Мастера подключения сетевого диска;
2. выберите в меню Диск символ, которым будет обозначаться подключаемый к вашей системе сетевой диск, затем щелкните на расположенной рядом кнопке Обзор;
3. в открывшемся окне Обзор папки выберите из списка доступный для совместного использования диск удаленного компьютера и нажмите кнопку ОК.
4. если вы хотите, чтобы соединение с данным сетевым диском автоматически восстанавливалось всякий раз при включении вашего компьютера, в окне Мастера подключения сетевого диска установите флажок рядом с функцией Восстанавливать при входе в систему. Щелкните на кнопке Готово.

Созданный вами сетевой диск будет обозначен в окне Мой компьютер выбранным вами символом и сетевым именем компьютера, которому фактически принадлежит. Например, сетевой диск E on Veronika (K:) является диском E подключенного к сети компьютера Veronika, но в вашей системе он обозначен символом K.

Чтобы отключить сетевой диск, щелкните на его изображении в окне Мой компьютер правой кнопкой мыши и в появившемся контекстном меню выберите пункт Отключить.

Вопросы по теме:

1. Каким образом внешний компьютер идентифицируется на вашем компьютере?

2. Дайте определение одноранговых локальных вычислительных сетей.
3. Как осуществить доступ к Вашим каталогам с другого ПК?
4. В каких случаях лучше использовать мастер настройки сети, а в каких лучше самостоятельно настроить

Практическая работа № 4.12. Тестирование программных продуктов

Цель работы: изучить способы тестирования программных продуктов

Теоретический материал

Основные понятия

Отладка ПС - это деятельность, направленная на обнаружение и исправление ошибок в ПС с использованием процессов выполнения его программ. *Тестирование* ПС - это процесс выполнения его программ на некотором наборе данных, для которого заранее известен результат применения или известны правила поведения этих программ. Указанный набор данных называется *тестовым* или просто *тестом*. Таким образом, отладку можно представить в виде многократного повторения трех процессов: тестирования, в результате которого может быть констатировано наличие в ПС ошибки, поиска места ошибки в программах и документации ПС и редактирования программ и документации с целью устранения обнаруженной ошибки. Другими словами,:

Отладка = Тестирование + Поиск ошибок + Редактирование.

В зарубежной литературе отладку часто понимают только как процесс поиска и исправления ошибок (без тестирования), факт наличия которых устанавливается при тестировании. Иногда тестирование и отладку считают синонимами. В нашей стране в понятие отладки обычно включают и тестирование, поэтому мы будем следовать сложившейся традиции. Впрочем, совместное рассмотрение в данной лекции этих процессов делает указанное разночтение не столь существенным. Следует, однако, отметить, что тестирование используется и как часть процесса аттестации ПС.

Принципы и виды отладки программного средства

Успех отладки ПС в значительной степени предопределяет рациональная организация тестирования. При отладке ПС отыскиваются и устраняются, в основном, те ошибки, наличие которых в ПС устанавливается при тестировании. Как было уже отмечено, тестирование не может доказать правильность ПС , в лучшем случае оно может продемонстрировать наличие в нем ошибки. Другими словами, нельзя гарантировать, что тестированием ПС практически выполнимым набором тестов можно установить наличие каждой имеющейся в ПС ошибки. Поэтому возникает две задачи. Первая задача: подготовить такой набор тестов и применить к ним ПС, чтобы обнаружить в нем по возможности большее число ошибок. Однако чем дольше продолжается процесс тестирования (и отладки в целом), тем большей становится стоимость ПС. Отсюда вторая задача: определить момент окончания отладки ПС (или отдельной его компоненты). Признаком возможности окончания отладки является полнота охвата пропущенными через ПС тестами (т.е. тестами, к которым применено ПС) множества различных ситуаций, возникающих при выполнении программ ПС, и относительно редкое проявление ошибок в ПС на последнем отрезке процесса тестирования. Последнее определяется в соответствии с требуемой степенью надежности ПС, указанной в спецификации его качества.

Для оптимизации набора тестов, т.е. для подготовки такого набора тестов, который позволял бы при заданном их числе (или при заданном интервале времени, отведенном на тестирование) обнаруживать большее число ошибок в ПС, необходимо, во-первых, заранее планировать этот набор и, во-вторых, использовать рациональную стратегию планирования (проектирования) тестов. Проектирование тестов можно начинать сразу же после завершения этапа внешнего описания ПС. Возможны разные подходы к выработке стратегии проектирования тестов, которые можно условно графически разместить между следующими двумя крайними подходами. Левый крайний подход заключается в том, что тесты проектируются только на основании изучения спецификаций ПС (внешнего описания, описания архитектуры и спецификации

модулей). Строение модулей при этом никак не учитывается, т.е. они рассматриваются как черные ящики. Фактически такой подход требует полного перебора всех наборов входных данных, так как в противном случае некоторые участки программ ПС могут не работать при пропуске любого теста, а это значит, что содержащиеся в них ошибки не будут проявляться. Однако тестирование ПС полным множеством наборов входных данных практически неосуществимо. Правый крайний подход заключается в том, что тесты проектируются на основании изучения текстов программ с целью протестировать все пути выполнения каждой программ ПС. Если принять во внимание наличие в программах циклов с переменным числом повторений, то различных путей выполнения программ ПС может оказаться также чрезвычайно много, так что их тестирование также будет практически неосуществимо.

Оптимальная стратегия проектирования тестов расположена внутри интервала между этими крайними подходами, но ближе к левому краю. Она включает проектирование значительной части тестов по спецификациям, но она требует также проектирования некоторых тестов и по текстам программ. При этом в первом случае эта стратегия базируется на принципах:

- на каждую используемую функцию или возможность - хотя бы один тест,
- на каждую область и на каждую границу изменения какой-либо входной величины - хотя бы один тест,
- на каждую особую (исключительную) ситуацию, указанную в спецификациях, - хотя бы один тест.

Во втором случае эта стратегия базируется на принципе: каждая команда каждой программы ПС должна проработать хотя бы на одном тесте.

Оптимальную стратегию проектирования тестов можно конкретизировать на основании следующего принципа: для каждого программного документа (включая тексты программ), входящего в состав ПС, должны проектироваться свои тесты с целью выявления в нем ошибок. Во всяком случае, этот принцип необходимо соблюдать в соответствии с определением ПС и содержанием понятия технологии программирования как технологии разработки надежных ПС. В связи с этим Майерс даже определяет разные виды тестирования в зависимости от вида программного документа, на основании которого строятся тесты. В нашей стране различаются два основных вида отладки (включая тестирование): автономную и комплексную отладку ПС. *Автономная* отладка ПС означает последовательное раздельное тестирование различных частей программ, входящих в ПС, с поиском и исправлением в них фиксируемых при тестировании ошибок. Она фактически включает отладку каждого программного модуля и отладку сопряжения модулей. *Комплексная* отладка означает тестирование ПС в целом с поиском и исправлением фиксируемых при тестировании ошибок во всех документах (включая тексты программ ПС), относящихся к ПС в целом. К таким документам относятся определение требований к ПС, спецификация качества ПС, функциональная спецификация ПС, описание архитектуры ПС и тексты программ ПС.

Правила отладки программного средства

В этом разделе даются общие рекомендации по организации отладки ПС. Но сначала следует отметить некоторый феномен, который подтверждает важность предупреждения ошибок на предыдущих этапах разработки: по мере роста числа обнаруженных и исправленных ошибок в ПС *растет* также относительная вероятность существования в нем необнаруженных ошибок. Это объясняется тем, что при росте числа ошибок, обнаруженных в ПС, уточняется и наше представление об общем числе допущенных в нем ошибок, а значит, в какой-то мере, и о числе необнаруженных еще ошибок.

Ниже приводятся рекомендации по организации отладки в форме.

1. Считайте тестирование ключевой задачей разработки ПС, поручайте его самым квалифицированным и одаренным программистам; нежелательно тестировать свою собственную программу.
2. Хорош тот тест, для которого высока вероятность обнаружить ошибку, а не тот, который демонстрирует правильную работу программы.
3. Готовьте тесты как для правильных, так и для неправильных данных.

4. Документируйте пропуск тестов через компьютер; детально изучайте результаты каждого теста; избегайте тестов, пропуск которых нельзя повторить.
5. Каждый модуль подключайте к программе только один раз; никогда не изменяйте программу, чтобы облегчить ее тестирование.
6. Пропускайте заново все тесты, связанные с проверкой работы какой-либо программы ПС или ее взаимодействия с другими программами, если в нее были внесены изменения (например, в результате устранения ошибки).

Автономная отладка программного средства

При автономной отладке ПС каждый модуль на самом деле тестируется в некотором программном окружении, кроме случая, когда отлаживаемая программа состоит только из одного модуля. Это окружение состоит из других модулей, часть которых является модулями отлаживаемой программы, которые уже отлажены, а часть - модулями, управляющими отладкой. Таким образом, при автономной отладке тестируется всегда некоторая программа (*тестируемая программа*), построенная специально для тестирования отлаживаемого модуля. Эта программа лишь частично совпадает с отлаживаемой программой, кроме случая, когда отлаживается последний модуль отлаживаемой программы. В процессе автономной отладки ПС производится наращивание тестируемой программы отлаженными модулями: при переходе к отладке следующего модуля в его программное окружение добавляется последний отлаженный модуль. Такой процесс наращивания программного окружения отлаженными модулями называется *интеграцией* программы. Отладочные модули, входящие в окружение отлаживаемого модуля, зависят от порядка, в каком отлаживаются модули этой программы, от того, какой модуль отлаживается и, возможно, от того, какой тест будет пропускаться.

При восходящем тестировании это окружение будет содержать только один отладочный модуль (кроме случая, когда отлаживается последний модуль отлаживаемой программы), который будет головным в тестируемой программе. Такой отладочный модуль называют *ведущим* (или драйвером). Ведущий отладочный модуль подготавливает информационную среду для тестирования отлаживаемого модуля (т. е. формирует ее состояние, требуемое для тестирования этого модуля, в частности, путем ввода некоторых тестовых данных), осуществляет обращение к отлаживаемому модулю и после окончания его работы выдает необходимые сообщения. При отладке одного модуля для разных тестов могут составляться разные ведущие отладочные модули.

При нисходящем тестировании окружение отлаживаемого модуля в качестве отладочных модулей содержит *отладочные имитаторы* (заглушки) некоторых еще не отлаженных модулей. К таким модулям относятся, прежде всего, все модули, к которым может обращаться отлаживаемый модуль, а также еще не отлаженные модули, к которым могут обращаться уже отлаженные модули (включенные в это окружение). Некоторые из этих имитаторов при отладке одного модуля могут изменяться для разных тестов.

На практике в окружении отлаживаемого модуля могут содержаться отладочные модули обоих типов, если используется смешанная стратегия тестирования. Это связано с тем, что как восходящее, так и нисходящее тестирование имеет свои достоинства и свои недостатки.

К *достоинствам восходящего тестирования* относятся:

- простота подготовки тестов,
- возможность полной реализации плана тестирования модуля.

Это связано с тем, что тестовое состояние информационной среды готовится непосредственно перед обращением к отлаживаемому модулю (ведущим отладочным модулем).

Недостатками восходящего тестирования являются следующие его особенности:

- тестовые данные готовятся, как правило, не в той форме, которая рассчитана на пользователя (кроме случая, когда отлаживается последний, головной, модуль отлаживаемой программ);
- большой объем отладочного программирования (при отладке одного модуля приходится составлять много ведущих отладочных модулей, формирующих подходящее состояние информационной среды для разных тестов);

– необходимость специального тестирования сопряжения модулей.

К *достоинствам нисходящего тестирования* относятся следующие его особенности:

– большинство тестов готовится в форме, рассчитанной на пользователя;
– во многих случаях относительно небольшой объем отладочного программирования (имитаторы модулей, как правило, весьма просты и каждый пригоден для большого числа, нередко - для всех, тестов);
– отпадает необходимость тестирования сопряжения модулей.

Недостатком нисходящего тестирования является то, что тестовое состояние информационной среды перед обращением к отлаживаемому модулю готовится косвенно - оно является результатом применения уже отлаженных модулей к тестовым данным или данным, выдаваемым имитаторами. Это, во-первых, затрудняет подготовку тестов и требует высокой квалификации тестовика (разработчика тестов), а во-вторых, делает затруднительным или даже невозможным реализацию полного плана тестирования отлаживаемого модуля. Указанный недостаток иногда вынуждает разработчиков применять восходящее тестирование даже в случае нисходящей разработки. Однако чаще применяют некоторые модификации нисходящего тестирования, либо некоторую комбинацию нисходящего и восходящего тестирования. Исходя из того, что нисходящее тестирование, в принципе, является предпочтительным, остановимся на приемах, позволяющих в какой-то мере преодолеть указанные трудности.

Прежде всего, необходимо организовать отладку программы таким образом, чтобы как можно раньше были отлажены модули, осуществляющие ввод данных, - тогда тестовые данные можно готовить в форме, рассчитанной на пользователя, что существенно упростит подготовку последующих тестов. Далеко не всегда этот ввод осуществляется в головном модуле, поэтому приходится в первую очередь отлаживать цепочки модулей, ведущие к модулям, осуществляющим указанный ввод. Пока модули, осуществляющие ввод данных, не отлажены, тестовые данные поставляются некоторыми имитаторами: они либо включаются в имитатор как его часть, либо вводятся этим имитатором.

При нисходящем тестировании некоторые состояния информационной среды, при которых требуется тестировать отлаживаемый модуль, могут не возникать при выполнении отлаживаемой программы ни при каких входных данных. В этих случаях можно было бы вообще не тестировать отлаживаемый модуль, так как обнаруживаемые при этом ошибки не будут проявляться при выполнении отлаживаемой программы ни при каких входных данных. Однако так поступать не рекомендуется, так как при изменениях отлаживаемой программы (например, при сопровождении ПС) не использованные для тестирования отлаживаемого модуля состояния информационной среды могут уже возникать, что требует дополнительного тестирования этого модуля (а этого при рациональной организации отладки можно было бы не делать, если сам данный модуль не изменялся). Для осуществления тестирования отлаживаемого модуля в указанных ситуациях иногда используют подходящие имитаторы, чтобы создать требуемое состояние информационной среды. Чаще же пользуются модифицированным вариантом нисходящего тестирования, при котором отлаживаемые модули перед их интеграцией предварительно тестируются отдельно (в этом случае в окружении отлаживаемого модуля появляется ведущий отладочный модуль, наряду с имитаторами модулей, к которым может обращаться отлаживаемый модуль). Однако, представляется более целесообразной другая модификация нисходящего тестирования: после завершения нисходящего тестирования отлаживаемого модуля для достижимых тестовых состояний информационной среды следует его отдельно протестировать для остальных требуемых состояний информационной среды.

Часто применяют также комбинацию восходящего и нисходящего тестирования, которую называют методом *сандвича* . Сущность этого метода заключается в одновременном осуществлении как восходящего, так и нисходящего тестирования, пока эти два процесса тестирования не встретятся на каком-либо модуле где-то в середине структуры отлаживаемой программы. Этот метод при разумном порядке тестирования позволяет воспользоваться достоинствами как восходящего, так и нисходящего тестирования, а также в значительной степени нейтрализовать их недостатки.

Весьма важным при автономной отладке является тестирование

сопряжения модулей. Дело в том, что спецификация каждого модуля программы, кроме головного, используется в этой программы в двух ситуациях: во-первых, при разработке текста (иногда говорят: тела) этого модуля и, во-вторых, при написании обращения к этому модулю в других модулях программы. И в том, и в другом случае в результате ошибки может быть нарушено требуемое соответствие заданной спецификации модуля. Такие ошибки требуется обнаруживать и устранять. Для этого и предназначено тестирование сопряжения модулей. При нисходящем тестировании тестирование сопряжения осуществляется попутно каждым пропускаемым тестом, что считают достоинством нисходящего тестирования. При восходящем тестировании обращение к отлаживаемому модулю производится не из модулей отлаживаемой программы, а из ведущего отладочного модуля. В связи с этим существует опасность, что последний модуль может приспособиться к некоторым "заблуждениям" отлаживаемого модуля. Поэтому, приступая (в процессе интеграции программы) к отладке нового модуля, приходится тестировать каждое обращение к ранее отлаженному модулю с целью обнаружения несогласованности этого обращения с телом соответствующего модуля (и не исключено, что виноват в этом ранее отлаженный модуль). Таким образом, приходится частично повторять в новых условиях тестирование ранее отлаженного модуля, при этом возникают те же трудности, что и при нисходящем тестировании.

Автономное тестирование модуля целесообразно осуществлять в четыре последовательно выполняемых шага.

Шаг 1. На основании спецификации отлаживаемого модуля подготовьте тесты для каждой возможности и каждой ситуации, для каждой границы областей допустимых значений всех входных данных, для каждой области изменения данных, для каждой области недопустимых значений всех входных данных и каждого недопустимого условия.

Шаг 2. Проверьте текст модуля, чтобы убедиться, что каждое направление любого разветвления будет пройдено хотя бы на одном тесте. Добавьте недостающие тесты.

Шаг 3. Проверьте текст модуля, чтобы убедиться, что для каждого цикла существуют тесты, обеспечивающие, по крайней мере, три следующие ситуации: тело цикла не выполняется ни разу, тело цикла выполняется один раз и тело цикла выполняется максимальное число раз. Добавьте недостающие тесты.

Шаг 4. Проверьте текст модуля, чтобы убедиться, что существуют тесты, проверяющие чувствительность к отдельным особым значениям входных данных. Добавьте недостающие тесты.

Комплексная отладка программного средства.

Как уже было сказано выше, при комплексной отладке тестируется ПС в целом, причем тесты готовятся по каждому из документов ПС. Тестирование этих документов производится, как правило, в порядке, обратном их разработке. Исключение составляет лишь тестирование документации по применению, которая разрабатывается по внешнему описанию параллельно с разработкой текстов программ - это тестирование лучше производить после завершения тестирования внешнего описания. Тестирование при комплексной отладке представляет собой применение ПС к конкретным данным, которые в принципе могут возникнуть у пользователя (в частности, все тесты готовятся в форме, рассчитанной на пользователя), но, возможно, в моделируемой (а не в реальной) среде. Например, некоторые недоступные при комплексной отладке устройства ввода и вывода могут быть заменены их программными имитаторами.

Тестирование архитектуры ПС. Целью тестирования является поиск несоответствия между описанием архитектуры и совокупностью программ ПС. К моменту начала тестирования архитектуры ПС должна быть уже закончена автономная отладка каждой подсистемы. Ошибки реализации архитектуры могут быть связаны, прежде всего, с взаимодействием этих подсистем, в частности, с реализацией архитектурных функций (если они есть). Поэтому хотелось бы проверить все пути взаимодействия между подсистемами ПС. При этом желательно хотя бы протестировать все цепочки выполнения подсистем без повторного вхождения последних. Если

заданная архитектура представляет ПС в качестве малой системы из выделенных подсистем, то число таких цепочек будет вполне обозримо.

Тестирование внешних функций. Целью тестирования является поиск расхождений между функциональной спецификацией и совокупностью программ ПС. Несмотря на то, что все эти программы автономно уже отлажены, указанные расхождения могут быть, например, из-за несоответствия внутренних спецификаций программ и их модулей (на основании которых производилось автономное тестирование) функциональной спецификации ПС. Как правило, тестирование внешних функций производится так же, как и тестирование модулей на первом шаге, т.е. как черного ящика.

Тестирование качества ПС. Целью тестирования является поиск нарушений требований качества, сформулированных в спецификации качества ПС. Это наиболее трудный и наименее изученный вид тестирования. Ясно лишь, что далеко не каждый примитив качества ПС может быть испытан тестированием (об оценке качества ПС см. лекцию 14). Завершенность ПС проверяется уже при тестировании внешних функций. На данном этапе тестирование этого примитива качества может быть продолжено, если требуется получить какую-либо вероятностную оценку степени надежности ПС. Однако, методика такого тестирования еще требует своей разработки. Могут тестироваться такие примитивы качества, как точность, устойчивость, защищенность, временная эффективность, в какой-то мере - эффективность по памяти, эффективность по устройствам, расширяемость и, частично, независимость от устройств. Каждый из этих видов тестирования имеет свою специфику и заслуживает отдельного рассмотрения. Мы здесь ограничимся лишь их перечислением. Легкость применения ПС (критерий качества, включающий несколько примитивов качества, см. лекцию 4) оценивается при тестировании документации по применению ПС.

Тестирование документации по применению ПС. Целью тестирования является поиск несогласованности документации по применению и совокупностью программ ПС, а также выявление неудобств, возникающих при применении ПС. Этот этап непосредственно предшествует подключению пользователя к завершению разработки ПС (тестированию определения требований к ПС и аттестации ПС), поэтому весьма важно разработчикам сначала самим воспользоваться ПС так, как это будет делать пользователь. Все тесты на этом этапе готовятся исключительно на основании только документации по применению ПС. Прежде всего, должны тестироваться возможности ПС как это делалось при тестировании внешних функций, но только на основании документации по применению. Должны быть протестированы все неясные места в документации, а также все примеры, использованные в документации. Далее тестируются наиболее трудные случаи применения ПС с целью обнаружить нарушение требований относительности легкости применения ПС.

Тестирование определения требований к ПС. Целью тестирования является выяснение, в какой мере ПС не соответствует предъявленному определению требований к нему. Особенность этого вида тестирования заключается в том, что его осуществляет организация-покупатель или организация-пользователь ПС как один из путей преодоления барьера между разработчиком и пользователем. Обычно это тестирование производится с помощью контрольных задач - типовых задач, для которых известен результат решения. В тех случаях, когда разрабатываемое ПС должно придти на смену другой версии ПС, которая решает хотя бы часть задач разрабатываемого ПС, тестирование производится путем решения общих задач с помощью как старого, так и нового ПС (с последующим сопоставлением полученных результатов). Иногда в качестве формы такого тестирования используют *опытную* эксплуатацию ПС - ограниченное применение нового ПС с анализом использования результатов в практической деятельности. По существу, этот вид тестирования во многом перекликается с испытанием ПС при его аттестации, но выполняется до аттестации, а иногда и вместо аттестации.

Ход работы

1. С помощью системы создания инсталляторов создайте из программы установочный файл.
2. Выполните тестирование удобства установки.

3. Выполните тестирование конфигурации оборудования.
4. Выполните тестирование восстановления.
5. Выполните тестирование удобства эксплуатации.
6. Результаты выполнения практического задания запишите в отчет.

Практическая работа № 4.13. Сравнение результатов тестирования с требованиями технического задания и/или спецификацией

Цель работы: описать и проанализировать информационную систему, распределить роли в группе разработчиков.

Ход работы

Процесс создания профессионального ПО всегда является субъектом бюджетной политики организации, где оно разрабатывается, и имеет временные ограничения. Работа руководителя программного проекта по большому счету заключается в том, чтобы гарантировать выполнение этих бюджетных и временных ограничений с учетом бизнес-целей организации относительно разрабатываемого ПО.

Руководители проектов призваны спланировать все этапы разработки программного продукта. Они также должны контролировать ход выполнения работ и соблюдения всех требуемых стандартов. Постоянный контроль за ходом выполнения работ необходим для того, чтобы процесс разработки не выходил за временные и бюджетные ограничения. Хорошее управление не гарантирует успешного завершения проекта, но плохое управление обязательно приведет к его провалу. Это может выразиться в задержке сроков сдачи готового ПО, в превышении сметной стоимости проекта и в несоответствии готового ПО спецификации требований.

Процесс разработки ПО существенно отличается от процессов реализации технических проектов, что порождает определенные сложности в управлении программными проектами:

Программный продукт нематериален. Программное обеспечение нематериально, его нельзя увидеть или потрогать. Руководитель программного проекта не видит процесс "роста" разрабатываемого ПО. Он может полагаться только на документацию, которая фиксирует процесс разработки программного продукта.

Не существует стандартных процессов разработки ПО. На сегодняшний день не существует четкой зависимости между процессом создания ПО и типом создаваемого программного продукта. Другие технические дисциплины имеют длительную историю, процессы разработки технических изделий многократно опробованы и проверены. Процессы создания большинства технических систем хорошо изучены. Изучением же процессов создания ПО специалисты занимаются только последнее время. Поэтому пока нельзя точно предсказать, на каком этапе процесса разработки ПО могут возникнуть проблемы, угрожающие всему программному проекту.

Большие программные проекты - это часто "одноразовые" проекты. Большие программные проекты, как правило, значительно отличаются от проектов, реализованных ранее. Поэтому, чтобы уменьшить неопределенность в планировании проекта, руководители проектов должны обладать очень большим практическим опытом. Но постоянные технологические изменения в компьютерной технике и коммуникационном оборудовании обесценивают предыдущий опыт. Знания и навыки, накопленные опытом, могут не востребоваться в новом проекте.

Перечисленные отличия могут привести к тому, что реализация проекта выйдет из временного графика или превысит бюджетные ассигнования. Программные системы зачастую оказываются новинками как в "идеологическом", так и в техническом плане. Поэтому, предвидя возможные проблемы в реализации программного проекта, следует всегда помнить, что многим из них свойственно выходить за рамки временных и бюджетных ограничений.

Процесс управления разработкой программного обеспечения

Невозможно описать и стандартизировать все работы, выполняемые в проекте по созданию ПО. Эти работы весьма существенно зависят от организации, где выполняется разработка ПО, и от типа создаваемого программного продукта. Но всегда можно выделить следующие:

– Написание предложений по созданию ПО.
– Планирование и составление графика работ по созданию ПО.
– Оценивание стоимости проекта.
– Подбор персонала.
– Контроль за ходом выполнения работ.
– Написание отчетов и представлений.

Первая стадия программного проекта может состоять из написания предложений по реализации этого проекта. Предложения должны содержать описание целей проектов и способов их достижения. Они также обычно включают в себя оценки финансовых и временных затрат на выполнение проекта. При необходимости здесь могут приводиться обоснования для передачи проекта на выполнение сторонней организации или команде разработчиков.

Написание предложений — очень ответственная работа, так как для многих организаций вопрос о том, будет ли проект выполняться самой организацией или разрабатываться по контракту сторонней компанией, является критическим. Не существует каких-либо рекомендаций по написанию предложений, многое здесь зависит от опыт.

На этапе планирования проекта определяются процессы, этапы и полученные на каждом из них результаты, которые должны привести к выполнению проекта. Реализация этого плана приведет к достижению целей проекта. Определение стоимости проекта напрямую связано с его планированием, поскольку здесь оцениваются ресурсы, требующиеся для выполнения плана.

Контроль за ходом выполнения работ (мониторинг проекта) — это непрерывный процесс, продолжающийся в течение всего срока реализации проекта. Руководитель должен постоянно отслеживать ход реализации проекта и сравнивать фактические и плановые показатели выполнения работ с их стоимостью. Хотя многие организации имеют механизмы формального мониторинга работ, опытный руководитель может составить ясную картину о стадии развитии проекта просто путем неформального общения с разработчиками.

Неформальный мониторинг часто помогает обнаружить потенциальные проблемы, которые в явном виде могут обнаружиться позднее. Например, ежедневное обсуждение хода выполнения работ может выявить отдельные недоработки в создаваемом программном продукте. Вместо ожидания отчетов, в которых будет отражен факт "пробуксовки" графика работ, можно обсудить со специалистами намечающиеся программистские проблемы и не допустить срыва графика работ.

В течение реализации проекта обычно происходит несколько формальных контрольных проверок хода выполнения работ по созданию ПО. Такие проверки должны дать общую картину хода реализации проекта в целом и показать, насколько уже разработанная часть ПО соответствует целям проекта.

Время выполнения больших программных проектов может занимать несколько лет. В течение этого времени цели и намерения организации, заказавшей программный проект, могут существенно измениться. Может оказаться, что разрабатываемый программный продукт стал уже ненужным либо исходные требования к создаваемому ПО просто устарели и их необходимо кардинально менять. В такой ситуации руководство организации-разработчика может принять решение о прекращении разработки ПО или об изменении проекта в целом с тем, чтобы учесть изменившиеся цели и намерения организации-заказчика.

Руководители проектов обычно обязаны сами подбирать исполнителей для своих проектов. В идеальном случае профессиональный уровень исполнителей должен соответствовать той работе, которую они будут выполнять в ходе реализации проекта. Однако во многих случаях руководители должны полагаться на команду разработчиков, которая далека от идеальной. Такая ситуация может быть вызвана следующими причинами:

1. Бюджет проекта не позволяет привлечь высококвалифицированный персонал. В таком случае за меньшую плату привлекаются менее квалифицированные специалисты.

2. Бывают ситуации, когда невозможно найти специалистов необходимой квалификации как в самой организации-разработчике, так и вне ее. Например, в организации "лучшие люди" могут быть уже заняты в других проектах.
3. Организация хочет повысить профессиональный уровень своих работников. В этом случае она может привлечь к участию в проекте неопытных или недостаточно квалифицированных работников, чтобы они приобрели необходимый опыт и поучились у более опытных специалистов.

Таким образом, почти всегда подбор специалистов для выполнения проекта имеет определенные ограничения и не является свободным. Вместе с тем необходимо, чтобы хотя бы несколько членов группы разработчиков имели квалификацию и опыт, достаточные для работы над данным проектом. В противном случае невозможно избежать ошибок в разработке ПО.

Руководитель проекта обычно обязан посылать отчеты о ходе его выполнения как заказчику, так и подрядным организациям. Это должны быть краткие документы, основанные на информации, извлекаемой из подробных' отчетов о проекте. В этих отчетах должна быть та информация, которая позволяет четко оценить степень готовности создаваемого программного продукта.

В рамках курса «Технология разработки программного обеспечения» выделены следующие роли в группе по разработке ПО:

- Руководитель – общее руководство проектом, написание документации, общение с заказчиком ПО
- Системный аналитик – разработка требований (составление технического задания, проекта программного обеспечения)
- Тестер – составление плана тестирования и аттестации готового ПО (продукта), составление сценария тестирования, базовый пример, проведение мероприятий по плану тестирования
- Разработчик – моделирование компонент программного обеспечения, кодирование

Планирование проекта разработки программного обеспечения

Эффективное управление программным проектом напрямую зависит от правильного планирования работ, необходимых для его выполнения. План помогает руководителю предвидеть проблемы, которые могут возникнуть на каких-либо этапах создания ПО, и разработать превентивные меры для их предупреждения или решения. План, разработанный на начальном этапе проекта, рассматривается всеми его участниками как руководящий документ, выполнение которого должно привести к успешному завершению проекта. Этот первоначальный план должен максимально подробно описывать все этапы реализации проекта.

Процесс планирования начинается, исходя из описания системы, с определения проектных ограничений (временные ограничения, возможности наличного персонала, бюджетные ограничения и т.д.). Эти ограничения должны определяться параллельно с оцениванием проектных параметров, таких как структура и размер проекта, а также распределением функций среди исполнителей. Затем определяются этапы разработки и то, какие результаты документация, прототипы, подсистемы или версии программного продукта) должны быть получены по окончании этих этапов. Далее начинается циклическая часть планирования. Сначала разрабатывается график работ по выполнению проекта или дается разрешение на продолжение использования ранее созданного графика. После этого проводится контроль выполнения работ и отмечаются расхождения между реальным и плановым ходом работ.

Далее, по мере поступления новой информации о ходе выполнения проекта, возможен пересмотр первоначальных оценок параметров проекта. Это, в свою очередь, может привести к изменению графика работ. Если в результате этих изменений нарушаются сроки завершения проекта, должны быть пересмотрены (и согласованы с заказчиком ПО) проектные ограничения. Конечно, большинство руководителей проектов не думают, что реализация их проектов пройдет гладко, без всяких проблем. Желательно описать возможные проблемы еще до того, как они проявят себя в ходе выполнения проекта. Поэтому лучше составлять "пессимистические" графики работ, чем "оптимистические". Но, конечно, невозможно построить план, учитывающий

все, в том числе случайные, проблемы и задержки выполнения проекта, поэтому и возникает необходимость периодического пересмотра проектных ограничений и этапов создания программного продукта.

План проекта должен четко показать ресурсы, необходимые для реализации проекта, разделение работ на этапы и временной график выполнения этих этапов. В некоторых организациях план проекта составляется как единый документ, содержащий все виды планов, описанных выше. В других случаях план проекта описывает только технологический процесс создания ПО. В таком плане обязательно присутствуют ссылки на планы других видов, но они разрабатываются отдельно от плана проекта.

Детализация планов проектов очень разнится в зависимости от типа разрабатываемого программного продукта и организации-разработчика. Но в любом случае большинство планов содержат следующие разделы.

1. Введение. Краткое описание целей проекта и проектных ограничений (бюджетных, временных и т.д.), которые важны для управления проектом.
2. Организация выполнения проекта. Описание способа подбора команды разработчиков и распределение обязанностей между членами команды.
3. Анализ рисков. Описание возможных проектных рисков, вероятности их проявления и стратегий, направленных на их уменьшение.
4. Аппаратные и программные ресурсы, необходимые для реализации проекта. Перечень аппаратных средств и программного обеспечения, необходимого для разработки программного продукта. Если аппаратные средства требуется закупать, приводится их стоимость совместно с графиком закупки и поставки.
5. Разбиение работ на этапы. Процесс реализации проекта разбивается на отдельные процессы, определяются этапы выполнения проекта, приводится описание результатов ("выходов") каждого этапа и контрольные отметки.
6. График работ. В этом графике отображаются зависимости между отдельными процессами (этапами) разработки ПО, оценки времени их выполнения и распределение членов команды разработчиков по отдельным этапам.
7. Механизмы мониторинга и контроля за ходом выполнения проекта. Описываются предоставляемые руководителем отчеты о ходе выполнения работ, сроки их предоставления, а также механизмы мониторинга всего проекта.

План должен регулярно пересматриваться в процессе реализации проекта. Одни части плана, например, график работ, изменяются часто, другие более стабильны. Для внесения изменений в план требуется специальная организация документопотока, позволяющая отслеживать эти изменения.

Ход работы

1. Составить подробное описание информационной системы.
2. На основании описания системы провести анализ осуществимости. В ходе анализа ответить на вопросы:

- Что произойдет с организацией, если система не будет введена в эксплуатацию?
- Какие текущие проблемы существуют в организации и как новая система поможет их решить?
- Каким образом система будет способствовать целям бизнеса?
- Требует ли разработка системы технологии, которая до этого не использовалась в организации?

Результатом анализа должно явиться заключение о возможности реализации проекта.

4. Распределить роли в группе (руководитель проекта-разработчик, системный аналитик-разработчик, тестер-разработчик).
5. Заполнить разделы плана:

- Введение
- Организация выполнения проекта
- Анализ рисков

Разделы должны содержать рекомендации относительно разработки системы, базовые предложения по объёму требуемого бюджета, числу разработчиков, времени и требуемому программному обеспечению.

Практическая работа № 4.14. Анализ рисков

Цель работы: анализ рисков информационной системы\

Теоретический материал

1 Идентификация рисков

1.1 Риск утери информации о клиентах

риск информационный потеря безубыточность

Так как внедрено программное обеспечение, в котором хранится вся информация о клиентах и данных компании, имеется риск потери этих данных, несанкционированный доступ к ним, вследствие чего возможна кража конфиденциальной информации о клиентах, а так же блокировка абонементов, что понесет существенные убытки компании.

Данный риск относится к информационным рискам. В результате воздействия этого события на обрабатываемую, хранящуюся или передаваемую информацию может произойти ее искажение, подделка, утечка, хищение, потеря, т.е. собственнику, владельцу информационных ресурсов может быть нанесен ущерб.

1.2 Риск отказа работы ПО на длительное время

Так же имеет место быть риск отказа работы ПО, что несет за собой временное отсутствие доступа ко всем данным, в том числе и бухгалтерским. В результате этого бухгалтер не сможет совершать никаких операций. Например, для бухгалтера будет недоступна подача данных в налоговую инспекцию, что повлечет за собой начисления штрафных санкции для организации, а так же может привести к лишению лицензии фитнес-клуба.

Так же, в результате воздействия этого риска, бухгалтер не сможет начислить сотрудникам заработную плату.

Нестабильные выплаты заработной платы портят репутацию компании и ведут к увольнению сотрудников.

Данный риск относится к производственным рискам, связанным с убытками от остановки производства или коммерческой деятельности либо предоставления услуг в сфере информатизации вследствие воздействия различных факторов, прежде всего, связанных с утратой или повреждением информации.

2 Оценка рисков

Оценка рисков производится по качественному и количественному методам.

Обычно считается, что риск тем больше, чем больше вероятность происшествия и тяжесть последствий, т.е. наблюдается зависимость от двух факторов: вероятности происшествия и тяжести возможных последствий.

2.1 Риск утери информации о клиентах

Сначала необходимо определить значения шкал. Значения субъективной шкалы вероятностей происшествия равно С – вероятность события – около 0,5.

Значения субъективной шкалы серьезности последствий равно Мо (Moderate – умеренный, средний) – происшествие с умеренными результатами: ликвидация последствий не связана с крупными затратами, воздействие на информационную технологию небольшое и не затрагивает критически важных задач. По матрице результатов оценивания рисков строится график.

Ход работы

Задание 1. Общая стоимость компании составляет 2 000 000 руб. Риск утери данных может нанести ущерб с фактором воздействия 50%. Вероятность происшествия – раз в 5 лет, т.е. 0,5.

Расчет цены потерь (Q) производится по формуле 1.

Q = K * C, (1)

K – Общая стоимость компании

C – фактор воздействия риска

Q = 2 000 000 * 0,5 = 1 000 000 рублей.

Величина риска (R) в данной ситуации зависит от двух факторов и считается по формуле

R = P * Q (2)

P – Вероятность наступления риска

R = 0,5 * 1 000 000 = 500 000 рублей, т.е. существует риск потери такой суммы ежегодно.

Задание 2. Риск потери информации в результате отказа работы ПО на длительное время

Значения субъективной шкалы вероятностей происшествия равно B – событие случается редко.

Значения субъективной шкалы серьезности последствий равно C (Critical – критический) – происшествие приводит к невозможности решения критически важных задач.

По матрице результатов оценивания рисков строится график

Общая стоимость компании составляет 2 000 000 руб. Риск утери данных может нанести ущерб с фактором воздействия 95%. Вероятность происшествия – раз в 10 лет, т.е. 0,1.

Расчет цены потерь производится по формуле 1.

Q = 2 000 000 * 0,95 = 1 900 000 рублей.

Величина риска в данной ситуации зависит от двух факторов и считается по формуле 2.

R = 0,1 * 1 900 000 = 190 000 рублей, т.е. существует риск потери такой суммы ежегодно.

Рекомендации по минимизации рисков

Планирование рисков подразумевает их уменьшение за счет принятия некоторых мер, например, грамотное управление паролями снижает риск несанкционированного доступа (НСД).

Для минимизации риска отказа ПО, необходима постоянная его техническая поддержка, которая производится компанией на аутсорсинге.

Если не удается уклониться от риска или эффективно его уменьшить, можно принять некоторые меры страховки, например, заключить договор с поставщиками ПО о сопровождении и компенсации ущерба, вызванного нештатными ситуациями.

Практическая работа № 4.15. Выявление первичных и вторичных ошибок

Цель работы: способы выявление первичных и вторичных ошибок.

Теоретический материал

Неоднократно экспериментально установлено, что в любом сложном комплексе программ в процессе эксплуатации обнаруживаются ошибки[Липаев_3], даже если проведено самое тщательное тестирование. Важной особенностью тестирования программ является невозможность получения полностью определенной абсолютно корректной программы, которую можно было бы использовать в качестве эталона без ошибок для сравнения с ней создаваемой программы. Поэтому при тестировании первоначально обнаруживаются вторичные ошибки, которые являются отклонениями некоторых результатов функционирования программ от предполагаемых эталонных значений. Тестирование для локализации ошибки позволяет установить причину вторичной ошибки и выявить первичную ошибку, подлежащую исправлению.

Первичные ошибки в программах в различной степени искажают результаты, которые первоначально обнаруживаются как вторичные ошибки, в процессе тестирования. Каждая первичная ошибка k-го типа отражается как вторичная ошибка j-го результата с некоторым коэффициентом пропорциональности Dkj. Если в некоторый момент тестирования имеющиеся первичные ошибки характеризуются вероятностями проявления Qk, то они будут приводить к интегральным вторичным ошибкам с интенсивностью (весом), рассчитываемой как

где m— полное число возможных типов ошибок в программах;

q— полное число контролируемых результатов, в которых могут проявляться вторичные ошибки.

Однако прямыми измерениями невозможно установить коэффициент влияния первичных ошибок на вторичные Dk j, а также вероятность первичных ошибок Qk. Поэтому исследованы, в основном, обобщенные характеристики вторичных ошибок и некоторые общие закономерности проявления первичных ошибок в процессе тестирования программ.

В результате анализа и обобщения экспериментальных данных предложено несколько математических моделей, описывающих основные закономерности изменения суммарного числа вторичных ошибок в программах. Эти модели предназначены для оценки: возможного изменения надежности функционирования в процессе отладки, испытаний и эксплуатации; числа ошибок, оставшихся не выявленными в тестируемых программах; времени, требующегося для обнаружения следующей ошибки в функционирующей или тестируемой программе; времени, необходимого для выявления всех ошибок с заданной вероятностью.

Точное определение полного числа не выявленных ошибок в ПО прямыми методами измерения невозможно, поскольку в противном случае их можно было бы все зафиксировать и устранить. Однако имеются косвенные пути для приближенной статистической оценки их полного числа или вероятности ошибки в каждой команде программы.

Такие оценки базируются на построении математических моделей в предположении жесткой корреляции между общим числом ошибок и их проявлениями в некотором ПО после его отладки в течение времени t, т.е. между следующими параметрами:

суммарным числом первичных ошибок в ПО (n0) или вероятностью ошибки в каждой команде программы (p0):

числом вторичных ошибок, выявляемых в единицу времени в процессе тестирования и отладки при постоянных усилиях на ее проведение (dn/dt);

интенсивностью искажений результатов в единицу времени (l) на выходе программы (вследствие невыявленных первичных ошибок) при функционировании системы в типовых условиях.

В результате может быть построена экспоненциальная математическая модель распределения ошибок в программах и установлена связь между интенсивностью обнаружения вторичных ошибок при тестировании dn/dt, интенсивностью l проявления ошибок при нормальном функционировании ПО и числом выявленных первичных ошибок n. При этом учитываются все виды ошибок независимо от источников их происхождения (технологические, программные, алгоритмические, системные).

При постоянных усилиях на тестирование интенсивность обнаружения искажений и вычислительного процесса, программ или данных вследствие еще невыявленных ошибок пропорциональна числу n0оставшихся первичных ошибок в ПО. Тогда dn/dt= K'l = Kn0=K(N0 – n),

где N0— число ошибок в ПО в начале отладки, а коэффициенты К и К' учитывают: масштаб времени, используемого для описания процесса обнаружения ошибок, быстродействие ЭВМ, распределение тестовых значений на входе проверяемого комплекса программ и другие параметры. Значение коэффициента К' можно, в принципе, определить как изменение темпа проявления ошибок при переходе от функционирования программ на специальных тестах к функционированию на нормальных типовых исходных данных. Так как предполагается, что в начале отладки при t= 0 отсутствуют обнаруженные ошибки, то n = N0 [1 – exp(-Kt)].

Число оставшихся первичных ошибок в ПО n0 = N0 exp(-Kt), пропорционально интенсивности обнаружения dn/dt с точностью до коэффициента К.

Длительность функционирования программ (наработка) между проявлением ошибок, которые рассматриваются как обнаруживаемые искажения программ, данных или вычислительного процесса, равна величине, обратной интенсивности обнаружения ошибок

Если известны все моменты обнаружения ошибок tii каждый раз в эти моменты обнаруживается и достоверно устраняется одна первичная ошибка, то, используя метод максимального правдоподобия, можно получить уравнение для определения значения начального числа первичных ошибок N0 а также выражение для расчета коэффициента пропорциональности

В результате можно рассчитать число оставшихся в программе первичных ошибок и среднюю наработку Т до обнаружения следующей ошибки. С помощью преобразований можно получить затраты времени на тестирование, которые позволяют устранить dn ошибок и соответственно повысить наработку между очередными обнаружениями ошибок от значения Т1 до Т2

Необходимо подчеркнуть статистический характер приведенных соотношений. Неравномерность выбора маршрутов исполнения программы при нормальной эксплуатации, разное влияние конкретных типов ошибок в программах на проявление их при функционировании, а также сравнительно небольшие значения n и dn, особенно на заключительных этапах отладки приводят к тому, что затраты времени на тестирование могут быть весьма значительными.

Ход работы

Математические модели надежности программных средств подразделяются на **аналитические** и **эмпирические.** Аналитические модели дают возможность рассчитать количественные показатели надежности, основываясь на данных о поведении программы в процессе тестирования. Эмпирические модели базируются на анализе структурных особенностей программ.

Аналитические модели, в свою очередь, делятся на две группы: **динамические** и **статические.** В динамических моделях поведение ПО, т. е. появление отказов, анализируется во времени. От способа фиксации момента отказа модели могут быть модели с непрерывным временем или модели с дискретным временем.

В **статических моделях** учитывают зависимость количества ошибок от числа тестовых прогонов.

К аналитическим динамическим дискретным моделям относятся, например, модели Шумана и Ла Падула.

Задание 1. Рассмотрим принципы построения модели Шумана подробнее. Введем основные показатели надежности. Определим вероятность безотказной работы как дополнительную функцию распределения: *Pit) = P{T > t}* где *T* — длительность наработки, *t* — текущее время.

1. Обозначим через *Q(t)* функцию распределения наработки до отказа невосстанавливаемых объектов *Q(t) - P{T < t).* Тогда *Q(t)* — **вероятность отказа** объекта за время *t.* Поскольку событие отказа и событие работоспособности являются дополнительными, вероятность безотказной работы *Pit)* может быть определена по формуле *P{t)* = 1 - *Q(t).* Плотность распределения *fit)* наработки до отказа определяется по формуле $f(t) = \frac{dQ(t)}{dt}.$

Интенсивность отказов — *Xit)* или условная плотность вероятности возникновения отказа невосстанавливаемого объекта, определяемая для рассматриваемого момента времени при условии, что до этого момента отказ не возникал. Интенсивность отказов определяется по формуле $\lambda(t) = \frac{f(t)}{P(t)}.$

Средняя наработка до отказа — математическое ожидание наработки до первого отказа может быть вычислена по формуле $T_o = MT = \int_0^\infty P(t)dt.$

2. Тогда, выразив вероятность безотказной работы через интенсивность отказов, получим:

$$P(t) = \exp\left(\int_0^t -\lambda(u)du\right).$$

В данной модели предполагается, что значение функции *X(t)* пропорционально числу ошибок, оставшихся в ПО после израсходованного на тестирование времени т:

$$\lambda(t) = C * \varepsilon_r(\tau), \qquad (7.1)$$

где *С* — некоторая константа; *t* — время работы ПО без отказа; е $_{г}$ (т) — удельное число ошибок на одну машинную команду, оставшихся в системе после времени тестирования т, т. е.:

$$\varepsilon_r(\tau) = \frac{E_\text{т}}{I_\text{т}} - \varepsilon_C(\tau), \qquad (7.2)$$

где E_m — число ошибок до начала тестирования; /$_{т}$ — общее число машинных команд, которое предполагается постоянным в рамках этапа тестирования; ?$_{с}$(т) — удельное количество обнаруженных ошибок на одну машинную команду в течение времени т. Тогда окончательно получаем:

$$P(t,\ \tau) = \exp\left[C * \left(\frac{E_\text{т}}{I_\text{т}} - \varepsilon_C(\tau)\right) * t\right]. \qquad (7.3)$$

3. Средняя наработка на отказ может быть вычислена по формуле

$$T_\text{ср} = \frac{1}{C * \left(\frac{E_\text{т}}{I_\text{т}} - \varepsilon_C(\tau)\right)}. \qquad (7.4)$$

4. Далее в процессе тестирования собирается информация по каждому прогону программы: время прогона и количество ошибок на прогоне. Общее время тестирования определяется как сумма всех времен прогонов. Также предполагается, что интенсивность появления ошибок постоянна и равна *X*. Тогда, например, при помощи функции максимального правдоподобия вычисляют неизвестные величины: начальное значение ошибок в ПО — E_m и коэффициент пропорциональности С.

К **аналитическим динамическим непрерывным** моделям относятся, например, модели Джелинского—Моранды, Шика—Волвертона, Мусса, переходных вероятностей (см., например.

Основное положение, на котором базируется модель Джелинского—Моранды, заключается в том, что в процессе тестирования ПО значение интервалов времени тестирования между обнаружением двух ошибок имеет экспоненциальное распределение с интенсивностью отказов, пропорциональной числу еще не выявленных ошибок. Проявление ошибок равновероятно, и их появление не зависит друг от друга. Каждая обнаруженная ошибка исправляется мгновенно, и в процессе ее исправления новой ошибки не вносится, т. е. число оставшихся ошибок уменьшается на единицу. Очевидно, что при этом интенсивность отказов в интервале между двумя соседними моментами проявления ошибок постоянна.

При данных предположениях функция плотности распределения времени обнаружения /-й ошибки, отсчитываемого от момента выявления (/ - 1)-й ошибки, имеет вид:

$$P(t_i) = \exp(-\lambda_i t_i), \qquad (7.5)$$

где *Xj* — интенсивность отказов, которая пропорциональна числу еще не выявленных ошибок в программе:

$$\lambda_i = C * (N - i + 1), \qquad (7.6)$$

где N — число ошибок, первоначально присутствующих в программе; С — коэффициент пропорциональности.

5. Наиболее вероятные значения величин N *и* C определяются на основе данных, полученных при тестировании. Для этого фиксируют время выполнения программы до очередного отказа At_1, At_2, ... Затем на основе методики максимума правдоподобия значение N получают как решение нелинейного уравнения:

$$K \frac{\sum_{i=1}^{K} \Delta t_i}{\sum_{i=1}^{K} \frac{1}{N+1-i}} = \sum_{i=1}^{K} (N+1-i)\Delta t_i, \qquad (7.7)$$

где K — число экспериментально полученных интервалов между отказами.

Реально значение N получают методом подбора, основываясь на том, что это целое число.

Значение коэффициента пропорциональности С получают по формуле

$$C = \frac{\sum_{i=1}^{K} \left(\frac{1}{N+1-i} \right)}{\sum_{i=1}^{K} \Delta t_i}. \qquad (7.8)$$

Данная методика работает для $K > 2$, т. е. надо иметь хотя бы два экспериментально полученных интервала между моментами возникновения ошибок.

6. Достаточно часто для аналогичных оценок используется **простая экспоненциальная модель,** которая отличается от рассмотренной выше модели Джелинского—Моранды тем, что интенсивность отказов X между соседними моментами проявления ошибок не постоянна, а имеет вид:

$$\lambda(t) = K(N - n(t)), \qquad (7.9)$$

где K — коэффициент пропорциональности; N — число первоначально содержащихся в программе ошибок; $n(t)$ — число обнаруженных к моменту времени t ошибок.

Следует заметить, что время t соответствует длительности исполнения программ на ЭВМ и не учитывает время, необходимое для анализа результатов и корректировки программ, поскольку, как и ранее, считается, что ошибки исправляются мгновенно и при их исправлении новые не вносятся. Коэффициент K в данной формуле (как и в предыдущем случае) вычисляется на основе статистических данных.

Также достаточно часто применяется модель **Шика—Волвертона** [34] — модификация модели Джелинского—Моранды для случая возникновения более одной ошибки на каждом промежутке времени.

7. К **аналитическим статическим** моделям относятся, например, модели Миллса, Липова, простая интуитивная модель, модель Коркорэ- на, модель Нельсона.

Для использования **модели Миллса** характерно внесение некоторого количества искусственных ошибок перед началом тестирования. Ошибки вносятся случайным образом и фиксируются в протоколе искусственных ошибок. Специалист, проводящий тестирование, не знает ни количества, ни характера внесенных ошибок до момента оценки показателей надежности по модели Миллса. Предполагается, что все ошибки (как естественные, так и искусственно внесенные) имеют равную вероятность быть найденными в процессе тестирования.

8. Тестируя программу в течение некоторого времени, собирают статистику об ошибках. В момент оценки надежности по протоколу искусственных ошибок все ошибки делятся на собственные и искусственные. Соотношение, называемое формулой Миллса, дает возможность оценить первоначальное число ошибок в программе:

$$N = \frac{S \cdot n}{V},$$

где N — первоначальное количество ошибок в программе; S — количество искусственно внесенных ошибок; n — число найденных собственных ошибок; V — число обнаруженных к моменту оценки искусственных ошибок.

Модель Липова [29] является модификацией модели Миллса. Она основана на анализе вероятности обнаружения ошибок при использовании различного числа тестов.

9. Использование **простой интуитивной модели** предполагает проведение тестирования двумя группами программистов, использующими независимые тестовые наборы, независимо одна от другой. В процессе тестирования каждая из групп фиксирует все найденные ею ошибки. При оценке числа оставшихся в программе ошибок результаты тестирования обеих групп собираются и сравниваются. Пусть первая группа обнаружила N_x ошибок, вторая — N_2, а N_n — это ошибки, обнаруженные обеими группами, т. е. дважды. Если обозначить через N общее количество ошибок, присутствовавших в программе до начала тестирования, то можно эффективность тестирования каждой из /7 ^{N}l Γ $^{N}2$ X групп определить как: $E, = $ —-; $E_2 = $ —-. Тогда, предполагая, что N N возможность обнаружения всех ошибок одинакова для обеих групп, можно оценить

$$N = \frac{N_1 \cdot N_2}{N_{12}}.$$

Предыдущие статические модели надежности основывались на анализе области ошибок. Модель Нельсона основывается на анализе области данных.

10. **Модель Нельсона** при расчете надежности ПО учитывает вероятность выбора определенного тестового набора для очередного выполнения программы. Предполагается, что область данных, необходимых для выполнения тестирования программного средства, разделяется на $к$ взаимоисключающих подобластей Z/, / = 1,2,..., $к$. Пусть Pi — вероятность того, что набор данных Zi будет выбран для очередного выполнения программы. Если к моменту оценки надежности было выполнено Ni прогонов программы на Zi наборе данных и из них прогонов закончились отказом, то надежность ПО определяется равенством:

$$P = 1 - \sum_{i=1}^{k} \frac{n_i}{N_i} P_i$$

Что касается **эмпирических моделей,** то они, как правило, основываются на сложности ПО и характеризуются размером программного обеспечения: количеством программных модулей, количеством и сложностью межмодульных интерфейсов. Под программным модулем в данном случае понимают программную единицу, выполняющую определенную функцию и взаимосвязанную с другими модулями ПО. Существуют также модели для объектно-ориентированного подхода (см., например, [33]).

Существует несколько разновидностей модели сложности, основанных на метриках сложности ПО. В каждой из них определяется некоторая оценка сложности программы, которая считается пропорциональной ее надежности.

Практическая работа № 4.16. Обнаружение вируса и устранение последствий его влияния

Цель работы: способы обнаружения вирусов и устранение последствий их влияния

Теоретический материал

Классификация компьютерных вирусов

В настоящее время в мире насчитывается более 40 тысяч только зарегистрированных компьютерных вирусов. Так как подавляющее большинство современных вредительских программ обладают способностью к саморазмножению, то часто их относят к компьютерным вирусам. Все компьютерные вирусы могут быть классифицированы по следующим признакам:

- **по среде обитания**;
- **по способу заражения**;
- **по степени опасности деструктивных (вредительских) воздействий**;
- **по алгоритму функционирования**.

По **среде обитания** компьютерные вирусы делятся на:

- сетевые;
- файловые;
- загрузочные;
- комбинированные.

Средой обитания ***сетевых*** вирусов являются элементы компьютерных сетей. ***Файловые*** вирусы размещаются в исполняемых файлах. ***Загрузочные*** вирусы находятся в загрузочных секторах (областях) внешних запоминающих устройств (boot-секторах). Иногда загрузочные вирусы называют бутовыми. ***Комбинированные*** вирусы размещаются в нескольких средах обитания. Примером таких вирусов служат загрузочно-файловые вирусы. Эти вирусы могут размещаться как в загрузочных секторах накопителей на магнитных дисках, так и в теле загрузочных файлов.

По **способу заражения среды обитания** компьютерные вирусы делятся на:

- резидентные;
- нерезидентные.

Резидентные вирусы после их активизации полностью или частично перемещаются из среды обитания (сеть, загрузочный сектор, файл) в оперативную память ЭВМ. Эти вирусы, используя, как правило, привилегированные режимы работы, разрешенные только операционной системе, заражают среду обитания и при выполнении определенных условий реализуют деструктивную функцию. В отличие от резидентных ***нерезидентные*** вирусы попадают в оперативную память ЭВМ только на время их активности, в течение которого выполняют деструктивную функцию и функцию заражения. Затем вирусы полностью покидают оперативную память, оставаясь в среде обитания. Если вирус помещает в оперативную память программу, которая не заражает среду обитания, то такой вирус считается нерезидентным.

Арсенал деструктивных или вредительских возможностей компьютерных вирусов весьма обширен. Деструктивные возможности вирусов зависят от целей и квалификации их создателя, а также от особенностей компьютерных систем.

По **степени опасности для информационных ресурсов** пользователя компьютерные вирусы можно разделить на:

- *безвредные вирусы*;
- *опасные вирусы*;
- *очень опасные вирусы*.

Безвредные компьютерные вирусы создаются авторами, которые не ставят себе цели нанести какой-либо ущерб ресурсам КС. Ими, как правило, движет желание показать свои возможности программиста. Другими словами, создание компьютерных вирусов для таких людей - своеобразная попытка самоутверждения. Деструктивное воздействие таких вирусов сводится к выводу на экран монитора невинных текстов и картинок, исполнению музыкальных фрагментов и т. п.

Однако при всей кажущейся безобидности таких вирусов они наносят определенный ущерб КС. Во-первых, такие вирусы расходуют ресурсы КС, в той или иной мере снижая ее

эффективность функционирования. Во-вторых, компьютерные вирусы могут содержать ошибки, вызывающие опасные последствия для информационных ресурсов КС. Кроме того, при модернизации операционной системы или аппаратных средств КС вирусы, созданные ранее, могут приводить к нарушениям штатного алгоритма работы системы.

К ***опасным*** относятся вирусы, которые вызывают существенное снижение эффективности КС, но не приводящие к нарушению целостности и конфиденциальности информации, хранящейся в запоминающих устройствах. Последствия таких вирусов могут быть ликвидированы без особых затрат материальных и временных ресурсов. Примерами таких вирусов являются вирусы, занимающие память ЭВМ и каналы связи, но не блокирующие работу сети; вирусы, вызывающие необходимость повторного выполнения программ, перезагрузки операционной системы или повторной передачи данных по каналам связи и т. п.

Очень опасными следует считать вирусы, вызывающие нарушение конфиденциальности, уничтожение, необратимую модификацию (в том числе и шифрование) информации, а также вирусы, блокирующие доступ к информации, приводящие к отказу аппаратных средств и наносящие ущерб здоровью пользователям. Такие вирусы стирают отдельные файлы, системные области памяти, форматируют диски, получают несанкционированный доступ к информации, шифруют данные и т. п.

Известны публикации, в которых упоминаются вирусы, вызывающие неисправности аппаратных средств. Предполагается, что на резонансной частоте движущиеся части электромеханических устройств, например в системе позиционирования накопителя на магнитных дисках, могут быть разрушены. Именно такой режим и может быть создан с помощью программы-вируса. Другие авторы утверждают, что возможно задание режимов интенсивного использования отдельных электронных схем (например, больших интегральных схем), при которых наступает их перегрев и выход из строя.

Использование в современных ПЭВМ постоянной памяти с возможностью перезаписи привело к появлению вирусов, изменяющих программы BIOS, что приводит к необходимости замены постоянных запоминающих устройств.

Возможны также воздействия на психику человека - оператора ЭВМ с помощью подбора видеоизображения, выдаваемого на экран монитора с определенной частотой (каждый двадцать пятый кадр). Встроенные кадры этой видеоинформации воспринимаются человеком на подсознательном уровне. В результате такого воздействия возможно нанесение серьезного ущерба психике человека. В 1997 году 700 японцев попали в больницу с признаками эпилепсии после просмотра компьютерного мультфильма по телевидению. Предполагают, что именно таким образом была опробована возможность воздействия на человека с помощью встраивания 25-го кадра.

В соответствии с **особенностями алгоритма функционирования** вирусы можно разделить на два класса:

- вирусы, не изменяющие среду обитания (файлы и секторы) при распространении;
- вирусы, изменяющие среду обитания при распространении.

В свою очередь, вирусы, ***не изменяющие среду обитания***, могут быть разделены на две группы:

- *вирусы-"спутники" (compаnion)*;
- *вирусы-"черви" (worm)*.

Вирусы-"спутники" не изменяют файлы. Механизм их действия состоит в создании копий исполняемых файлов. Например, в MS-DOS такие вирусы создают копии для файлов, имеющих расширение .EXE. Копии присваивается то же имя, что и исполняемому файлу, но расширение изменяется на .COM. При запуске файла с общим именем операционная система первым загружает на выполнение файл с расширением .COM, который является программой-вирусом. Файл-вирус запускает затем и файл с расширением .EXE.

Вирусы-"черви" попадают в рабочую станцию из сети, вычисляют адреса рассылки вируса по другим абонентам сети и осуществляют передачу вируса. Вирус не изменяет файлов и

не записывается в загрузочные секторы дисков. Некоторые вирусы-"черви" создают рабочие копии вируса на диске, другие - размещаются только в оперативной памяти ЭВМ.

По сложности, степени совершенства и особенностям маскировки алгоритмов вирусы, ***изменяющие среду обитания***, делятся на:

- *студенческие*;
- *"стелс" - вирусы (вирусы-невидимки)*;
- *полиморфные*.

К *студенческим* относят вирусы, создатели которых имеют низкую квалификацию. Такие вирусы, как правило, являются нерезидентными, часто содержат ошибки, довольно просто обнаруживаются и удаляются.

"Стелс"-вирусы и полиморфные вирусы создаются квалифицированными специалистами, хорошо знающими принцип работы аппаратных средств и операционной системы, а также владеющими навыками работы с машиноориентированными системами программирования.

"Стелс"-вирусы маскируют свое присутствие в среде обитания путем перехвата обращений операционной системы к пораженным файлам, секторам и переадресуют ОС к незараженным участкам информации. Вирус является резидентным, маскируется под программы ОС, может перемещаться в памяти. Такие вирусы активизируются при возникновении прерываний, выполняют определенные действия, в том числе и по маскировке, и только затем управление передается на программы ОС, обрабатывающие эти прерывания. "Стелс"-вирусы обладают способностью противодействовать резидентным антивирусным средствам.

Полиморфные вирусы не имеют постоянных опознавательных групп - сигнатур. Обычные вирусы для распознавания факта заражения среды обитания размещают в зараженном объекте специальную опознавательную двоичную последовательность или последовательность символов (сигнатуру), которая однозначно идентифицирует зараженность файла или сектора. Сигнатуры используются на этапе распространения вирусов для того, чтобы избежать многократного заражения одних и тех же объектов, так как при многократном заражении объекта значительно возрастает вероятность обнаружения вируса. Для устранения демаскирующих признаков полиморфные вирусы используют шифрование тела вируса и модификацию программы шифрования. За счет такого преобразования полиморфные вирусы не имеют совпадений кодов.

Любой вирус, независимо от принадлежности к определенным классам, должен иметь три функциональных блока: блок заражения (распространения), блок маскирования и блок выполнения деструктивных действий. Разделение на функциональные блоки означает, что к определенному блоку относятся команды программы вируса, выполняющие одну из трех функций, независимо от места нахождения команд в теле вируса.

После передачи управления вирусу, как правило, выполняются определенные функции блока маскировки. Например, осуществляется расшифрование тела вируса. Затем вирус осуществляет функцию внедрения в незараженную среду обитания. Если вирусом должны выполняться деструктивные воздействия, то они выполняются либо безусловно, либо при выполнении определенных условий.

Завершает работу вируса всегда блок маскирования. При этом выполняются, например, следующие действия: шифрование вируса (если функция шифрования реализована), восстановление старой даты изменения файла, восстановление атрибутов файла, корректировка таблиц ОС и др.

Последней командой вируса выполняется команда перехода на выполнение зараженных файлов или на выполнение программ ОС.

Для удобства работы с известными вирусами используются каталоги вирусов. В каталог помещаются следующие сведения о стандартных свойствах вируса: имя, длина, заражаемые файлы, место внедрения в файл, метод заражения, способ внедрения в ОП для резидентных

вирусов, вызываемые эффекты, наличие (отсутствие) деструктивной функции и ошибки. Наличие каталогов позволяет при описании вирусов указывать только особые свойства, опуская стандартные свойства и действия.

Известны следующие методы обнаружения вирусов:

- сканирование;
- обнаружение изменений;
- эвристический анализ;
- использование резидентных сторожей;
- вакцинирование программ;
- аппаратно-программная защита от вирусов.

Сканирование - один из самых простых методов обнаружения вирусов. Сканирование осуществляется программой-сканером, которая просматривает файлы в поисках опознавательной части вируса - сигнатуры. Программа фиксирует наличие уже известных вирусов, за исключением полиморфных вирусов, которые применяют шифрование тела вируса, изменяя при этом каждый раз и сигнатуру. Программы-сканеры могут хранить не сигнатуры известных вирусов, а их контрольные суммы. Программы-сканеры часто могут удалять обнаруженные вирусы. Такие программы называются полифагами.

Метод сканирования применим для обнаружения вирусов, сигнатуры которых уже выделены и являются постоянными. Для эффективного использования метода необходимо регулярное обновление сведений о новых вирусах.

Самой известной программой-сканером в России является Aidstest Дмитрия Лозинского.

Метод обнаружения изменений базируется на использовании программ-ревизоров. Эти программы определяют и запоминают характеристики всех областей на дисках, в которых обычно размещаются вирусы. При периодическом выполнении программ- ревизоров сравниваются хранящиеся характеристики и характеристики, получаемые при контроле областей дисков. По результатам ревизии программа выдает сведения о предположительном наличии вирусов.

Обычно программы-ревизоры запоминают в специальных файлах образы главной загрузочной записи, загрузочных секторов логических дисков, характеристики всех контролируемых файлов, каталогов и номера дефектных кластеров. Могут контролироваться также объем установленной оперативной памяти, количество подключенных к компьютеру дисков и их параметры.

Главным достоинством метода является возможность обнаружения вирусов всех типов, а также новых неизвестных вирусов. Совершенные программы-ревизоры обнаруживают даже "стелс"-вирусы. Например, программа-ревизор Adinf, разработанная Д. Ю Мостовым, работает с диском непосредственно по секторам через Bios. Это не позволяет использовать "стелс"-вирусам возможность перехвата прерываний и "подставки" для контроля нужной вирусу области памяти.

Имеются у этого метода и недостатки. С помощью программ-ревизоров невозможно определить вирус в файлах, которые поступают в систему уже зараженными. Вирусы будут обнаружены только после размножения в системе.

Программы-ревизоры непригодны для обнаружения заражения макровирусами, так как документы и таблицы очень часто изменяются.

Эвристический анализ сравнительно недавно начал использоваться для обнаружения вирусов. Как и метод обнаружения изменений, данный метод позволяет определять неизвестные вирусы, но не требует предварительного сбора, обработки и хранения информации о файловой системе.

Сущность эвристического анализа заключается в проверке возможных сред обитания вирусов и выявление в них команд (групп команд), характерных для вирусов. Такими командами могут быть команды создания резидентных модулей в оперативной памяти, команды прямого обращения к дискам, минуя ОС. Эвристические анализаторы при обнаружении "подозри-

тельных" команд в файлах или загрузочных секторах выдают сообщение о возможном заражении. После получения таких сообщений необходимо тщательно проверить предположительно зараженные файлы и загрузочные сектора всеми имеющимися антивирусными средствами. Эвристический анализатор имеется, например, в антивирусной программе Doctor Web.

Метод использования *резидентных сторожей* основан на применении программ, которые постоянно находятся в ОП ЭВМ и отслеживают все действия остальных программ.

В случае выполнения какой-либо программой подозрительных действий (обращение для записи в загрузочные сектора, помещение в ОП резидентных модулей, попытки перехвата прерываний и т. п.) резидентный сторож выдает сообщение пользователю. Программа-сторож может загружать на выполнение другие антивирусные программы для проверки "подозрительных" программ, а также для контроля всех поступающих извне файлов (со сменных дисков, по сети).

Существенным недостатком данного метода является значительный процент ложных тревог, что мешает работе пользователя, вызывает раздражение и желание отказаться от использования резидентных сторожей. Примером резидентного сторожа может служить программа Vsafe, входящая в состав MS DOS.

Под вакцинацией программ понимается создание специального модуля для контроля ее целостности. В качестве характеристики целостности файла обычно используется контрольная сумма. При заражении вакцинированного файла, модуль контроля обнаруживает изменение контрольной суммы и сообщает об этом пользователю. Метод позволяет обнаруживать все вирусы, в том числе и незнакомые, за исключением "стеле"- вирусов.

Самым надежным методом защиты от вирусов является использование *аппаратно-программных антивирусных средств*. В настоящее время для защиты ПЭВМ используются специальные контроллеры и их программное обеспечение. Контроллер устанавливается в разъем расширения и имеет доступ к общей шине. Это позволяет ему контролировать все обращения к дисковой системе. В программном обеспечении контроллера запоминаются области на дисках, изменение которых в обычных режимах работы не допускается. Таким образом, можно установить защиту на изменение главной загрузочной записи, загрузочных секторов, файлов конфигурации, исполняемых файлов и др.

При выполнении запретных действий любой программой контроллер выдает соответствующее сообщение пользователю и блокирует работу ПЭВМ.

Аппаратно-программные антивирусные средства обладают рядом достоинств перед программными:

- работают постоянно;
- обнаруживают все вирусы, независимо от механизма их действия;
- блокируют неразрешенные действия, являющиеся результатом работы вируса или неквалифицированного пользователя.

Недостаток у этих средств один - зависимость от аппаратных средств ПЭВМ. Изменение последних ведет к необходимости замены контроллера.

Примером аппаратно-программной защиты от вирусов может служить комплекс Sheriff.

Ход работы

Порядок действий пользователя при обнаружении заражения ЭВМ вирусами

Даже при скрупулезном выполнении всех правил профилактики возможность заражения ЭВМ компьютерными вирусами полностью исключить нельзя. И если вирус все же попал в КС, то последствия его пребывания можно свести к минимуму, придерживаясь определенной последовательности действий.

О наличии вируса в КС пользователь может судить по следующим событиям:

- появление сообщений антивирусных средств о заражении или о предполагаемом заражении;
- явные проявления присутствия вируса, такие как сообщения, выдаваемые на монитор или принтер, звуковые эффекты, уничтожение файлов и другие аналогичные действия, однозначно указывающие на наличие вируса в КС;

- неявные проявления заражения, которые могут быть вызваны и другими причинами, например, сбоями или отказами аппаратных и программных средств КС.

К неявным проявлениям наличия вирусов в КС можно отнести "зависания" системы, замедление выполнения определенных действий, нарушение адресации, сбои устройств и тому подобное.

Получив информацию о предполагаемом заражении, пользователь должен убедиться в этом. Решить такую задачу можно с помощью всего комплекса антивирусных средств. Убедившись в том, что заражение произошло, пользователю следует выполнить следующую последовательность шагов:

Шаг 1. Выключить ЭВМ для уничтожения резидентных вирусов.

Шаг 2. Осуществить загрузку эталонной операционной системы со сменного носителя информации, в которой отсутствуют вирусы.

Шаг 3. Сохранить на сменных носителях информации важные для вас файлы, которые не имеют резервных копий.

Шаг 4. Использовать антивирусные средства для удаления вирусов и восстановления файлов, областей памяти. Если работоспособность ЭВМ восстановлена, то осуществляется переход к шагу 8, иначе - к шагу 5.

Шаг 5. Осуществить полное стирание и разметку (форматирование) несъемных внешних запоминающих устройств. В ПЭВМ для этого могут быть использованы программы MS-DOS FDISK и FORMAT. Программа форматирования FORMAT не удаляет главную загрузочную запись на жестком диске, в которой может находиться загрузочный вирус . Поэтому необходимо выполнить программу FDISK с недокументированным параметром MBR, создать с помощью этой же программы разделы и логические диски на жестком диске. Затем выполняется программа FORMAT для всех логических дисков.

Шаг 6. Восстановить ОС, другие программные системы и файлы с дистрибутивов и резервных копий, созданных до заражения.

Шаг 7. Тщательно проверить файлы, сохраненные после обнаружения заражения, и, при необходимости, удалить вирусы и восстановить файлы;

Шаг 8. Завершить восстановление информации всесторонней проверкой ЭВМ с помощью всех имеющихся в распоряжении пользователя антивирусных средств.

При выполнении рекомендаций по профилактике заражения компьютерными вирусами, а также при умелых и своевременных действиях в случае заражения .вирусами, ущерб информационным ресурсам КС может быть сведен к минимуму.

В процессе удаления последствий заражения вирусами осуществляется удаление вирусов, а также восстановление файлов и областей памяти, в которых находился вирус. Существует два метода удаления последствий воздействия вирусов антивирусными программами.

Первый метод предполагает восстановление системы после воздействия известных вирусов. Разработчик программы-фага, удаляющей вирус, должен знать структуру вируса и его характеристики размещения в среде обитания.

Второй метод позволяет восстанавливать файлы и загрузочные сектора, зараженные неизвестными вирусами. Для восстановления файлов программа восстановления должна заблаговременно создать и хранить информацию о файлах, полученную в условиях отсутствия вирусов. Имея информацию о незараженном файле и используя сведения об общих принципах работы вирусов, осуществляется восстановление файлов. Если вирус подверг файл необратимым изменениям, то восстановление возможно только с использованием резервной копии или с дистрибутива. При их отсутствии существует только один выход - уничтожить файл и восстановить его вручную.

Если антивирусная программа не может восстановить главную загрузочную запись или загрузочные сектора, то можно попытаться это сделать вручную. В случае неудачи следует отформатировать диск и установить ОС.

Существуют вирусы, которые, попадая в ЭВМ, становятся частью его ОС. Если просто удалить такой вирус, то система становится неработоспособной.

Одним из таких вирусов является вирус One Half. При загрузке ЭВМ вирус постепенно зашифровывает жесткий диск. При обращении к уже зашифрованным секторам резидентный вирус One Half перехватывает обращения и расшифровывает информацию. Удаление вируса приведет к невозможности использовать зашифрованную часть диска. При удалении такого вируса необходимо сначала расшифровать информацию на диске. Для этого необходимо знать механизм действия вируса.

Практическая работа № 4.17. Установка и настройка антивируса. Настройка обновлений с помощью зеркала

Цель работы: изучить системные требования антивируса, настройка обновлений с помощью зеркал

Ход работы

Задание 1. Системные требования

Убедитесь, что ваша операционная система является Microsoft Windows XP и что на вашем компьютере не установлено другое антивирусное ПО, совместная работа с которым может вызвать конфликты.

Системные требования обычно приводятся в сопровождающем дистрибутив текстовом файле и/или в документации к продукту. Также всегда с ними можно ознакомиться на сайте компании-производителя.

В этом задании нужно сравнить системные требования **Антивируса Касперского 6.0** с конфигурацией Вашего компьютера и убедиться, что установка этого приложения возможна.

1. Узнайте версию операционной системы, в которой Вы работаете. Для этого найдите иконку **Мой компьютер**, выведите ее контекстное меню (щелкнув на ней правой кнопкой мыши) и выберите пункт **Свойства**

2. Открывшееся окно **Свойства системы** содержит основные сведения о компьютере и установленной на нем операционной системе. На первой закладке, **Общие**, представлена сводная информация, в том числе название и версия операционной системы. На картинке это **Microsoft Windows XP Professional** с установленным **Service Pack 2**. Запомните название и версию Вашей операционной системы

3. Откройте файл с документацией к **Антивирусу Касперского 6.0**, kav6.0ru. pdf.

4. Перейдите к разделу **2.3. Аппаратные и программные требования к системе** и найдите в списке операционных систем Вашу, например, **"Microsoft Windows XP Home Edition или XP Professional (Service Pack 1 или выше)"**. Непосредственно после указания операционной системы будет идти перечень системных требований, предъявляемых к компьютеру с Вашей операционной системой. Соберите воедино все требования, предъявляемые к Вашей системе и заполните столбец «Требования Антивируса Касперского» следующей таблицы:

Параметр	Требования Антивируса Касперского	Параметры системы
Процессор		
Оперативная память		
Свободное место на диске		
Браузер		

5. Далее необходимо убедиться, что конфигурация системы позволяет установить Антивирус Касперского. Для этого вернитесь к окну **Свойства системы** (см. пункты 1 и 2 этого задания). В разделе **Компьютер** можно получить информацию и о процессоре, и об объеме оперативной памяти. В примере это Intel(R) Celeron(R) 1,70 ГГц и 192 МБ оперативной памяти

6. Внесите полученные данные в третий столбец «Параметры системы» таблицы пункта 5.

7. Проверьте наличие свободного места на диске. Для этого откройте папку **Мой компьютер** и задержите на пару секунд курсор мыши над иконкой системного диска. В появившемся сообщении будет указан объем свободного пространства на нем и общий объем диска. На рисунке это локальный диск С: общей емкостью 3,99 ГБ, на котором свободно 1,38 ГБ. Занесите полученные Вами данные в общую таблицу в строку «Свободное место на диске»

Узнайте версию установленного на Вашем компьютере браузера. Браузер **Internet Explorer** встроен в любую операционную систему семейства **Microsoft Windows**, однако версия его может отличаться от требуемой. Запустите браузер, откройте меню **Справка** и выберите пункт **О программе**

8. В открывшемся окне найдите версию **Internet Explorer**, в данном случае это 6.0.2900. Внесите это значение в таблицу из пункта 5 (графа «Браузер») и закройте приложение

9.Проанализируйте заполненную таблицу и сделайте выводы о возможности установки **Антивируса Касперского 6.0** на Ваш компьютер

10. Далее необходимо ознакомиться со списком установленных на компьютер программ и убедиться, что среди них нет других антивирусов. Для этого вызовите **Панель управления (Пуск / Настройка / Панель управления)**

11. В Панели управления найдите элемент **Установка и удаление программ** и откройте его

12. Ознакомьтесь со списком установленных на компьютере программ и убедитесь, что среди них нет других антивирусов

13. Обратите внимание на системную дату, установленную на Вашем компьютере. Для этого задержите на пару секунд курсор мышки над системным временем в правом нижнем углу экрана. Системная дата должна соответствовать реальной дате, это будет необходимо для корректной активации продукта

На этом подготовительный этап окончен и можно переходить непосредственно к установке.

Задание 2. Установка

Стандартная процедура установки включает в себя копирование необходимых в работе программы файлов на диск (в нужное место) и регистрацию в реестре операционной системы. Иногда для завершения установки требуется перезагрузка компьютера.

Для успешной установки **Антивируса Касперского** требуется дистрибутив и лицензионный ключ (файл с расширением. key, содержащий данные, удостоверяющие легальность приобретенного продукта). Эти файлы обычно записываются на CD и передаются пользователю при покупке. В случае приобретения в Интернет-магазине, дистрибутив можно либо загрузить с сайта **Лаборатории Касперского**, либо заказать отправку почтой или курьером на CD, лицензионный ключ высылается по e-mail.

В этом задании необходимо произвести установку **Антивируса Касперского 6.0**. Для этого нужно запустить **Мастер установки** и проследовать за всеми его указаниями. По окончании установки запустится **Мастер настройки**. Он позволяет в режиме диалога с пользователем произвести настройку основных параметров работы антивируса. В большинстве случаев после этой процедуры дополнительная настройка по окончании инсталляции не требуется.

1.Откройте папку с дистрибутивом **Антивируса Касперского**.

2. Найдите файл **setup. exe** и запустите его

3. Если система удовлетворяет всем необходимым **Антивирусу Касперского** требованиям, запустится **Мастер установки**. В первом окне он поприветствует Вас и сообщит, что собирается сделать. Внимательно прочтите предложенный текст, выполните указание закрыть все сторонние открытые приложения (если таковые имеются) и нажмите кнопку **Далее** для перехода к следующему окну **Мастера**

4. На втором шаге **Мастера** необходимо ознакомиться с Лицензионным соглашением между Вами и **Лабораторией Касперского**, производителем **Антивируса Касперского**. В нем описаны все права и обязанности обоих сторон, в том числе ответственность за нарушение авторских прав и самостоятельное изготовление копий антивируса. Внимательно прочтите его. Установку можно продолжить только согласившись со всеми положениями, для этого нужно отметить пункт **Я принимаю условия Лицензионного соглашения** и нажать ставшую активной кнопку **Далее**

5. На следующем шаге нужно определить директорию, куда будут скопированы основные системные файлы антивируса. По умолчанию предлагается использовать C:\Program

Files\Kaspersky Lab\Kaspersky Anti-Virus 6.0\. Если она по каким-то причинам не подходит, с помощью кнопки **Обзор** всегда можно выбрать другую. Для продолжения установки и перехода к следующему окну тут и в дальнейшем используйте кнопку **Далее**

6. Далее нужно выбрать тип установки: полную или выборочную. Полная означает установку всех компонентов **Антивируса Касперского**, а выборочная позволяет некоторые из них отключить. Выберите **Выборочную**, нажав на квадратную кнопку слева от описания этого типа установки

7. Как и было обещано, в следующем окне можно указать какие компоненты **Антивируса Касперского** необходимо установить, а какие пропустить. На рисунке изображен вид этого окна по умолчанию, соответствующий полной установке. Будет ли установлен тот или иной компонент символизирует иконка слева от него: - устанавливать, - нет

Тут же можно получить краткое описание каждого компонента - для этого необходимо выделить (щелкнуть правой кнопкой мыши) интересующий компонент и внизу окна появится нужная информация. На рисунке выделен **Антивирус Касперского 6.0**, следовательно внизу показано описание самой программы.

Оставьте установку всех компонентов и продолжите инсталляцию, нажав **Далее**

8.Далее **Мастер** проверяет наличие на компьютере других антивирусных программ, полный список которых можно найти в файле release_notes. txt в разделе "**Установка**". Если такие найдутся, то пользователю будет выведено соответствующее уведомление с предложением их удалить. Но в нашем случае компьютер чист и этот этап в интерфейсе никак не отображается

9. На следующем этапе нужно подтвердить намерение установить программу, нажав **Установить**. После этого начнется непосредственное копирование файлов и регистрация программы в реестре, и вернуться к предыдущим окнам **Мастера** установки будет невозможно.

Расположенный в центре окна флаг **Включить защиту модулей до начала установки** рекомендуется оставить включенным. Но в дальнейшем, при повторной инсталляции этой же версии **Антивируса Касперского** его следует очищать. Он отвечает за сохранность сделанных во время установки настроек, они могут потребоваться в дальнейшем для восстановления **Антивируса Касперского** в случае повреждения его программных модулей

10. Нажмите кнопку **Установить** и проследите за действиями **Мастера**. Они описываются непосредственно над индикатором процесса установки

11. По окончании инсталляции **Мастер установки** выводит информационное окно. Вам необходимо ознакомится с расположенным в нем текстом и запустить **Мастер настройки** приложения. Для этого нажмите **Далее**

12. На первом этапе настройки нужно активировать приложение. Это можно сделать одним из четырех предложенных вариантов:

Используя код активации, коммерческий или пробный. Такой код может быть выдан при покупке через Интернет, в этом случае активация происходит также через Интернет Активировать используя полученный ранее ключевой файл - именно этот способ будет использован в этой лабораторной работе Активировать позже - если ключевого файла нет, то можно установить антивирус в пробном режиме, но в этом случае не будет доступно обновление антивирусных баз и следовательно, надежную защиту получить не получится

Выберите вариант **Использовать полученный ранее лицензионный ключ** и нажмите **Далее**

13. В следующем окне нужно указать путь к лицензионному файлу. Для этого нажмите кнопку **Обзор**

14. Перейдите к указанной преподавателем папке с ключевым файлом, выделите его и нажмите **Открыть**

15. После открытия выбранного файла, в окне **Мастера** появится информация о нем. Ознакомьтесь с ней и нажмите **Далее**

16. На этапе перехода к следующему окну проводится проверка открытого лицензионного ключа. Если он действителен, то происходит его активация. Для продолжения настройки нажмите **Далее**

17. После активации начинается этап первоначальной настройки антивируса. **Мастер установки** предлагает настроить только основные параметры работы приложения и все сделанные в ходе инсталляции настройки впоследствии можно будет легко изменить с помощью графического интерфейса.

Первое окно предлагает выбрать режим интерактивной защиты. Прочитайте описание различий между этими двумя режимами, оставьте выбранную по умолчанию **Базовую защиту** и нажмите **Далее**

18. Далее предлагается определить режим обновления, по умолчанию выбран пункт **Автоматически**. Он подходит для большинства пользователей. В этой лабораторной работе оставьте все настройки по умолчанию, поскольку задача обновления антивирусных баз будет подробно рассмотрена в одной из следующих лабораторных работ. Однако нужно знать, что в общем случае настроить и обновить антивирусные баз можно уже прямо в ходе установки (для этого предназначены кнопки **Настройка** и **Обновить сейчас** и меню выбора режима обновления)

19. В следующем окне можно задать настройки и расписание запуска проверки на наличие вирусов объектов автозапуска, критических областей и полной проверки компьютера.

Для большинства пользователей рекомендуется настроить проверку объектов автозапуска (как наиболее часто поражаемой области компьютера) при каждой перезагрузке **Антивируса Касперского**. Это обычно соответствует каждой перезагрузке компьютера.

Под проверкой критических областей подразумевается поиск вирусов в важных системных областях. По умолчанию это системная память, объекты автозапуска, загрузочные секторы дисков и папки C:\Windows и C:\Windows\system32.

Полную проверку компьютера рекомендуется проводить раз в неделю. Однако поскольку она требует несколько больше системных ресурсов и соответственно может снижать общую производительность компьютера, оптимального расписания для всех пользователей нет. Поэтому если при установке на домашний компьютер Вы заранее знаете, что в определенный день и час полная проверка не будет мешать Вашей работе, то можете смело отмечать флаг **Каждый 1 день** в поле **Полная проверка компьютера** и с помощью размещенной рядом и ставшей активной кнопки **Изменить** устанавливать расписание - например, каждую пятницу в 20:00. Иначе необходимо помнить о важности регулярной полной проверки и запускать ее вручную, но опять же, не реже раза в неделю

В этой лабораторной работе оставьте отмеченным только флаг проверки объектов автозапуска и нажмите **Далее**

20. **Антивирус Касперского** позволяет поставить защиту паролем на ряд операций: изменения настроек, выгрузки антивируса или остановки работы компонентов и задач поиска вирусов. Если такая защита установлена, то при попытке совершить защищенную операцию будет предложено ввести пароль. Это может быть полезно, если компьютер используется несколькими пользователями и кому-то из них нельзя доверять.

В этой лабораторной работе устанавливать пароли не нужно, поэтому оставьте флаг **Включить**

защиту паролем пустым и нажмите **Далее**

21. На последнем этапе **Мастер настройки** проводит анализ Вашей системы и собирает данные об установленных программах. В дальнейшем эта информация пригодится для контроля целостности приложений, дополнительного компонента антивирусной защиты.

Дождитесь окончания сбора сведений о системе

22. Следующее окно информирует, что установка завершена, но требует перезагрузки. Без перезагрузки установка **Антивируса Касперского** не может считаться завершенной. Поэтому убирать отметку с флага **Перезагрузить компьютер** можно только в исключительных случаях. В данном случае это не требуется.

Оставьте отмеченным флаг **Перезагрузить компьютер** и нажмите **Готово**

23. Дождитесь завершения перезагрузки компьютера и войдите в систему под своей учетной записью

24. Обратите внимание, что после перезагрузки в правом нижнем углу экрана появилось сообщение о необходимости провести полную проверку компьютера на вирусы. О том, как настраивать такие уведомления, пойдет речь в одной из следующих лабораторных работ

Заключение

Эта лабораторная работа заканчивается полной установкой, включающей в себя предварительную настройку **Антивируса Касперского 6.0**. Если в ходе инсталляции **Мастер установки** не выводил сообщений об ошибках, она должна быть успешной.

Практическая работа № 4.18. Настройка политики безопасности

Цель работы: ознакомиться с методами ограничения доступа к информации

Теоретический материал

Разграничение доступа является достаточно эффективным средством предупреждения возможного ущерба вследствие нарушения целостности или конфиденциальности информации. В том случае, если доступ к самому компьютеру или к его ресурсам может получить пользователь, который имеет злой умысел или недостаточный уровень подготовки, он может случайно или преднамеренно исказить информацию или уничтожить ее полностью или частично.

Это же обстоятельство может привести к раскрытию закрытой информации или несанкционированному тиражированию открытой, например, программ, баз данных, разного рода документации, литературных произведений и т. д. в нарушение прав собственников информации, авторских прав…

С точки зрения разграничения доступа, в информационных системах следует различать *субъекты доступа и объекты доступа*. В число *субъектов доступа* могут войти либо персонал информационной системы, либо посторонние лица. *Объектами доступа* являются аппаратно-программные элементы информационных систем. Чаще всего в качестве объектов доступа рассматриваются файлы (в том числе папки и файлы программ). Доступ к объекту может рассматриваться либо как чтение (получение информации из него), либо как изменение (запись информации в него). Тогда виды доступа определяются следующими возможными сочетаниями этих операций:

- ни чтение, ни изменение;
- только чтение;
- только изменение;
- и чтение, и изменение.

Очевидно, что различие функциональных обязанностей субъектов обусловливает необходимость предоставления им соответствующих видов доступа.

Управление доступом пользователей и глобальными параметрами на членах домена осуществляется на двух уровнях: локальной системы и домена. На отдельных компьютерах доступ пользователей конфигурируют на уровне локальной системы, а одновременно для нескольких систем или ресурсов, входящих в домен, — на уровне домена.

Права доступа пользователя определяются руководителем организации и прописываются на рабочей станции системным администратором (администратором домена).

Процедура проверки прав доступа включает авторизацию и аутентификацию. Авторизация предполагает проверку уровня доступа к объекту, а аутентификация — проверку подлинности пользователя. Для аутентификации обычно используются имя пользователя (login) и пароль (password). Системный администратор осуществляет разграничение прав доступа в соответствии с заданной системной политикой, которая предполагает:

- ограничения на минимальную длину, сложность и срок действия пароля;

– требование уникальности паролей;
– блокировку пользователя при неудачной аутентификации;
– ограничение времени и места работы пользователя.

Система разграничения доступа реализована так, что при повседневной работе пользователи не должны замечать, что любое обращение к любому объекту проходит проверку на соответствие установленным правам доступа. Списки прав доступа можно задавать на каждый документ отдельно. При создании документа автоматически задается такой доступ на него, чтобы создатель имел все права.

Система разграничения доступа предназначена для реализации определенных администратором защиты правил на выполнение операций пользователями над объектами хранилища.

Система ограничения прав доступа не может дать полной гарантии безопасности информации. Дело в том, что злоумышленник может получить или подобрать пароль легального пользователя. Кроме того, опытный специалист может обойти систему разграничения доступа. Средством обнаружения несанкционированного доступа к ресурсам служат системы аудита, которые автоматически фиксируют доступ к файлам и папкам и системные события.

Модели разграничения доступа

Наиболее распространенные модели разграничения доступа:

– дискреционная (избирательная) модель разграничения доступа;
– полномочная (*мандатная*) модель разграничения доступа.

Дискреционная модель характеризуется следующими правилами:

– любой объект имеет владельца;
– владелец имеет право произвольно ограничивать доступ субъектов к данному объекту;
– для каждого набора субъект – объект – метод право на доступ определен однозначно;
– наличие хотя бы одного привилегированного пользователя (например, администратора), который имеет возможность обращаться к любому объекту с помощью любого метода доступа.

В дискреционной модели определение прав доступа хранится в матрице доступа: в строках перечислены субъекты, а в столбцах – объекты. В каждой ячейке матрицы хранятся права доступа данного субъекта к данному объекту.

Полномочная модель характеризуется следующими правилами:

– каждый объект обладает грифом секретности. Гриф секретности имеет числовое значение: чем оно больше, тем выше секретность объекта;
– у каждого субъекта доступа есть уровень допуска. Допуск к объекту в этой модели субъект получает только в случае, когда у субъекта значение уровня допуска не меньше значения грифа секретности объекта.

Преимущество полномочной модели состоит в отсутствии необходимости хранения больших объемов информации о разграничении доступа. Каждым субъектом выполняется хранение лишь значения своего уровня доступа, а каждым объектом – значения своего грифа секретности.

Методы разграничения доступа

Виды методов разграничения доступа:

Разграничение доступа по спискам

Суть метода состоит в задании соответствий: для каждого пользователя задается список ресурсов и права доступа к ним или для каждого ресурса определяется список пользователей и права доступа к этим ресурсам. С помощью списков возможно установление прав с точностью до каждого пользователя. Возможен вариант добавления прав или явного запрета доступа. Метод доступа по спискам используется в подсистемах безопасности операционных систем и систем управления базами данных.

Использование матрицы установления полномочий

При использовании матрицы установления полномочий применяется матрица доступа (таблица полномочий). В матрице доступа в строках записываются идентификаторы субъектов, которые имеют доступ в компьютерную систему, а в столбцах – объекты (ресурсы) компьютерной системы. В каждой ячейке матрицы может содержаться имя и размер ресурса, право доступа (чтение, запись и др.), ссылка на другую информационную структуру, которая уточняет права доступа, ссылка на программу, которая управляет правами доступа и др. Данный метод является достаточно удобным, так как вся информация о полномочиях сохраняется в единой таблице. Недостаток матрицы – ее возможная громоздкость.

Разграничение доступа по уровням секретности и категориям

Разграничение по степени секретности разделяется на несколько уровней. Полномочия каждого пользователя могут быть заданы в соответствии с максимальным уровнем секретности, к которому он допущен. При разграничении по категориям задается и контролируется ранг категории пользователей. Таким образом, все ресурсы компьютерной системы разделены по уровням важности, причем каждому уровню соответствует категория пользователей.

Парольное разграничение доступа

Парольное разграничение использует методы доступа субъектов к объектам с помощью пароля. Постоянное использование паролей приводит к неудобствам для пользователей и временным задержкам. По этой причине методы парольного разграничения используются в исключительных ситуациях.

На практике принято сочетать разные методы разграничений доступа. Например, первые три метода усиливаются парольной защитой. Использование разграничения прав доступа является обязательным условием защищенной информационной системы.

Ход работы

1. *Освоить средства разграничения доступа пользователей к папкам*:
   - выполнить команду «Общий доступ и безопасность» контекстного меню папки (если эта команда недоступна, то выключить режим «Использовать простой общий доступ к файлам» на вкладке «Вид» окна свойств папки) или команду «Свойства»;
   - открыть вкладку «Безопасность» и включить в отчет сведения о субъектах, которым разрешен доступ к папке и о разрешенных для них видах доступа;
   - с помощью кнопки «Дополнительно» открыть окно дополнительных параметров безопасности папки (вкладка «Разрешения»);
   - включить в отчет сведения о полном наборе прав доступа к папке для каждого из имеющихся в списке субъектов;
   - открыть вкладку «Владелец», включить в отчет сведения о владельце папки и о возможности его изменения обычным пользователем;
   - открыть папку «Аудит», включить в отчет сведения о назначении параметров аудита, устанавливаемых на этой вкладке, и о возможности их установки обычным пользователем;
   - закрыть окно дополнительных параметров безопасности.

2. *Освоить средства разграничения доступа пользователей к файлам*:
   - выполнить команду «Свойства» контекстного меню файла;
   - повторить все задания п. 1, но применительно не к папке, а к файлу.

3. *Освоить средства разграничения доступа к принтерам*:
   - выполнить команду «Принтеры и факсы» меню «Пуск»;
   - выполнить команду «Свойства» контекстного меню установленного в системе принтера;
   - повторить все задания п. 1, но применительно не к папке, а к принтеру.

4. *Освоить средства разграничения доступа к разделам реестра операционной системы*:

– с помощью команды «Выполнить» меню «Пуск» запустить программу редактирования системного реестра regedit (regedt32);
– с помощью команды «Разрешения» меню «Правка» редактора реестра определить сведения о правах доступа пользователей к корневым разделам реестра, их владельцах и параметрах политики аудита;
– включить в отчет сведения о правах доступа пользователей к данной папке и о ее владельце.

Практическая работа № 4.19 Настройка программы-браузера

Цель работы: изучить способы настройки браузера

Теоретический материал

Браузер – это программа для просмотра web-страниц.

Настройка браузера. Все браузеры позволяют выполнить некоторые настройки для оптимизации работы пользователей в Интернете. В браузереInternet Explorer основная часть настроек содержится в меню Сервис – Свойства обозревателя.

Вкладка Общие позволяет задать адрес домашней страницы, которая будет автоматически загружаться в окно браузера при его запуске, цвета гиперссылок по умолчанию, название шрифта по умолчанию. Здесь же определяется сколько дней будет храниться ссылка посещенных страниц в журнале. Кроме того, для ускорения просмотра. Все посещенные страницы помещаются в специальную папку, и с помощью кнопки Параметры можно задать разные способы обновления таких страниц.

С помощью **вкладки Безопасность** можно создать списки надежных узлов и узлов с ограниченными функциями. Зона Интернет будет при этом включать все остальные узлы, не вошедшие в эти две папки. Для каждой из них с помощью кнопки Другой можно изменить параметры безопасности, установленные для них по умолчанию. Здесь можно запретить выполнение сценариев, отображение всплывающих окон, загрузку файлов и т.д.

Вкладка Конфиденциальность дает возможность настроить работу с файлами cookie, с помощью которых информация о пользователе автоматически передается на сервер.

Вкладка Содержание позволяет ограничить доступ к некоторой информации (насилие, ненормативная лексика и т.д.).

Вкладка Подключения позволяет установить подключение к Интернету.

На вкладке Дополнительно можно задать некоторые дополнительные параметры работы (отключить загрузку графических изображений, отменить подчеркивание ссылок, запретить отладку сценариев и т.д.).

Вкладка Программы позволяет определить программы, которые будут по умолчанию использоваться службами Интернета (почтовые программы,html-редакторы и т.п.).

Ход работы

1. Создайте папку на рабочем столе и переименуйте её.
2. Откройте броузер Internet Explorer.
3. На вкладке Панели инструментов меню Вид уберите все флажки напротив всех панелей инструментов.
4. В меню Вид уберите флажок со вкладки Строка состояния.
5. Нажмите кнопку Print Screen.
6. Откройте графический редактор и вставьте скопированное в рабочую область.
   Настройка панелей инструментов Internet Explorer.
7. Вернитесь к обозревателю и при помощи действий Вид→ Панели инструментов, отобразите на экране Ссылки. Скопируйте в Paint данное окно, сравните с предыдущим рисунком и вырежьте все части, которые дублируют первый рисунок. Вставьте получившееся на фон рабочей области рисунка и подпишите «ссылки».

Вернитесь снова к обозревателю и, проделав аналогичные действия, вставьте в тот же рисунок Адресную строку, Обычные кнопки, строку состояния и подпишите их. Скопируйте аналогичным образом Панели обозревателя: Избранное (часто посещаемые веб-страницы), Журнал (список недавно посещённых веб-страниц), Поиск, Папки.

8. Для просмотра веб-страницами вам нужно научиться изменять размер шрифта, отключить графику для увеличения скорости отображения всех веб-страниц.

Для того, чтобы установить оптимальный для просмотра страницы размер шрифта, нужно сделать следующее Вид→Размер шрифта. Выберите Самый крупный.

9. Чтобы отключить графику для увеличения скорости отображения всех веб-страниц, меню **Сервис** обозревателя Internet Explorer выберите команду **Свойства обозревателя**.
10. Выберите вкладку **Дополнительно**. В группе **Мультимедиа** снимите один или несколько из флажков: **Отображать рисунки**, **Воспроизводить анимацию на веб-страницах**, **Воспроизводить видео на веб-страницах** и **Воспроизводить звуки на веб-страницах**.
11. Чтобы увеличить размер дискового пространства, выделяемого для временного хранения веб-страниц, в меню**Сервис** обозревателя Internet Explorer выберите команду **Свойства обозревателя**.
12. На вкладке **Общие** нажмите кнопку **Параметры**.
13. Чтобы увеличить размер дискового пространства, выделяемого для временного хранения страниц, переместите движок вправо.

Практическая работа № 4.20 Работа с реестром

Цель работы: изучить структуру ключей реестра, типы параметров ключей, способы редактирования реестра; получить практические навыки работы с редактором реестра RegEdit.

Теоретический материал

Структура и основные принципы работы с реестром

Реестр (Registry) – это системная база данных Windows . Она является хранилищем множества параметров и установок, необходимых для нормального функционирования Windows на данном конкретном компьютере.

Реестр – это не статическая база данных настроек, он работает постоянно и постоянно обновляется. Не существует двух одинаковых реестров.

Файлы системного реестра

База данных системного реестра Windows 95 хранится в двух файлах – System.dat и User.dat. Это скрытые системные файлы, доступные только для чтения. Данные хранятся в них в двоичном виде и не могут быть просмотрены при помощи обычного текстового редактора. Для внесения изменения в реестр должен использоваться специальный редактор Regedit.exe, который изображает эти два файла как одну систему.

По умолчанию файлы System.dat и User.dat хранятся в папке \Windows.

В файле System.dat хранятся сведения об аппаратуре, на котором работает система Windows, а также об установленном на нем программном обеспечении. Значения, хранящиеся в этом файле, автоматически изменяются при изменении аппаратной конфигурации, а также при установке и удалении приложений.

В файле User.dat хранится информация, относящаяся к пользователю. В частности, это могут быть данные о «чувствительности» мыши, цветовой схеме, курсорах, шрифтах, клавиатуре и прочем. В этом же файле находятся сведения о конфигурации рабочего стола и сети для разных пользователей – так называемые пользовательские конфигурации.

Аналогичные файлы для хранения базы данных системного реестра существуют и в других ОС семейства Windows 9x/NT. Отличаться могут количество и, соответственно, названия файлов.

Редактор реестра

Фирма Microsoft предусмотрела множество элементов интерфейса пользователя, предназначенных для изменения конфигурации системы , т.е. реестра – это и Панель Управления (Control Panel), и диалоговые окна свойств, и многое другое. При этом изменения параметров отражаются на функционировании системы немедленно. Вместе с тем в некоторых случаях этого оказывается недостаточно. Однако, изменять системный реестр, используя редактор реестра, следует только в том случае, когда это действительно необходимо. Если вы редактируете базу данных реестра, то для того, чтобы хранящиеся в ней параметры были прочтены в память и вступили в силу, чаще всего необходимо перезапустить компьютер.

Для запуска редактора реестра следует выполнить команду Пуск - Выполнить- RegEdit. Файл запуска реестра RegEdit.exe всегда находится в папке \ WINDOWS.

Объекты системного реестра

Реестр содержит три типа объектов: ключи, параметры и значения.

Ключи - вершина иерархической структуры реестра. Под ключами реестра могут располагаться другие узлы иерархического дерева (подключи). Кроме этого, каждый ключ может содержать один или несколько параметров. Все ключи и параметры в пределах подключа должны иметь уникальные имена.

Параметры имеются у каждого ключа и подключа. У каждого ключа обязательно есть хотя бы один параметр -" По умолчанию". Если значения параметров не заданы, то они имеют значение Null.

Параметры состоят из трех частей: тип параметра, имя параметра и его значение. Допустимы следующие типы параметров: двоичные, двойное слово и строковые. Каждому типу параметров соответствует своя пиктограмма в окне редактора реестра.

String (строковое). Представляет из себя ASCIIZ–строку (заканчивается символом с кодом 0). Имеет переменную длину, максимальный размер 64 кБ. Значение строки всегда заключается в кавычки.

Binary (двоичное). Максимальный размер 64 кБ. В окне редактора реестра представлено в виде 16-ричного значения.

DWORD (двойное слово). Представляет собой число размером 32 бита (в реестре 8-значное шестнадцатеричное число). Чтобы отличить этот тип данных от двоичного, перед численным значением DWORD всегда есть два символа: 0x.

Структура системного реестра

Вся база системного реестра разделена на шесть основных разделов, которые принято называть ветвями. Каждая ветвь содержит в себе параметры, относящиеся к определенному набору ключей. Ниже кратко описано назначение этих разделов.

[HKEY_CLASSES_ROOT]

Содержит сведения о встраивании и связывании объектов (Object Linking and Embedding, OLE) и ассоциации файлов с приложениями.

[HKEY_USERS]

Содержит информацию обо всех пользователях данной рабочей станции. Здесь хранятся данные о каждом пользователе, а также типовые настройки, служащие шаблоном для новых ключей, создаваемых пользователем. Типовые настройки включают различные значения по умолчанию для программ, событий, конфигураций рабочего стола и т.д.

[HKEY_CURRENT_USER]

Содержит настройки системы и программ, относящиеся к текущему пользователю. Он создается при регистрации пользователя в системе на основе информации из соответствующего ключа [HKEY_USERS]. Именно здесь хранится информация о том, как данный пользователь сконфигурировал рабочую станцию.

[HKEY_LOCAL_MACHINE]

Содержит спецификации рабочей станции, драйверов и др. системные настроцки, включая информацию о типах установленного оборудования, настройках портов конфигурации программного обеспечения. Эта информация специфична для компьютера, а не для пользователя.

[HKEY_CURRENT_CONFIG]

Содержит информацию о текущей конфигурации аппаратуры компьютера, используется в основном на компьютерах с несколькими аппаратными конфигурациями, например, при подключении портативного ПК к стыковочной станции и отключении от нее. Информация , содержащаяся в этом ключе, копируется из ключа [HKEY_LOCAL_MACHINE].

[HKEY_DYN_DATA]

Содержит динамическую информацию о состоянии различных устройств, причем она создается заново при каждом старте системы. Этот ключ используется как часть системы измерения производительности и для конфигурации устройств Plug-and-Play.

Состав основных разделов

Каждый из вышеперечисленных разделов содержит в себе другие разделы — как и файловая система, Registry имеет структуру дерева. Каждый узел (раздел или подраздел) называется ключом. Вы можете открывать новые ветви до тех пор, пока не доберетесь до уровня, на котором находятся только параметры.Рис.1. Окно редактора реестра

Registry Editor	
Registry Edit View Help	
HKEY_CLASSES_ROOT HKEY_CURRENT_USER HKEY_LOCAL_MACHINE HKEY_USERS HKEY_CURRENT_CONFIG HKEY_DYN_DATA	

Hkey_Classes_Root

Структура раздела несколько отличается от всех остальных. Для каждого зарегистрированного расширения файла имеется подключ (например, .bmp).

Значение этого ключа "По умолчанию" указывает на подключ описания документа ("ACDC_BMP"), который расположен в той же ветви основного раздела. В подключе описания документа и содержится цепочка ключей, хранящих информацию об ассоциациях, OLE, DDE.

Hkey_Local_Machine

Информация, сохраненная здесь, используется приложениями, устройствами и системой, и не зависит от того, кто был заявлен в качестве пользователя. Устройства могут помещать информацию в системный реестр с помощью P1ug&Play-интерфейса, программные средства — посредством стандартного API. Hkey_Local_Machine содержит ряд подразделов, описанных в табл.1.

Подраздел Config

• Содержит информацию о различных конфигурациях аппаратных средств.

• Каждая конфигурация имеет уникальное обозначение и хранится в отдельном подразделе с соответствующим именем.

• Конфигурации перечислены в списке в окне утилиты **Система.** Здесь же их можно обрабатывать.

• При запуске Windows проводится проверка конфигурации аппаратных средств. При этом может произойти следующее:

• В большинстве случаев конфигурационные данные позволяют Windows автоматически выбрать соответствующую конфигурацию.

• При первом после изменения оборудования запуске компьютера Windows создает новый элемент конфигурации для новых конфигурационных данных. В результате создается и новый Config-элемент в системном реестре.

• Когда конфигурационные данные не позволяют системе Windows однозначно решить, какую из описанных конфигураций следует выбрать, пользователю при загрузке системы предлагается меню, посредством которого он может выбрать подходящую конфигурацию.

Подраздел Enum

• Windows располагает специальными программами, которые отвечают за построение дерева аппаратуры в системном реестре (например, Диспетчер устройств, вызываемый через Панель управления - Система- Устройства).

• Каждому устройству присваивается уникальный идентификационный код.

• В системном реестре хранится идентификационная информация о каждом устройстве, например, тип устройства, идентификационный код (ID) устройства, информация об изготовителе и информация о драйвере.

Информация о составе данного раздела приведена в табл.2.

Подраздел Software

• Содержит информацию о каждом программном средстве, установленном на компьютере.

• Содержимое этого раздела является общим для всех пользователей данного компьютера.

• Hkey_Local_Machine\Software содержит ряд подразделов и сведения о различных подразделах (их описание), которые могут появиться в системном реестре (см.табл.3).

Подраздел System

• Данные в подразделе System содержат все параметры драйверов устройств и служб, используемые при запуске Windows.

• Вся информация хранится в подразделе CurrentControlSet. Он содержит два следующих подраздела:

• *Control*: Подраздел включает информацию, используемую, при запуске системы, например, сетевое имя компьютера и запускаемые подсистемы.

• *Services*: Подраздел включает информацию, необходимую для контроля загрузки и конфигурирования драйверов, файловой системы, и др. Здесь также определяется, как отдельные службы вызывают одна другую.

Состав двух вышеназванных подразделов приведен в табл.4 и 5.

Hkey_Current_User и Hkey_Users

• Содержит Default-подраздел и подразделы для всех пользователей, заявленных в системе.

• Информация из подраздела Default используется для того, чтобы создать конфигурацию для нового пользователя.

• Hkey_Current_User содержит информацию о пользователе, работающем на компьютере в текущем сеансе (см. табл.6).

Если существуют одинаковые параметры в Hkey_Local_Machine и Hkey_Current_User, то используются значения параметров, взятые из Hkey_Current_User.

Hkey_Current_Config и Hkey_Dyn_Data

• Hkey_Current_Config указывает на текущую системную конфигурацию, которая сохранена в Hkey_Local_Machine\Config.

• Часть системной информации в Windows должна постоянно присутствовать в оперативной памяти, поскольку системе необходим быстрый доступ к этой информации и Windows не может ожидать, пока нужные данные будут прочитаны с жесткого диска. Вся эта информация находится в Hkey_Dyn_Data.

• Подраздел Hkey_Dyn_Data\Configuration Manager, называемый также деревом аппаратуры, представляет собой хранящееся в оперативной памяти описание текущей системной конфигурации.

• Дерево аппаратуры создается заново при каждом запуске системы и адаптируется, если в состав или конфигурацию аппаратуры были внесены изменения. Присутствующие в этом разделе данные можно просмотреть с помощью Редактора реестра, они всегда соответствуют текущему состоянию аппаратуры компьютера.

• Hkey_Dyn_Data содержит статистическую информацию о различных сетевых компонентах в системе. Она находится в подразделе PerfStats.

Таблица 1. Состав основного раздела Hkey_Local_Machine

Раздел	Назначение
Config	Различные конфигурации компьютера.
Enum	Информация о подключенных к данному компьютеру устройствах.
Hardware	Информация о последовательных интерфейсах и модемах, которые используются программой HyperTerminal.
Network	Хранящаяся здесь сетевая информация создается при входе пользователя в сеть: имя пользователя, регистрационная информация, первичный поставщик услуг и другие сведения.
Security	Информация о том, какой компьютер в сети следит за безопасностью сети и поддерживает ли (допускает ли) данный компьютер удаленное управление.
Software	Информация о программных средствах, установленных на данном компьютере, и различные конфигурационные данные программ.
System	Информация данного раздела управляет запуском системы, загрузкой драйверов устройств, сервисом Windows и поведением системы.

Таблица 2.Состав подраздела Hkey_Local_Machine \Enum

Подраздел	**Устройства**
ESDI	Жесткие диски ESDI -
FLOP	Дисководы для гибких дисков
ISAPNP	Plug & Play устройства, подключенные к ISA-
Monitor	Дисплеи
Network	Сетевые протоколы
Root	Другие компоненты системы

Таблица 3. Состав подраздела Hkey_Local_Machine \Software

Подраздел	Назначение
Classes	Подраздел Classes имеет особое значение. Он определяет типы документов и возможные OLE-связи. Hkey Classes Root является псевдонимом (Alias) данного подраздела. Кроме того, он имеет решающее значение для совместимости с Windows 3.1-реестром. Подраздел Classes содержит два типа подразделов. Первый тип подразделов: соответствующие расширениям имен файлов, содержащие информацию, с помощью которой система в состоянии открыть документ с данным расширением. Второй — описания OLE или DDE параметров (протоколов) для определенного класса документов.
Description	Содержит имя и номер версии программного средства, установленного на компьютере. Пользовательская информация о конфигурации приложения сохраняется в аналогичном подразделе в Hkey Current User.
Microsoft	Содержит информацию о программах, которые поддерживают сервис, встроенный в систему Windows .

Таблица 4. Состав подраздела Hkey_Local_Machine \System\CurrentControlSet\Services

Подраздел	Хранящаяся в подразделе информация
Arbitrators	Информация, необходимая для разрешения конфликтов между устройствами, например, данные об адресах, канале DMA, диапазоне ввода/вывода и запроса на прерывание
Class	Содержит подраздел для каждого из типов устройств, поддерживаемых системой.
MSNP32, NWNP32	Содержит подраздел для 32-разрядного сетевого драйвера защищенного режима и информацию о заявке в сети.
VxD	Содержит подраздел для каждого виртуального драйвера устройств (VxD).

Таблица 5.Состав подраздела Hkey_Local_Machine\System\CurrentControlSet\Control

Подраздел	**Хранящаяся в подразделе информация**
ComputerName	Сетевое имя компьютера (см. 6.2.2)
FileSystem	Тип и установки используемой файловой системы.
IDConfigDB	Идентификационный код текущей конфигурации.
Keyboard layouts	Список раскладок клавиатуры и соответствующих DLL-модулей для поддерживаемых языков.
MediaResources	Описание мультимедиа-компонентов и ' информация о соответствующих драйверах
NetworkProvider	Имена подразделов Services
Nis	Информация о национальной языковой поддержке.
PerfStats	Статистика о компонентах системы. Ее можно просмотреть с помощью утилиты Системный монитор.
Print	Информация об установленных принтерах или сервисном программном обеспечении. Включает ряд подразделов.
Session Manager	Содержит глобальные переменные системы, информацию о программах, которые могут конфликтовать с Windows, и список библиотек DLL, номера версий которых должны быть проверены.
TimeZonelnformation	Параметры для установки времени с учетом часовых поясов.
Update	Информация о том, была ли Windows установлена поверх предыдущей версии.

Таблица 6. Состав основного раздела Hkey_Current_User

Подраздел	**Хранящаяся в подразделе информация**
AppEvents	Пути и имена звуковых файлов, используемых для генерации звуков при определенных событиях в системе.
Control Panel	Установки из Панели управления.
Keyboard layouts	Текущая раскладка клавиатуры.
Network	Информация о текущем состоянии сети.
InstallLocationsMRU	Путь к установочным файлам.
Software	Установки активного пользователя, определяющие режимы работы программ (приложений).

Внимание! Перед началом редактирования реестра обязательно выполните следующие действия:

- подготовьте копии реестра;
- убедитесь, что были исчерпаны все остальные средства, менее опасные, чем редактирование реестра.

Запуск редактора реестра

1) Нажмите **Пуск** и выберите **Выполнить...**

2) В поле **Открыть** введите ***Regedit*** и нажмите **ОК.**

Копирование реестра

Копии реестра создаются автоматически в ОС в файлах System.DA0 и User.Da0. Для принудительной создании копии реестра можно использовать следующие методы:

- скопировать файлы реестра (System.DAT и User.DAT в . Windows95) в файлы с любым другим именем;
- экспортировать реестр в файл с расширением .REG, используя возможности редактора реестра.

Восстановление реестра

1) Нажмите Пуск и выберите Завершение работы...
2) Выберите Перезагрузить компьютер в режиме эмуляции MS-DOS и нажмите Да.
3) После перезагрузки выполните копирование файлов System.DA0 и User.Da0 в файлы System.Dat и User.Dat . Перед копированием для изменения атрибутов файлов реестра используйте внешнюю команду DOS- ATTRIB, которая находится в каталоге C:\ Windows. После копирования восстановите снятые атрибуты файлов реестра.
4) Для получения справки по использованию команды ATTRIB запустите ATTRIB с ключом ? в командной строке DOS.
5) Перезагрузите компьютер. Теперь системный реестр находится в том состоянии, в котором он находился при последнем успешном запуске компьютера.

Ход работы

1) Выполнить резервное копирование файлов системного реестра.
2) Изучить функции редактора реестра Registry Editor:

a) создать новый ключ в разделе Hkey_Current_config, создать для него параметр строкового типа и задать его значение –"Мой"; какой параметр для вновь созданного ключа появляется по умолчанию?

В отчете указать иерархию ключа, названия и значения созданного параметра и параметра по умолчанию.

b) удалить созданные ключ и параметр;

c) найти первых два ключа с полным именем "Setup" ;

в отчете указать иерархию ключа.

d) проверить, имеет ли реестр ключ со значением любого его параметра 35;

отразить результата поиска в отчете;

e) проверить возможность экспортировать реестр в новый файл и импортировать его из ранее сохраненного файла; какое расширение имеют файлы импорта-экспорта реестра?

3) Исследование раздела Hkey_Clauses_Root.

a) Найти ссылку на подключ для файлов с расширением DOC.

b) Найти подключ, на который указывает эта ссылка.

c) Для найденного подключа определить следующие ключи настройки Word: вид графического значка (icon);

командная строка для запуска исполняемого файла.

d) Отредактировать значения параметра "По умолчанию" для этих двух ключей таким образом, чтобы изменился графический значок Word , а также изменилось приложение, которое автоматически запускается при открытии файлов с расширением Doc.

e) Проверить выполненные установки, открыв любой файл Doc.

В отчете указать иерархию двух ключей, название исследуемых параметров, их новое и старое значения.

4) Исследование раздела Hkey_Local_Machine.

Найти подключи конфигурации оборудования. Сколько конфигураций имеет данный компьютер?

В отчете указать иерархию ключа, название исследуемого параметра, его значения.

5) Исследование раздела Hkey_Current_config.

a) Копией какого ключа является данный раздел?
b) Найти ключ, отвечающий за настройки дисплея.
c) Ознакомиться со списком параметров этого ключа.
d) Изменить текущую разрешающую способность монитора на значение "640,480".
e) Для проверки выполнения перегрузить операционную систему (ОС).

В отчете указать иерархию ключа, название исследуемого параметра, его новое и старое значения.

6) Исследование раздела Hkey_Current_user.
   a) Копией какого ключа является данный раздел?
   b) Найти ключ, отвечающий за настройки Рабочего стола. Ознакомиться со списком вложенных ключей . Для произвольно выбранных из списка 5 ключей исследовать, аналогом каких настроек Панели управления они являются.

   В отчете указать иерархию пяти ключей и соответствующие настройки Панели управления.

   c) Изменить с помощью реестра ширину полосы прокрутки и строки командного меню в окнах Windows. Проверить выполненные настройки.

   В отчете указать иерархию ключа, название исследуемого параметра, его новое и старое значения.

   d) В подразделе установленного программного обеспечения для текущего пользователя найти ключ, хранящий полное имя файла справки Word.

   В отчете указать иерархию ключа.

   e) Для приложения Word найти ключ, хранящий информацию о каталоге автоматически сохраняемых документов. Сравнить его с каталогом, указанным в параметрах Word. (Запустить Word, вызвать Сервис-Параметры - Расположение-Автосохраненные).

   Иерархию ключа и результаты сравнения отразить в отчете.

   Проверить влияние изменения параметров приложения Word через меню Сервис на значение параметра автосохранения ключа в реестре, а также обратную связь. *В отчете указать иерархию ключа, название исследуемого параметра.*

   f) Для приложения Excel найти ключ, хранящий информацию о последних 9 загруженных файлах XLS. Запомнить названия параметров и значение одного из них. *Информацию о ключе, параметре и его значении отразить в отчете.*
7) Восстановить состояние системного реестра из резервных копий или из копий ОС.

Практическая работа № 4.21. Работа с программой восстановления файлов и очистки дисков

Цель работы: изучение работы программ восстановления файлов и очистки дисков

Ход работы

Задание 1. Используя задания Сведения о системе, определите следующие параметры компьютерной системы: Мультимедиа, запоминающие устройства, системные драйверы, группы программ, автоматически загружаемые программы.

Для запуска программы **Сведения о системе** выберите в меню Пуск команду **Программы-Стандартные-Служебные-Сведения о системе**.

Для получения сведений об устройствах мультимедиа выберите в дереве категорий в левой области окна программы категорию Компоненты, а в ней подкатегорию Мультимедиа. Выбирая в этой подкатегории элементы мультимедиа аудио- и видеокодеки, CD-ROM, звуковое устройство, дисплей, просмотрите в правой части окна программы сведения, относящиеся к элементу, выделенному в дереве категорий.

Для просмотра сведений о запоминающих устройствах выберите в левой части окна категорию Запоминающие устройства. Выбирая в этой категории различные подкатегории, в области сведений просмотрите информацию об устройствах внешней памяти.

Для просмотра сведений о системных драйверах выберите категорию Программная среда, а в ней подкатегорию Системные драйверы.

Для просмотра данных о группах программ найдите в категории Программная среда подкатегорию Группы программ, чтобы вывести сведения о них в правой области окна.

Аналогично найдите сведения о программах, автоматически загружаемых при старте Windows XP.

Закройте окно программы **Сведения о системе**.

Задание 2. Используя стандартную программу Windows **Проверка диска**, проверьте диск A: на наличие поврежденных секторов и ошибок файловой системы. При этом если будут обнаружены ошибки, то задайте режим восстановления поврежденных секторов диска автоматического исправления системных ошибок.

Перед запуском проверки диска закройте все файлы на нем. Открыв окно *Мой компьютер*, выберите локальный диск A:, затем в меню **Файл** выберите команду **Свойства**. На вкладке **Сервис**в группе **Проверка диска** нажмите кнопку «Выполнить проверку». В группе **Параметры проверки диска** установите флажки **Автоматически исправлять системные ошибки** и **Проверять и восстанавливать поврежденные сектора**.

Для начала процесса сканирования диска на наличие ошибок щелкните на кнопке «Запуск». По окончании проверки диска на экран будет выведено сообщение об окончании проверки диска.

3. Используя стандартную программу **Очистка диска**, выполните очистку диска C:.

Для запуска программы выберите в меню **Пуск** команду **Программы-Стандартные-Служебные-Очистка диска**. В рабочем окне программы выберите логический диск C:, который будет подвергнут процедуре очистки, и щелкните на кнопке «ОК». После этого мастер очистки диска перейдет к процедуре проверки состояния файлов на данном диске. После завершения анализа текущего состояния диска программа представит Отчет о проделанной работе, указав, сколько места можно освободить. Определив, что подлежит удалению при очистке диска, щелкните на кнопке «ОК», а затем подтвердите удаление файлов при очистке диска, щелкнув на кнопке «Да». После этого запускается процесс очистки диска.

4. Используя стандартную программу **Дефрагментация диска**, выполните оценку фрагментированности файлов на диске C: и, если требуется, то выполните дефрагментацию этого диска.

Для запуска программы Дефрагментации диска выберите в меню **Пуск** команду **Программы-Стандартные-Служебные-Дефрагментация диска**. После этого выберите диск C: и нажмите кнопку «Анализ». По завершении анализа тома программа дефрагментации диска выведет результаты анализа и сообщение о том, нуждается ли данный том в дефрагментации. Если в окне сообщения программа рекомендует выполнить дефрагментацию диска, то щелкните на кнопке «Дефрагментация», если иначе - щелкните кнопку «Закрыть». Если была запущена процедура дефрагментации, то после ее окончания результаты будут отображены в графическом представлении с цветовой кодировкой в полях результатов анализа и дефрагментации. Чтобы просмотреть подробный отчет о дефрагментации, нажмите кнопку «Вывести отчет». Закройте окно программы **Дефрагментация диска**.

Задание 3. Используя служебную программу **Архивация данных**, архивируйте данные из папки C:\Program Files\Microsoft Office\Templates в архив с именем Templates на диске D:.

Для запуска приложения Архивация данных выберите в меню **Пуск** команды **Программы-Стандартные-Служебные-Архивация данных**. Если программа архивации запускается в режиме мастера, то для переключения в расширенный режим нажмите кнопку «Расширенный» в окне мастера архивации.

Для архивации выбранных файлов и папок на жестком диске перейдите на вкладку **Архивация** и установите флажок в списке Установите флажки для папки C:\Program Files\Microsoft Offlce\ Templates, данные из которой вы хотите заархивировать.

Задайте в качестве носителя диск D: и имя файла для архива Templates, нажмите на кнопку «Архивировать», а затем в окне *Сведения о задании архивации* выберите вариант **Затереть данные носителя этим архивом**.

Щелчком на кнопке «Архивировать» запустите процедуру архивации. После этого в окне *Ход архивации* пронаблюдайте за процессом архивации, по окончании которого будет выведено окно сообщения о завершении архивации с краткими сведениями. Для просмотра подробного текста отчета щелкните на кнопке «Отчет».

Задание 4. Используя служебную программу **Архивация данных**, создайте архив системных файлов и дискету аварийного восстановления, которые могут быть использованы в целях восстановления системы в случае ее отказа.

Приготовьте чистую дискету емкостью 1,44 Мбайта для сохранения параметров системы, затем запустите приложение Архивация в режиме **Расширенный**. В меню **Сервис** выберите команду **Мастер аварийного восстановления системы**. Следуйте инструкциям, появляющимся на экране. Для перехода к следующему шагу мастера щелкайте на кнопке «Далее». Выбрав тип носителя для системного архива и имя носителя для хранения архивных данных, например, D:\Arxiv\Backup.bkf, щелкните на кнопке «Далее» для создания архива. После этого будет выполнена архивация системных файлов, необходимых для загрузки системы, и создание дискеты аварийного восстановления.

По окончании процесса архивации в ответ на предложение вставить дискету вставьте чистую дискету, после этого будет создана дискета аварийного восстановления. Для просмотра подробного отчета щелкните на кнопке «Отчет». Закройте окно программы **Архивация данных**.

Литература

1. Гвоздева, В. А. Основы построения автоматизированных информационных систем: учебник / В. А. Гвоздева, И. Ю. Лаврентьева. — Москва: ФОРУМ: ИНФРА-М, 2020. — 318 с. — (Среднее профессиональное образование). - ISBN 978-5-8199-0705-4. - Текст: электронный. - URL: https://znanium.com/catalog/product/1066509
2. Исаченко, О. В. Программное обеспечение компьютерных сетей: учебное пособие / О.В. Исаченко. — 2-е изд., испр. и доп. — Москва: ИНФРА-М, 2021. — 158 с. — (Среднее профессиональное образование). - ISBN 978-5-16-015447-3. - Текст: электронный. - URL: https://znanium.com/catalog/product/1189344
3. Лисьев, Г. А. Программное обеспечение компьютерных сетей и web-серверов: учебное пособие / Г.А. Лисьев, П.Ю. Романов, Ю.И. Аскерко. — Москва: ИНФРА-М, 2021. — 145 с. — (Среднее профессиональное образование). - ISBN 978-5-16-014514-3. - Текст: электронный. - URL: https://znanium.com/catalog/product/1189343
4. Организация сетевого администрирования: учебник / А.И. Баранчиков, П.А. Баранчиков, А.Ю. Громов, О.А. Ломтева. — Москва: КУРС: ИНФРА-М, 2020. — 384 с. - ISBN 978-5-906818-34-8. - Текст: электронный. - URL: https://znanium.com/catalog/product/1069157
5. Федорова, Г. Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности: учебное пособие / Г. Н. Федорова. — Москва: КУРС: ИНФРА-М, 2021. — 336 с. - ISBN 978-5-906818-41-6. - Текст: электронный. - URL: https://znanium.com/catalog/product/1138896

Интернет-ресурсы

1. Электронная библиотечная система Znanium: сайт.- URL: https://znanium.com/ – Текст: электронный.
2. Электронная библиотечная система Юрайт: сайт. - URL: https://urait.ru/ -Текс: электронный.