Федеральное агентство по образованию

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

А.И. Елизаров, В.В. Романенко

ТЕХНОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Учебное методическое пособие
для студентов специальности 230105 —
«Программное обеспечение вычислительной техники
и автоматизированных систем»

2007

Корректор: Осипова Е.А.

Елизаров А.И., Романенко В.В.
Технология разработки программного обеспечения: Учебное методическое пособие. — Томск: Томский межвузовский центр дистанционного образования, 2007. — 119 с.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Учебной программой специальности 230105 в рамках изучения дисциплины «Технология разработки программного обеспечения» («ТРПО») для студентов дистанционной формы обучения предусмотрено выполнение двух лабораторных работ:

1. Составление технического задания и соглашения о требованиях.

2. Написание спецификаций и проведение тестирования ПО. Составление руководства системного программиста.

Подготовка курсового проекта является завершающим этапом изучения дисциплины «ТРПО». В период курсового проектирования закрепляются теоретические знания и приобретаются практические навыки разработки программного обеспечения (ПО) и программной документации.

Основной задачей курсового проектирования является разработка программной документации. Курсовой проект должен включать оттестированное программное обеспечение, соответствующее техническому заданию, и пояснительную записку. В состав курсового проекта входят результаты выполнения всех лабораторных работ. Т.о., пояснительная записка к курсовому проекту включает следующие разделы:

1. Техническое задание.
2. Соглашение о требованиях.
3. Внешняя и внутренняя спецификации.
4. Тестирование.
5. Руководство системного программиста.

При этом все замечания, оставшиеся после сдачи лабораторных работ, должны быть устранены.

Разрабатываемое ПО должно содержать не менее 300 операторов, работать в многооконном графическом режиме и поддерживать работу клавиатуры и манипулятора типа «мышь». Программная документация, входящая в состав курсового проекта, должна соответствовать требованиям стандартов Единой системы программной документации (ЕСПД).

Необходимо уделить внимание правильной нумерации разделов. Например, если выполняется лабораторная работа №2, то

подраздел «Режимы испытаний программ» будет иметь номер 2.4 (т.к. раздел «Тестирование» будет вторым в данной лабораторной работе) или просто 4 (если основные разделы не нумеровать). Если же выполняется курсовой проект, то нужно нумеровать его основные разделы, тогда «Тестирование» будет четвертым разделом работы, следовательно, подраздел «Режимы испытаний программ» будет иметь номер 4.4 (как и в данном методическом пособии). Примеры см. в приложениях.

1 ТЕХНИЧЕСКОЕ ЗАДАНИЕ

Составление технического задания — цель первой части первой лабораторной работы. Также техническое задание является первым разделом курсовой работы.

1.1 Содержание

Техническое задание оформляют в соответствии с ГОСТ 19.106-78. Для внесения изменений или дополнений в техническое задание на последующих стадиях разработки программы или программного изделия выпускают дополнение к нему. Согласование и утверждение дополнения к техническому заданию проводят в том же порядке, который установлен для технического задания.

Техническое задание должно содержать следующие разделы:

- введение;
- основания для разработки;
- назначение разработки;
- требования к программе или программному изделию;
- требования к программной документации;
- технико-экономические показатели;
- стадии и этапы разработки.

В зависимости от особенностей программы или программного изделия допускается уточнять содержание разделов, вводить новые разделы или объединять отдельные из них.

1.2 Введение

В разделе «Введение» указывают наименование, краткую характеристику области применения программы или программного изделия и объекта, в котором используют программу или программное изделие.

Пример. Получение заготовок во многих отраслях, таких как судостроение, авиастроение, легкая промышленность, автомобилестроение и др., основано на вырезке необходимых контуров из листового материала с помощью термических уст-

ройств. Минимизация пути режущего инструмента обеспечивает существенную экономию энергии и трудозатрат.

Наименование — система минимизации пути режущего инструмента при раскрое листовых материалов (далее просто минимизатор).

Краткая характеристика — двумерный минимизатор с локальной базой сформированных маршрутов резки.

1.3 Основание для разработки

В разделе «Основания для разработки» должны быть указаны:

– документ (документы), на основании которого ведется разработка;

– организация, утвердившая этот документ, и дата его утверждения;

– наименование и (или) условное обозначение темы разработки.

Пример. Задание для проведения лабораторных занятий и выполнения курсовой работы, выдано кафедрой АСУ ТУСУРа 01.09.2007. Наименование темы разработки — «Минимизатор».

1.4 Назначение разработки

В разделе «Назначение разработки» должно быть указано функциональное (чем является программное изделие) и эксплуатационное (область применения) назначение программы или программного изделия.

Пример. Минимизатор предназначен для формирования минимального пути вырезки многоугольных контуров, размещенных в прямоугольной области, а также для хранения полученных маршрутов в базе данных.

1.5 Требования к программе или программному изделию

Раздел «Требования к программе или программному изделию» должен содержать следующие подразделы:

– требования к функциональным характеристикам;

– требования к надежности;
– условия эксплуатации;
– требования к составу и параметрам технических средств;
– требования к информационной и программной совместимости.

1.5.1 Требования к функциональным характеристикам

В подразделе «Требования к функциональным характеристикам» должны быть указаны требования к составу выполняемых функций, организации входных и выходных данных, временным характеристикам и т.п.

Пример.

1. Редактор должен работать в многооконном графическом режиме и поддерживать работу как клавиатуры, так и манипулятора типа «мышь».

2. Пользователь, по своему желанию, должен иметь возможность установки масштабного поля для каждого окна.

3. Минимизатор должен обеспечивать нахождение минимального пути с проходом только один раз через каждое ребро каждого многоугольного контура детали в области размещения.

4. Найденный путь должен демонстрироваться на экране в различных режимах.

5. Информация о размещении контуров и сформированном маршруте может быть сохранена в локальной базе данных минимизатора.

6. Должен быть обеспечен графический просмотр базы данных с возможностью удаления из нее или копирования в активное окно указанного размещения с имеющимся маршрутом.

7. Информация о размещении и сформированном маршруте может быть выведена в форме файла геометрической информации следующей структуры: …

8. Перечисление вершин контуров деталей в соответствующем дескрипторе выходного файла должно соответствовать сформированному маршруту резки.

9. Программа должна использовать в качестве входной информации файл геометрической информации, первой деталью которого будет прямоугольник области размещения.

10. Программа должна обеспечивать просмотр выходного файла.

1.5.2 Требования к надежности

В подразделе «Требования к надежности» должны быть указаны требования к обеспечению надежного функционирования (устойчивое функционирование, контроль входной и выходной информации, время восстановления после отказа и т.п.).

Пример. Программа должна обрабатывать ошибочные действия пользователя и сообщать ему об этом. Программа должна обеспечивать контроль входной и выходной информации в форме файлов геометрической информации.

1.5.3 Условия эксплуатации

В подразделе «Условия эксплуатации» должны быть указаны условия эксплуатации (температура окружающего воздуха, относительная влажность и т.п. для выбранных типов носителей данных), при которых обеспечиваются заданные характеристики, а также вид обслуживания, необходимое количество и квалификация персонала.

1.5.4 Требования к составу и параметрам технических средств

В подразделе «Требования к составу и параметрам технических средств» указывают необходимый состав технических средств и их основные технические характеристики.

Пример. Программное обеспечение разрабатывается для персональной ЭВМ (IBM PC-совместимой) со следующими характеристиками:

- процессор с частотой не ниже 1 ГГц;
- объем ОЗУ не менее 128 Мб;
- графический адаптер SVGA;
- манипулятор типа «мышь».

1.5.5 Требования к информационной и программной совместимости

В подразделе «Требования к информационной и программной совместимости» должны быть указаны требования к информационным структурам на входе и выходе и методам решения, исходным кодам, языкам программирования и программным средствам, используемым программой. При необходимости должна обеспечиваться защита информации и программ.

Пример. ЭВМ должна работать под управлением операционной системы не ниже, чем Windows 98/NT 4.0. Требование информационной совместимости должно быть обеспечено работой с файлами геометрической информации определенной структуры в качестве входной и выходной информации.

1.6 Требования к программной документации

В разделе «Требования к программной документации» должен быть указан предварительный состав программной документации и, при необходимости, специальные требования к ней.

1.7 Технико-экономические показатели

В разделе «Технико-экономические показатели» должны быть указаны: ориентировочная экономическая эффективность, предполагаемая годовая потребность, экономические преимущества разработки по сравнению с лучшими отечественными и зарубежными образцами или аналогами.

1.8 Стадии и этапы разработки

В разделе «Стадии и этапы разработки» устанавливают необходимые стадии разработки, этапы и содержание работ (перечень программных документов, которые должны быть разработаны, согласованы и утверждены), а также, как правило, устанавливают сроки разработки и определяют исполнителей.

2 СОГЛАШЕНИЕ О ТРЕБОВАНИЯХ

Составление соглашения о требованиях — цель второй части первой лабораторной работы. Также соглашение о требованиях является вторым разделом курсовой работы.

Ниже дается описание разделов, которые должны присутствовать в любом соглашении о требованиях (СТ) согласно ГОСТу. Все последующие упоминания о соглашении о требованиях имеют в виду соглашение о требованиях, разработанное в соответствии с данным разделом. Описываемый подход предполагает также декомпозицию проекта и дальнейшую детализацию, отражаемую во внешней спецификации изделия.

Предполагается, что все утверждения, включенные в соглашение о требованиях, являются требованиями, если они не определены как цели. Различие состоит в том, что требование должно быть обязательно выполнено, а цель допускает ее достижение с некоторым приближением.

Описание информации, содержащейся в СТ, дается ниже очень подробно, и, возможно, после чтения первых абзацев появится желание ограничиться беглым просмотром. Однако если вам когда-нибудь придется разрабатывать свой собственный стандарт СТ, вы, вероятно, оцените предлагаемый здесь со всеми подробностями пример и используете его как образец. С учетом того, что сжатость заголовков разделов и большой объем их описаний скрывают порядок и простоту, лежащие в основе результирующего документа, в конце каждого пункта приводится пример оформления на конкретное изделие. Вероятно, вам опять не захочется читать все приложение целиком, поэтому рекомендуется для начала бегло просмотреть его, чтобы получить представление, как выглядит типичное СТ в целом. Затем можно будет обращаться к отдельным его разделам, чтобы иметь дополнительные примеры, иллюстрирующие приводимый материал.

2.1 Описание программного изделия

2.1.1 Наименование и шифры изделия

2.1.1.1 Полное наименование изделия

Указывается предлагаемое полное наименование изделия. После утверждения СТ не должны использоваться никакие другие наименования для данного изделия, кроме сокращенных наименований, которые приводятся ниже.

Пример. ASK (произносится «аск»).

2.1.1.2 Сокращенные наименования

Указываются все предлагаемые сокращения, которыми разрешается заменять наименование, приведенное в пункте 2.1.1.1. После утверждения СТ не рекомендуется использовать никакие другие сокращения. В противном случае делается пометка «Отсутствуют».

2.1.1.3 Шифры изделия

Указываются шифр или шифры изделия, присвоенные в соответствии с требованиями удобства управления его конфигурацией. Если предполагается выпуск печатных изданий и разработка планов поддержки, порядок присвоения шифров конечным результатам работы группы выпуска документации и группы поддержки может быть иным.

Пример. L301A.

2.1.1.4 Шифры проекта

Приводятся все шифры проекта, используемые в процессе разработки изделия.

Пример. C013.

2.1.2 Краткое описание изделия

Описываются кратко и в общих понятиях основные функциональные свойства изделия. Если программное изделие является расширением уже существующего, характеризуются только его новые свойства.

Пример. ASK позволяет специалисту по финансовому анализу или другому лицу с аналогичными аналитическими задачами в интерактивном режиме и на расстоянии запрашивать ЭВМ серии Stella 100 выполнить поиск и обработку финансовой информации из DATABASE, которая содержит фундаментальные сведения и зависимости по данным за 20 лет для большого числа корпораций и отраслей. ASK может также формировать дополнительные общедоступные или частные сведения, файлы корпораций и отраслей, выражения и элементы и использовать их вместе с информацией из DATABASE.

Программное изделие ASK в совокупности с DATABASE образует сервисную систему ASK DATABASE; все три системы являются собственностью фирмы ABC Services.

2.1.3 Сведения об авторском праве

Если предполагается заявить об установленной законом защите авторских прав на данное изделие, это должно реализоваться уже в СТ. В противном случае делается пометка «Не требуется».

Пример. Copyright © 1977 by ABC Computers Company.

2.1.4 Результирующие компоненты изделия

В данном разделе приводится таблица, подобная или эквивалентная таблице 2.1. В данном случае использована заранее подготовленная печатная форма, что уменьшает время подготовки информации и обеспечивает ее согласованность.

В указанной таблице в строке «Тип изделия» ставится метка X против соответствующей характеристики «Основное» или «Вспомогательное». Если изделие не используется для создания других изделий, оно отмечается как основное. В противном случае (например, ассемблер, компилятор или генератор) оно отмечается как вспомогательное.

В графе «Уровень поддержки» выбирается метка 1, 2 или 3 в соответствии с пояснениями в бланке.

Каждый элемент изделия отмечается меткой X в графе «Формируется целиком», если он будет создаваться заново, или в графе «Модифицируется», если будут вноситься только дополнения или изменения. Метка X ставится в графе «Распространяется» (за пределы группы разработки) или «Не распространяется» (за пределы группы разработки) в зависимости от того, что является правильным. В графе «Ответственная группа» пишется Р, И, П или С — по ключу на бланке. Если предполагается выпуск нестандартных изделий, этот факт отмечается в графе «Другие спецификации» и описание соответствующих документов дается сразу после таблицы.

Таблица 2.1 — Результирующие компоненты изделия

Обозначения: Основное изделие — не используется для создания других изделий Вспомогательное изделие — используется для создания других изделий Уровень поддержки 1: удовлетворяются заявки на исправление дефектов; возможно сообщение об изменениях; принимаются заявки на расширение функциональных возможностей изделия Уровень поддержки 2: удовлетворяются заявки на исправление дефектов; возможно сообщение об изменениях; заявки на расширение не принимаются Уровень поддержки 3: удовлетворяются заявки на исправление дефектов Р — группа разработки		Формируется целиком	Модифицируется	Распространяется	Не распространяется	Ответственная группа
	Спецификации					
	Внешняя спецификация	X			X	Р
	Внутренняя спецификация	X			X	Р
	Спецификация испытаний (не надо)					
	Спецификация сопровождения (не надо)					
	Другие спецификации					
	Документация					
	Техническое описание системы					
	Справочное руководство	X		X		Б

Окончание табл. 2.1

				Справочный буклет	Х		Х		Б
				Руководство оператора	Х		Х		Б
Тип изделия	Основное	Х	Начальный уровень поддержки	Указатель системных сообщений	Х		Х		Б
	Вспомогательное			Информационный листок выпуска					
			1	Х	Другие печатные издания					
			2		Рекламные материалы					
			3							
					Программное обеспечение					
					Листинги	Х			Х	Р
					Исходные модули	Х			Х	Р
					Объектные модули	Х				Р
					Контрольные примеры	Х			Х	Р И
					Средства разработки				Х	
					Прочие средства					

Листинги представляют собой комплект распечаток, полученных при компоновке и трансляции программного изделия.

Исходные модули содержат совокупность операторов, которые должны быть ассемблированы или откомпилированы для получения готовой программы.

Объектные модули представляют собой редактируемые или загружаемые программы на машинном языке, получаемые в результате ассемблирования или компиляции исходных модулей, или программу, которая порождается некоторым генератором (если она пригодна для непосредственного выполнения или интерпретации).

Отладочный материал используется для доказательства того, что программное изделие может быть правильно ассемблировано, откомпилировано, отредактировано, загружено и выполнено. Отладочный материал может включать листинги, исходные модули, объектные модули и процедуры.

Средства разработки включают ассемблеры, компиляторы, загрузчики, процедуры получения дампов и тому подобные средства, разработанные в рамках данного проекта с целью получения программного изделия.

2.2 Цели

Формулировка цели создания программного изделия отвечает на вопрос «зачем?». Поэтому в данном разделе указываются все причины выпуска изделия. Часто такой причиной может быть выполнение плана или договора либо устранение недостатков предшествующего изделия.

Пример. Коммерческим планом финансовых служб предусматриваются создание в фирме ABC Services и поставка на рынок интерактивной системы финансового анализа с дистанционным доступом, предназначенной для финансовых кругов. Система ASK призвана удовлетворить требования к программному обеспечению, изложенные в коммерческом плане.

2.2.1 Согласование заявок на проверку

Заявки на проверку представляют собой предписанные изменения изделия, которые являются результатом событий, находящихся вне сферы контроля разработчиков изделия. Если заявки на проверку отсутствуют, этот факт отмечается и пункты 2.2.1.1 и 2.2.1.2 опускаются.

2.2.1.1 Отклоненные заявки

Обычно такие заявки отсутствуют, поскольку внесение изменений является обязательным. Однако учет какого-либо конкретного изменения все же может оказаться нецелесообразным,

в этом случае отказ должен быть тщательно обоснован и документирован, начиная с данного раздела СТ.

2.2.1.2 Принятые заявки

Здесь перечисляются все предлагаемые в заявках изменения, которые были одобрены и будут включены в изделие. Если таких заявок нет, следует дать пометку «Отсутствуют».

2.2.2 Согласование заявок на расширение

Если заявок на расширение нет, этот факт отмечается и пункты 2.2.2.1 и 2.2.2.2 опускаются.

2.2.2.1 Отклоненные заявки

Перечисляются все предложения, касающиеся расширения изделия, которые были специально рассмотрены с точки зрения их реализации, но не могли быть приняты, указываются причины их отклонения. Если таких предложений нет, делается пометка «Отсутствуют».

2.2.2.2 Принятые заявки

Перечисляются все предложения на расширение, которые были одобрены и будут реализованы в изделии. Если таких предложений нет, делается пометка «Отсутствуют».

2.2.3 Согласование заявок на внесение исправлений

Если предметом рассмотрения СТ является новое изделие, которое не предназначается для замены другого изделия, внешних запросов на внесение исправлений быть не может. В этом случае делается пометка «Не требуется» и опускается пункт 2.2.3.1.

2.2.3.1 Отклоненные заявки

Если целью является переработка или расширение изделия либо замена изделия с известными ошибками, следует планировать исправление ошибок, обнаруженных на данный момент

времени. Поэтому в этом пункте пишется примерно следующее: «Все существенные ошибки, зарегистрированные до последнего срока подачи заявок на внесение исправлений (этап СТ), будут исправлены. Заявки, полученные в период между СТ и датой готовности продукта, будут удовлетворены по возможности». Здесь же отмечаются все существенные ошибки, зарегистрированные до появления СТ, которые не будут исправлены (как правило, в связи с технической сложностью), а также причины такого решения. По мере продвижения работы над проектом неудовлетворенные заявки на внесение исправлений могут накапливаться и фиксироваться. Если такие заявки накопятся, данный раздел СТ должен быть изменен.

2.2.4 Согласование планов

2.2.4.1 Исключенные пункты плана

Если имеются какие-либо плановые указания, требующие особых свойств и возможностей программных средств, которые не могут быть обеспечены, если изделие разрабатывается в соответствии с другими требованиями, описанными в СТ, каждое такое указание необходимо рассмотреть отдельно и пояснить, почему оно будет исключено. Если таких указаний нет, делается пометка «Отсутствуют». Отмечаются также и все другие свойства (не фигурирующие в заявках на расширение), которые были детально рассмотрены и исключены.

Пример. Широкий набор терминальных устройств, требуемый в коммерческом плане финансовых служб, не будет обеспечен. С системой ASK будет эффективно работать только устройство Telcoscope 43 или электрически и функционально эквивалентные терминалы. Рассмотрение команд графического вывода (вместо выдачи таблиц) и сортировки файлов откладывается до следующей версии.

2.2.4.2 Включенные пункты плана

Если необходимость создания изделия обоснована таким документом, как план выпуска изделия, план выпуска серии или описание задачи, то цитируется либо определенное место из ка-

ждого документа, либо приводится краткое содержание соответствующих разделов. Плановые указания устанавливают (в самых общих чертах), что должно делаться и почему. Эти указания отличаются от технических обоснований, определяющих в конкретных понятиях технические предпосылки осуществления того, что должно делаться. Для каждой цитаты делается отсылка к соответствующей записи в разделе 2.4.1 СТ.

Пример. Коммерческий план финансовых служб, раздел 5 (см. п. 2.4.1 а).

2.2.5 Перечень требований пользователя

Указываются заказчики изделия и поясняется, почему оно им необходимо. В этом разделе указывается также предполагаемый срок использования изделия. Обычно это будет срок службы оборудования или время до выпуска изделия-преемника. Если данное изделие должно быть заменено, указывается наименование изделия-преемника. Цель этого раздела — дать некоторое представление о степени сложности условий работы программ.

Пример. Система ASK предназначается для специалистов по финансовому анализу или других лиц с аналогичными аналитическими задачами, как, например, управляющий по контролю и регулированию портфеля заказов. Ожидается, что пользователи не знакомы с программированием и не имеют подготовленных операторов терминалов.

В соответствии с документом, указанным в пункте 2.4.1 а, первая версия ASK должна быть готова для продажи через 6—12 месяцев, а вторая версия — не позже, чем через 18 месяцев.

2.2.6 Рассмотренные альтернативы

Кратко описываются альтернативы данной разработки, которые были рассмотрены и отклонены, а также причины отклонения. Если программы должны быть закуплены, поясняется, почему они не могут быть разработаны собственными силами, и наоборот. Лица, критически анализирующие СТ, могут потребовать весомых доказательств необходимости разработки про-

грамм, поэтому следует всегда убеждаться, что в данном разделе имеется такой материал.

Пример. Поскольку в распоряжении фирмы ABC Services нет ни одного программного изделия, которое может выполнять функции ASK, должно быть разработано новое изделие. Действующих аналогов в готовом виде не существует. Фирма ABC Services решила заключить единственный договор с фирмой ABC Computers о создании и сопровождении ASK; это решение основывается на прошлом положительном опыте заключения договоров с фирмой ABC Computers.

2.2.7 Окупаемость капиталовложений

Определяется прибыль, которую даст создание изделия, в понятиях, соответствующих целевому назначению организации.

Пример. Фирма ABC Services ожидает, что объем сбыта в финансовой сфере увеличится на 10% в течение 3-х месяцев с момента выпуска изделия и достигнет примерно 170% в течение первого года с момента выпуска. В результате ожидаемой большой прибыли от такого увеличения объема сбыта затраты на разработку изделия ASK окупятся через 8 месяцев после выпуска, что без новой версии ASK даст полную валовую прибыль, в три раза превышающую суммарные затраты на его разработку и сопровождение.

2.3 Стратегия

Формулировка стратегии предполагает ответ на вопрос «что?». Поэтому в данном разделе указывается, что будет представлять собой предлагаемое изделие.

2.3.1 Соглашения относительно представления материала

Для краткого описания изделия может понадобиться специальная терминология. Данный раздел предназначен для того, чтобы представить читателям СТ специальный словарь. И если в

пунктах 2.3.1.1 и 2.3.1.2 внешней спецификации можно делать дополнения, то в данном разделе СТ ни одна формулировка не должна меняться.

2.3.1.1 Обозначения

Определяются все обозначения, используемые в СТ. Например, если применяются индексы, дается пример их использования и определяется принцип индексации. В противном случае делается пометка об отсутствии специальных обозначений.

2.3.1.2 Терминология

Четко определяется вся терминология, которая может оказаться специфической для данного изделия.

Пример. Вся специальная терминология определяется в контексте данного документа.

2.3.2 Генерируемое программное обеспечение

Генерируемое программное обеспечение — это то, что получается на выходе компилятора, ассемблера, загрузчика, редактора связей или генератора прикладных программ. Генерируемое программное обеспечение классифицируется как вспомогательное и порождается изделием, описываемым в СТ.

2.3.3 Системное программное обеспечение

Системное программное обеспечение — это все остальное программное обеспечение, включающее операционные системы, компиляторы, утилиты, пакеты прикладных программ и др. Это программное обеспечение относится к основному типу и является изделием, описываемым в СТ.

Разделы 2.3.2 и 2.3.3 строятся одинаково. Там, где слова «программное обеспечение» или «изделие» появляются без определений, они относятся как к генерируемому, так и к системному программному обеспечению. Выражение (a, b) означает a или b. Если это выражение появляется дважды, например (a_1, b_1)… (a_2, b_2), выбирается a_1 и a_2 или b_1 и b_2.

Поскольку большинство программных изделий являются основными, может появиться желание поменять местами разделы 2.3.2 и 2.3.3, а затем опустить раздел 2.3.3 для основного изделия. Причина выдвижения здесь генерируемого программного обеспечения на первое место состоит в том, что при правильном проектировании сверху вниз генерируемое программное обеспечение является основной целью проектирования и должно быть описано раньше, чем его генератор. Другими словами, структура генерируемых программ должна определять структуру генератора, а не наоборот. Если изделие является основным, под заголовком «2.3.2. Генерируемое программное обеспечение» делается пометка «Не используется» и опускаются пункты 2.3.2.1—2.3.2.3.

2.3.3.1 Общие характеристики функций

Необходимо рассматривать все изделие как один функциональный модуль, чтобы число подразделов было небольшим. Если невозможно адекватно описать изделие без разбиения его на отдельные функциональные модули, следует дать схему, показывающую, как связаны между собой функциональные модули, и присвоить каждому модулю собственное обозначение. Затем для каждого функционального модуля отводится подраздел раздела 2.3.(2,3), в заглавии которого используется слово «функция» с последующим именем функционального модуля (рис. 2.1).

Рис. 2.1 — Структурная схема из соглашения о требованиях для изделия ASK

Пример.

2.3.3.1 Общие характеристики функций ASK.

2.3.3.1.1 Внешние ограничения ASK.

Отметим, что здесь выполняется именно функциональная декомпозиция, поскольку СТ является функциональным документом и не указывает, как предлагаемое изделие будет физически разбито на модули. Во внутренних спецификациях, создаваемых на основе СТ, обязательно должно быть описано физическое разбиение на модули. Чтобы было легче сопоставлять внутренние спецификации с предшествующими им соглашениями о требованиях, удобно представить какое-либо конкретное физическое разбиение и попытаться определить функциональные модули, которые могут быть реализованы как физические модули. Но, поступая так, следует помнить, что в СТ более важным является четкое разбиение по функциям, а физическое разбиение только подразумевается, но не диктуется СТ.

Разделы 2.3.(2,3).X организуются по иерархическому принципу, чтобы охватить как можно больше вопросов в первой группе подразделов. Это позволяет при отсутствии новой существенной информации просто ссылаться на подразделы более высокого уровня, как это сделано в разделе 2.3.3.2 для системы, показанной на рисунке. Здесь и далее X — номер модуля системы. В рассмотренном примере X = 1 для общих функций ASK, X = 2 для интерфейса пользователя и X = 3 для процессора корректировок. В других проектах количество и состав модулей могут различаться.

2.3.3.1.1 Внешние ограничения

Перечисляются все ограничения, сфера действия которых шире, чем сфера действия СТ; сюда входят, например, промышленные ограничения или ограничения, касающиеся серии изделий. Может быть разрешено введение дополнительных ограничений во внешней и внутренней спецификации, сфера действия которых ограничена рамками данного изделия, например ограничения на распределение памяти.

В разделе 2.3.(2,3).1.1 перечисляются все возможные ограничения, относящиеся ко всем функциям. Однако, если список

оказывается длинным и некоторые ограничения относятся не ко всем функциям, они должны приводиться по ходу изложения как можно раньше.

2.3.3.1.1.1 Действующие стандарты

Перечисляются все промышленные стандарты и собственные технические условия организации, которые должны быть учтены при изготовлении изделия. При этом следует убедиться, что ссылки на эти стандарты имеются в разделе 2.4.1 соглашения о требованиях.

Пример. 2.3.3.1.1.1. Действующие стандарты: Стандарт АВС на программирование (см. п. 2.4.1 д).

2.3.3.1.1.2 Ограничения на совместимость

Всегда должно рассматриваться несколько аспектов совместимости: исходный язык, машинный язык, форматы данных и сообщений, форматы отчетов, форматы листингов и форматы языка управления заданиями (управляющие карты, структура команд и т.п.). Специально должна оговариваться совместимость со следующими программными изделиями:

– изделиями-предшественниками, т.е. такими, которые пользователь может заменить новым изделием; если в новом изделии отсутствует какая-либо возможность, которую имеет предшественник, должны быть приведены обоснования ее исключения;

– изделиями-компаньонами, относящимися к той же группе средств, которые являются альтернативами нового изделия; примером может служить программа задания выходного формата, которая отличается от другой аналогичной программы только самим форматом;

– подобными изделиями, т.е. такими, которые выполняют похожие функции в других программных изделиях той же серии; такими изделиями являются, например, программа сортировки с использованием диска и программа ленточной сортировки;

– конкурирующими изделиями, которые выполняют те же функции, но поставляются другими организациями, например

Кобол фирмы IBM является конкурентом Кобола фирмы Burroughs.

В каждом случае должно указываться, является ли данное изделие расширением или частным случаем другого продукта, и должна даваться соответствующая ссылка в разд. 2.4.1 СТ. Описываются также условия получения данного изделия из других версий или других изделий.

Пример. Не существует программных изделий или баз данных, совместимых с системой ASK. Файлы, генерируемые системой ASK, будут VSOS-файлами прямого доступа (см. п. 2.4.1 б) и поэтому могут быть непосредственно использованы другими программами.

2.3.3.1.1.3 Программные ограничения

Указывается, если это необходимо, операционная система, с которой должно работать предлагаемое программное изделие, а также другие программные средства, с которыми оно должно стыковаться в процессе работы. Следует убедиться, что указаны все средства, необходимые для загрузки программного изделия либо передачи управления ему или от него.

Если имеются программы, которые должны быть исключены из операционной системы, необходимо указать их и объяснить, почему они исключаются.

Пример. ASK работает с системой VSOS версии 4 (см. п. 2.4.1 б). Любой компонент семейства VSOS может работать одновременно с ASK столько времени, сколько будет доступна конфигурация устройств, описанная в пункте 2.3.3.1.1.4. Процессор корректировок может работать параллельно с интерфейсом пользователя, но интерфейсу пользователя закрыт доступ к файлу, который используется в данное время процессором корректировок.

2.3.3.1.1.4 Аппаратные ограничения

Приводится таблица устройств, используемых при работе программного изделия. Для каждого устройства указывается минимальное, номинальное и максимальное требуемое число. Номинальным является оптимальное число устройств или наи-

более вероятное, если оптимум не определен. В случае необходимости каких-либо дополнительных устройств они должны быть отмечены отдельно.

Пример. Помимо устройств, требуемых системами, указанными в пункте 2.4.1 б, для системы ASK потребуются устройства, перечисленные в таблице 2.2.

Таблица 2.2 — Необходимые для работы ASK устройства

Устройство	Минимальное число	Номинальное число	Максимальное число
M103 Блок операций с плавающей точкой	1	1	1
M107 Резервный блок питания	1	1	1
M1100 Модуль памяти	1	2	2
M3100 Дисковый модуль	1	3	8
M210 Пульт	1	1	1
Терминал Telcoscope 43	1	128	256

2.3.3.1.2 ВНЕШНИЕ ХАРАКТЕРИСТИКИ

Информация данного раздела должна быть расширена (но не изменена) во внешней спецификации, а в СТ должны быть представлены лишь самые основные ее аспекты. Если предлагаемое изделие является расширением какого-либо существующего изделия, описываются главным образом его дополнительные характеристики. Этот раздел должен иметь иерархическую структуру, в которой, прежде всего, должно уделяться внимание наиболее важным для конечного пользователя вопросам.

2.3.3.1.2.1 РЕЗУЛЬТАТЫ РАБОТЫ

Описываются все выходные данные программного изделия или функционального модуля с точки зрения их содержания и назначения — отчеты, файлы, записи, поля данных, сообщения, таблицы, флажки. Должны быть рассмотрены все возможные носители и средства отображения информации, в том числе программно опрашиваемые индикаторы, каналы прерывания, панельные индикаторы и т.п.

Пример. Записи в общедоступных и частных файлах, являющихся VSOS-файлами прямого доступа (см. п. 2.4.1 б). Записи в DATABASE для ведения системы индексно-последовательных файлов (см. п. 2.4.1 б) под управлением VSOS (см. п. 2.4.1 б). Поля и записи данных, передаваемые на терминалы и на пульт. Все результаты должны выдаваться под контролем модуля логического доступа VSOS.

2.3.3.1.2.2 Процессы обработки

Описываются операции, выполняемые программным изделием, которое при этом рассматривается в целом или по функциональным модулям как черный ящик (или совокупность черных ящиков). Как минимум, устанавливается соответствие входной и выходной информации, а если обсуждение идет на уровне модулей или этапов обработки, указываются также модули или этапы, которые требуются для получения определенной выходной информации. Определяются все параметры, необходимые для генерации и выполнения программ.

Пример. Рабочий режим состоит в интерактивном поиске, вычислении, выдаче сообщений и построении диаграмм на запросы специалистов по финансовому анализу.

Режим обслуживания состоит в интерактивном анализе базы данных, ее реорганизации и корректировке, производимых системными аналитиками, прошедшими курс обучения обслуживанию баз данных.

2.3.3.1.2.3 Входы системы

Описываются входные данные так же, как результаты работы в пункте 2.3.3.1.2.1.

Пример.

1. Блоки записей из DATABASE и файлов корректур DATABASE (а также файлы системы VSOS).

2. Записи из общедоступных и частных файлов системы ASK.

3. Поля данных с клавиатуры терминалов и пультов.

4. Все входные данные контролируются модулями логического доступа VSOS.

2.3.3.1.3 Эргономические характеристики

Эргономическими характеристиками изделия являются такие свойства, которые обеспечивают надежность, комфорт и продуктивность работы пользователей и операторов. В данном разделе описываются прикладные аспекты эргономики применительно к данному изделию и определяется, насколько разнообразным будет режим его работы.

2.3.3.1.3.1 Безопасность и секретность системы

Описываются в общих чертах характеристики изделия, обеспечивающие безопасность и секретность данных и программ.

Пример. Никакие общедоступные файлы, в том числе DATABASE, не могут быть записаны с терминала. Все секретные файлы защищены паролем в соответствии с соглашениями VSOS. Все операции записи в общедоступные файлы, включая DATABASE, должны выполняться с пульта.

2.3.3.1.3.2 Надежность

Под надежностью программных средств понимается способность к восстановлению нормальной работы при ошибках и сбоях в работе оборудования. Первостепенную важность имеет защита данных пользователя. Следует указать, могут ли ошибки в данных быть исправлены оператором, или определить, как должна продолжаться обработка после обнаружения ошибки, а также установить степень потери информации в различных ситуациях, включая сбои в работе оборудования.

Затем определяется степень защиты программ от ошибок, возникающих в других частях системы. Если некоторая программа должна работать в мультипрограммном режиме, необходимо рассмотреть возможные ошибки в других программах наряду с ошибками в работе оборудования. Если мультипрограммный режим не предусматривается, этот вопрос затрагивать не нужно. В противном случае необходимо рассмотреть защищенность от ошибок, которые возникают в программах, находящихся в памяти одновременно с данной программой.

Следует рассмотреть, как могут повлиять на работу предлагаемых программных средств ошибки в этих сопутствующих программах. При этом учитываются следующие моменты:

а) Операции распределения ресурсов, при которых другим программам может выделяться некоторая область памяти на том же самом устройстве или в той же его части, что и для данной программы. Следует указать, как программные средства изолируются друг от друга во избежание перекрытия отводимых им областей памяти. Если такая изоляция обеспечивается операционной системой или загрузчиком, достаточно сослаться на эти возможности.

б) Ошибки выполнения других программ. Здесь также можно ограничиться указанием соответствующих возможностей (если таковые имеются) аппаратных средств и операционной системы.

Пример. Многие ошибки фиксируются системой ASK, регистрируются в файле ошибок и обрабатываются соответствующими средствами восстановления, имеющимися в составе VSOS.

ASK не изменяет базы данных DATABASE; для псевдокорректировки базы данных используются виртуальные файлы.

ASK подчиняется всем ограничениям интерфейса VSOS и поэтому не вызывает сбоя в работе параллельно функционирующих программных средств, и наоборот, параллельно работающие средства, которые также подчиняются соглашениям, не вызовут сбоя в работе ASK.

Каждый блок записей в DATABASE содержит контрольную сумму, которая вычисляется и сравнивается системой ASK для проверки целостности данных.

Виртуальный телекоммуникационный метод доступа (VTAM) системы VSOS используется на логическом уровне 4, наивысшем уровне восстановления при ошибках (который регистрирует все сбои оборудования, допускающие восстановление).

2.3.3.1.3.3 РЕСТАРТ

Указываются возможности, обеспечивающие сохранение и использование данных при возобновлении работы после ава-

рийного прерывания, например при рестарте из контрольной точки.

Пример 1. Программы, генерируемые ASK, могут запускаться с любого этапа обработки данных.

Пример 2. Модуль SORT обеспечивает автоматическую или ручную регистрацию параметров и данных, необходимых для рестарта, из последней или любой предыдущей контрольной точки. Вся необходимая для этого информация при выключении электропитания сохраняется, чтобы отказы блока питания не влияли на рестарт.

2.3.3.1.3.4 Соответствие требованиям заказчика

Указываются свойства, которые позволяют программному изделию или его выходным данным удовлетворять конкретным требованиям. Перечисляются, если это возможно, модули, которые могут не удовлетворять требованиям заказчика. Указывается также, какие ресурсы машинного времени и памяти резервируются для учета требований заказчика и какие средства предусматриваются для обслуживания выходных каналов пользователя.

Пример. ASK характеризуется параметрами времени компоновки и времени запуска, позволяющими определять требуемую конфигурацию вычислительной системы. Параметры компоновки определяются для различного числа модулей памяти и дисковых модулей, а также для разного количества и типов контроллеров связи. Параметры запуска определяются для различного числа линий и терминалов, а также для различных структур идентификаторов терминалов и паролей. В систему встроены алгоритмы для оптимизации распределения дисковой памяти и для оценки ситуации, в соответствии с которой следует просто подтвердить прием команды вместо немедленного ее выполнения. Обеспечиваются служебные программы для корректировки, анализа и реорганизации общедоступных файлов ASK и файлов пользователя.

2.3.3.1.3.5 Рабочие характеристики

Приводится основная переменная или основной принцип, по которому должна измеряться эффективность работы про-

граммы; указывается соответствующее значение или диапазон значений для этой переменной. Главным здесь является установление количественной меры. Нельзя, например, писать «скорость работы генерируемой программы будет соответствовать быстродействию программы, написанной вручную на ассемблере». Такое утверждение нельзя проверить или использовать при принятии решений относительно внутренней структуры программного изделия. Фраза должна быть определенной, например «производительность будет составлять, как минимум, 21 сообщение в 1 минуту». Характеристики, включенные в данный раздел, должны быть доступны для измерения пользователем. Ими могут быть быстродействие, пропускная способность, скорость передачи данных, скорость компиляции, генерации или ассемблирования, расход машинных ресурсов, время реакции и т.п.

Пример 1. Каждое периферийное устройство должно работать с максимальной производительностью в предположении отсутствия других параллельных операций.

Пример 2. Для 16 активных терминалов будет обеспечиваться время реакции в пределах 5 секунд и меньше по 90% всех тривиальных операций взаимодействия, где тривиальное взаимодействие определяется как передача с терминала односимвольного запроса, поиск на диске односимвольного ответа и прием односимвольного ответа на том же терминале. Время ответа не будет увеличиваться более чем на 100% (до 10 секунд) при увеличении числа активных терминалов с 16 до 180.

Пример 3. Изделие не накладывает никаких ограничений на конфигурацию, помимо ограничений, определяемых оборудованием.

2.3.3.1.3.6 Удобство эксплуатации

Описываются свойства, которые делают взаимодействие «человек — машина» удобным для человека. Примерами являются свободный формат входных данных, диалоговый режим, синтаксическая совместимость, возможность ввода сокращенных команд и т.д. Если существуют другие программные средства, с которыми предполагаемые пользователи могут быть знакомы и которые имеют идентичные или похожие операции,

операторы, команды, сообщения об ошибках или другие структуры, следует указать, что соответствующие структуры предлагаемого программного изделия будут такими же, когда они одинаковы, или похожими, когда они сходны. Учитываются также состав специалистов, требуемых для использования тех или иных компонентов программного изделия, и уровень их квалификации, например специалисты в области сбыта, системные программисты, операторы. Необходимо стремиться использовать простой стандартный способ выделения уровней квалификации, например высокий (семь и более лет работы), средний (от двух до шести лет работы) или низкий (один год и менее).

Должно быть показано, как начинается и как заканчивается работа программного изделия. Для таких функций, как сбор, ввод и вывод данных, связь, вычисление, компиляция или генерация, необходимо указать соответствующие типы режима работы:

- пакетный;
- интерактивный;
- приоритетный;
- фоновый;
- режим интерпретации;
- диагностический;
- непрерывный;
- с прерываниями.

2.3.3.1.3.7 Мобильность

Описываются требования и цели обеспечения переноса программного изделия из одних рабочих условий в другие.

Пример. В ASK соблюдаются все условия интерфейса VSOS и используется модуль логического доступа VSOS для всех операций ввода-вывода. Поэтому ASK может работать без модификаций с любой операционной системой, для которой VSOS является подсистемой.

2.3.3.1.4 Внутренние характеристики

2.3.3.1.4.1 Удобство сопровождения

Описываются меры, гарантирующие идентифицируемость модулей, если этот вопрос не решен с помощью стандарта.

Пример 1. Каждый исходный и объектный модуль будет снабжаться шифром программного изделия и категорией выпуска, а также, если позволяет место, шифром проекта и идентификатором изделия. Эта информация должна располагаться таким образом, чтобы каждый модуль можно было идентифицировать на любом диске, ленте или в дампе памяти.

Описываются все встроенные средства отладки и компоновки, включая сопряжения с мониторами; указывается, возможно или нет их выборочное отключение. Чтобы можно было позднее оценить удобство сопровождения, следует указать языки, с которыми способно работать данное программное изделие, а также затраты машинных ресурсов на исправление ошибок.

Пример 2. В системе ASK используются средства программирования BIL3 (см. п. 2.4.1, е), предусматривающие коэффициент резервирования памяти, равный 10%. В соответствии с разделом 2.1.4 пользователям будут передаваться объектные программы. Для коррекции объектных программ используется программа UPDATE (см. п. 2.4.1, ж).

Указывается, какие лица, не входящие в группу сопровождения, могут вносить изменения. При этом необходимо определить условия их подготовки и требуемые ресурсы.

Пример 3. Инженер, являющийся специалистом по системам и успешно освоивший базовую версию VSOS, сможет затрачивать на внесение изменений в объектный модуль не более 2 часов своего времени и 0,5 часа машинного времени.

Следует убедиться в том, что усилия, затрачиваемые на достижение удобства сопровождения, согласуются со временем жизни изделия, определенным в разделе 2.2.4 СТ.

2.3.3.1.4.2 Алгоритмы

Если какие-то используемые алгоритмы или методы играют особую роль в обеспечении успеха или неудачи изделия в смысле его надежности и требуемых характеристик, отмечаются принимаемый риск или ожидаемые преимущества. В противном случае констатируется, что они подлежат описанию во внутренней спецификации.

2.3.3.2 Интерфейс пользователя

2.3.3.2.1 Внешние ограничения

2.3.3.2.1.1 Стандарты для интерфейса пользователя

Пример. ANSI X3.28-1971 (см. п. 2.4.1, г).

2.3.3.2.1.3 Программные ограничения на интерфейс пользователя

Пример. Необходимо наличие модулей VSOS DAM, ILSAM и VTAM (см. п. 2.4.1, б).

2.3.3.2.1.4 Аппаратные ограничения на интерфейс пользователя

Пример. Помимо устройств, указанных в разделе 2.3.3.1.1.4, интерфейсу пользователя требуются минимальные конфигурации, необходимые для VSOS ILSAM, DAM и VTAM (см. п. 2.4.1, б).

2.3.3.2.2 Внешние характеристики

2.3.3.2.2.1 Результаты работы интерфейса пользователя

Пример. Результаты такие же, как в разделе 2.3.3.1.2.1, исключая записи в DATABASE.

2.3.3.2.2.2 Процессы интерфейса пользователя

Пример. Интерфейс пользователя дает возможность:

– формировать критерии выбора и (или) зависимости;

– создавать файлы фирм или отраслей;

– добавлять данные в файлы;

– сортировать данные;

– выдавать сообщения, строить диаграммы и/или сохранять результаты.

2.3.3.2.2.3 Входы интерфейса пользователя

Пример. Такие же, как в разделе 2.3.3.1.2.3, кроме данных из файлов корректировки DATABASE.

2.3.3.2.3 Эргономические характеристики

2.3.3.2.3.3 Рестарт интерфейса пользователя

Пример. Состояние системы для всех активных пользователей (в том числе отключенных, но еще обслуживаемых) периодически запоминается на диске (с интервалом, оговариваемым в рамках определения времени компоновки). Обеспечиваются ав-

томатическая и ручная процедуры рестарта на основе использования этих данных.

1. Автоматический рестарт. При использовании вспомогательного блока питания M l07 интерфейс пользователя обеспечивает автоматическое восстановление питания. Все пользователи, активные в момент отключения питания, оповещаются о сбое. Эти активные пользователи опрашиваются и по их желанию либо выводятся из работы, либо их обслуживание продолжается, начиная с последней контрольной точки. Для неактивных пользователей восстановление не производится; работа с ними прекращается, за исключением случаев запоминания результатов выполнения команд, которые были выданы перед отключением питания. Такая же процедура выполняется и после любого другого сбоя, если это может быть сделано достаточно надежно.

2. Ручной рестарт. В каждой контрольной точке производится полный дамп памяти интерфейса пользователя. При каждом старте интерфейса пользователя оператор опрашивается с целью выяснения, выполнять ли новый старт или осуществлять пуск с контрольной точки. Если выбирается пуск с контрольной точки, интерфейс пользователя загружается из файла контрольной точки и вызывается процедура восстановления при сбое питания. Если же возникает сбой, при котором не может быть надежно инициирован автоматический рестарт, на пульт посылается диагностическое сообщение и оператору системы VSOS дается возможность предпринять ручной рестарт.

2.3.3.2.3.5 Характеристики интерфейса пользователя

Пример. При допущении, что на вычислительной машине выполняется только ASK и что параметр восстановления характеризуется одной контрольной точкой в 1 минуту, каждая команда должна выполняться или подтверждаться не более чем за 5 секунд с момента ее ввода (при среднем значении 3 секунды). Все команды, подтверждаемые, но не исполняемые при первой реакции, должны выполняться в течение 2 секунд (на каждый элемент данных по одной фирме и за один период).

В контексте данного раздела отметка «Выполнено» означает, что начался вывод данных на терминал, но команда при этом может быть отработана не полностью. Весь вывод на терминал должен выполняться со скоростью не менее 2 строк в секунду при проектной скорости 200 визуальных символов в секунду (при надлежащем использовании возможности табулирования).

2.3.3.2.3.6 Область применимости интерфейса пользователя

Пример. В типичном сеансе с ASK пользователь, не имеющий опыта программирования, подключается к системе с помощью терминала и вступает в диалог, в котором он определяет:

– интересующие его отрасли промышленности и фирмы;
– типы сравнений, которые он хочет выполнить;
– критерии, которые он хочет использовать для представления и сортировки данных;
– сообщения и диаграммы, которые ему нужны.

ASK будет отвечать на каждую команду либо диагностическим сообщением, в котором указываются ошибки и несоответствия в данных, вводимых пользователем, либо ответом на команду, либо подтверждением, что система начинает выполнение команды. Последний случай возникает, когда генерация ответа требует так много времени, что пользователь может предположить какую-либо неисправность, если не получит подтверждения (например, когда вводится команда создания файла). Когда ответ на такую команду готов, ASK посылает его на терминал, если только пользователь не выключился из работы. Если же пользователь уже отключился, ASK сохраняет результат и информирует пользователя о готовности данных при следующем подключении.

Доступны два типа информации из файлов: табличный и графический. Когда объем запрошенной информации превышает емкость экрана устройства Telcoscope 43, данные автоматически разбиваются на страницы. Терминалы оборудованы печатающими устройствами, способными печатать страницы по выбору.

2.3.3.2.4 Внутренние характеристики

2.3.3.2.4.2 Алгоритм интерфейса пользователя

Пример. ASK выполняет каждую команду в режиме интерпретации и немедленно; таким образом, накопление команд не разрешается (за исключением команд запоминания, которые будут рассмотрены ниже).

В системе используется концепция виртуального файла с сохранением дисковой памяти и присущих ей характеристик извлечения данных. Когда пользователь будет описывать некоторый файл как более высокий уровень какого-либо существующего файла (определенного пользователем или системой), никакие существующие данные не будут храниться в этом новом файле. Если в конце сеанса работы на терминале пользователь захочет сохранить такие созданные во время сеанса файлы, выполнение команды запоминания приведет к копированию и сохранению всех элементов данных. Если непосредственно за этой командой последует команда выхода из сеанса, команда запоминания будет выполняться после выхода, освобождая тем самым пользователя и терминал для другой работы.

Программа общего синтаксического анализа просматривает каждую запись, чтобы выполнить ее проверку по диапазону значений, типу и количеству символов и т.п. перед передачей управления процессору команд или любому другому процессору входа. Это дает возможность пользователю сразу корректировать ошибки во вводимой информации.

2.3.3.3 Функция «Процессор корректировок»

Для всех пропущенных подразделов см. соответствующие подразделы раздела 2.3.3.1.

2.3.3.3.1 Внешние ограничения

2.3.3.3.1.3 Программные ограничения для процессора корректировок

Пример. Требует только VSOS ILSAM.

2.3.3.3.1.4 Аппаратные ограничения

Пример. Помимо устройств, нужных для VSOS ILSAM (см. п. 2.4.1, б и в), процессору корректировок потребуются устройства, перечисленные в таблице 2.3.

Таблица 2.3 — Устройства, необходимые процессору корректировок

Устройство	Минимальное число	Номинальное число	Максимальное число
M103 Блок операций с плавающей точкой		1	
M107 Резервный блок питания		1	
M1100 Модуль памяти		2	
M3100 Дисковый модуль		3	
M210 Пульт		1	

2.3.3.3.2 Внешние характеристики

2.3.3.3.2.1 Результаты работы процессора корректировок

Такие же, как в разделе 2.3.3.1.2.1, за исключением записей в общедоступные и частные файлы и данных, передаваемых на терминалы.

2.3.3.3.2.2 Входы процессора корректировок

Такие же, как в разделе 2.3.3.1.2.3, за исключением общедоступных и частных файлов и данных, поступающих с терминалов.

2.3.3.3.3 Эргономические характеристики

2.3.3.3.3.3 Рестарт процессора корректировок

Указываются возможности контроля файлов, предусмотренные в программе UPDATE (см. п. 2.4.1, ж) и позволяющие иметь контрольные точки для рестарта.

2.3.3.3.3.5 Характеристики процессора корректировок

Используются концепции программы UPDATE, и заимствуется оттуда максимально возможное количество программ. Процессор корректировок должен работать с не меньшей скоростью, чем программа UPDATE, в случае сравнимых операций.

2.3.3.3.3.6 Использование процессора корректировок

В этой части процессор корректировок также базируется на UPDATE и будет иметь по возможности максимально похожий интерфейс оператора.

2.3.4 Внутренние ограничения

Важно определить не только то, каким будет изделие, но также и каким оно не будет. Ограничение — это свойство (или возможность), которое пользователю логично ожидать, но которое по тем или иным причинам исключено. Перечисляются все ограничения, упоминаемые в различных разделах СТ. Включаются также ограничения, не упоминаемые в СТ, но касающиеся таких свойств или возможностей, которые пользователь справедливо ожидает найти и исключение которых может вызвать его неудовлетворенность изделием. Не следует скрывать такие ограничения, как неполная взаимозаменяемость носителей, отсутствие поддержки для каких-либо возможностей оборудования, вынужденная имитация некоторых дополнительных устройств, невозможность одновременной работы или нахождения в памяти с изделиями-компаньонами. Как потенциальные перечисляются все такие ограничения, которые могут быть оставлены на усмотрение группы разработки.

Пример. Структура ASK предполагает, что пользователь захочет использовать максимально возможное число команд в интерактивном режиме и что «команды выдачи отчетных документов будут наиболее часто используемыми». Поэтому он должен формировать все критерии, зависимости и файлы до выдачи команд, которые их используют, чтобы избежать нарушения дисциплины ожидания для продолжительных процедур, выполняемых в процессе анализа. См. также раздел 2.3.3.2.1.3.

2.4 Используемые материалы

Этот раздел должен быть расширен в спецификации сопровождения, поэтому в СТ содержится только один подраздел.

2.4.1 Справочные документы

Отдельно указывается каждый плановый или технический документ, на который имеется ссылка в СТ. Каждый такой документ должен реально существовать (а не подразумеваться в

будущем) и должен быть зарегистрирован и принят на хранение группой контроля документации в целях управления конфигурацией программного изделия. Чтобы избежать неточностей, ссылки на документы, описывающие программы, должны содержать шифры проектов и шифры изделий.

Если в основе изделия лежит какой-либо стандартный язык или код связи, следует указать документ, определяющий этот стандарт. Если выпускается новая версия существующего программного изделия, должны быть перечислены все соответствующие ему проектные документы и публикации. Если использование программного изделия связано со специальным оборудованием, необходимо указать соответствующую документацию.

Пример. Справочные документы:

а) Коммерческий план финансовых служб, Дж. Э. Очинклосс, 13.6.77 (проект), разд. 5;

б) Интерфейс операционной системы с виртуальной памятью. Руководство, 12-6643-43;

в) Спецификация содержания и формата DATABASE, 1230711, редакция 7.2.77;

г) Американский национальный стандарт. Процедуры для использования управляющих символов стандартного кода обмена информацией в специальных каналах передачи данных, ANSI X3.28-1971, разд. 5.2, 5.4, 5.7;

д) Стандарт корпорации ABC на программирование, утвержденный 14.2.73 либо в последней редакции;

е) Язык BIL 3. Справочное руководство, 07-5411-67;

ж) Модуль UPDATE. Руководство оператора, 06-4160-36.

2.5 Передача заказчику и ввод в действие

2.5.1 Средства защиты права собственности на изделие

Указывается один из следующих уровней:

- засекречивание не требуется;
- промышленный секрет;
- авторское право;
- патент.

2.5.2 Ресурсы, обеспечивающие ввод в действие

Определяются ресурсы, требуемые для установки системы, наряду с ресурсами, описанными в разделе 2.5.3 (здесь имеются в виду машинное время, трудозатраты и необходимая квалификация персонала). Объемы и качество этих ресурсов должны быть определены в терминах наиболее вероятных потребностей.

Пример. Любой оператор, знакомый с системой VSOS и имеющий опыт работы 6 месяцев (или при эквивалентном обучении), сможет осуществить ввод в действие изделия ASK, используя модуль UPDATE, в течение 15 минут работы за пультом машины. После получасового ознакомления с процедурами верификации по информационному листку выпуска он сможет выполнить процедуру проверки в течение 10 минут работы в интерактивном режиме за терминалом.

Все сказанное предполагает, что необходимые устройства работают нормально и на машине не выполняются параллельные работы. Указываются все условия (в том числе технические и программные средства), необходимые для генерации и ввода программного изделия в действие и не описанные в разделах 2.3.(2,3).X.1.4.

2.5.3 Носители информации

Определяется тип запоминающих устройств для всех распространяемых компонентов программного изделия (например, магнитная лента, характеризуемая количеством дорожек и плотностью записи; пакет дисков, отдельные диски и т.п.). Если данное программное изделие должно работать совместно с другими изделиями, то последние должны быть названы либо должна даваться ссылка на соответствующее СТ. Необходимо убедиться в том, что требуемые формы представления данных и запоминающая среда позволяют осуществить ввод программного изделия в действие при наличии любой минимальной конфигурации устройств.

Пример. Объектные программы ASK будут распространяться на дисках по формату UPDATE (см. п. 2.4.1, ж). Исправления объектных программ будут распространяться в том же виде.

2.6 Тактика

Тактика определяет, каким образом будет реализовываться стратегия. Следовательно, в этом разделе говорится о том, как должно создаваться программное изделие.

2.6.1 Взаимосвязи

2.6.1.1 Требуемые взаимосвязи

Определяются требования, выдвигаемые данным программным изделием к другим проектам или функциям. Дается краткая характеристика каждого требования и указывается этап, на котором может быть установлен факт выполнения поставленного условия.

Пример 1. Отдел электронных интерфейсов должен обеспечивать проверку каналов с помощью диагностической программы, которую группа испытаний должна иметь на этапе О10 (раздел 7).

Фирма ABC Services должна обеспечить доступ к нормально функционирующей минимальной конфигурации ЭВМ серии Stella 100 в промежутке между этапами Р20 и ПЗО.

Пример 2. Интерфейс Electronics должен обеспечивать разветвление канала, используя диагностическую программу, которую группа испытаний должна иметь на этапе О10 (см. раздел 7).

Пример 3. На этапе РЗО необходимо иметь компилятор РПГ 11, настроенный на выполнение объектных программ.

2.6.1.2 Обеспечиваемые взаимосвязи

По структуре этот раздел аналогичен предыдущему, но содержит требования, налагаемые другими изделиями на данное изделие. Каждому требованию в разделе 2.6.1.2 должно соответствовать требование в разделе 2.6.1.1 со стороны другого изделия. Здесь же описывается влияние, оказываемое данным изделием на другие функции. Указываются все требования, которым должно соответствовать данное изделие, чтобы обеспечить работу других программных средств. Примером могут служить требования к обеспечению диагностики или сопряжению с ди-

агностическими испытательными средствами, такими, как файл ошибок или средства профилактического контроля в режиме on-line.

Пример. Структура изделия ASK полностью описывается во взаимосвязанных внешних спецификациях интерфейса пользователя ASK (C013/L321) и процессора корректировок ASK (C013/L331).

Справочное руководство и справочный буклет должны быть готовы в окончательном виде в большом количестве (можно ксерокопии) на этапе Д21 (раздел 7). Это требуется фирме ABC Services для проведения обучения.

2.6.2 Техническая ревизионная комиссия

В каждом СТ следует рекомендовать создание технической ревизионной комиссии (ТРК) с указанием места работы каждого члена комиссии и его фамилии, если это возможно, а также назначение председателя этой комиссии.

Пример. От каждого из следующих лиц было получено личное согласие работать в ТРК:

- Боб Уилбур (отдел испытаний программ) — председатель;
- К.В. Гаррисон (фирма ABC Services);
- Роберт Вонг (отдел выпуска документации);
- Боб Симе (отдел разработки прикладных программ).

2.6.3 Проверка изделия

2.6.3.1 Уровни испытаний

Испытания программ могут быть организованы в три этапа, проводиться в трех режимах и насчитывать десять категорий (см. раздел 5 «Тестирование»). Эта информация представляется в виде таблицы. Для каждого этапа и категории указывается, кто будет проводить испытания. Определяется роль группы испытаний посредством установления режимов испытаний.

Пример. Уровни испытаний приведены в таблице 2.4.

Таблица 2.4 — Уровни испытаний

Категория испытаний	Класс испытаний		
	A	B	C
Демонстрация в действии		/	/
Аттестация	Р	/	/
Полная функциональная проверка	Р	И	/
Проверка новых свойств			/
Эксплуатационные испытания	Р	И	
Испытания надежности	Р	И	/
Проверка устойчивости			/
Возвратная проверка			/
Пусковые испытания	Р	И	О
Испытания конфигураций	Р	И	О
Режимы испытаний: I — проводятся группой испытаний ( ) II — контролируются группой испытаний (X) III — группа испытаний не участвует ( )			
Подразделения, проводящие испытания: Р — группа разработки И — группа испытаний О — группа обслуживания / — испытания исключены			

Фирма ABC Services в течение части периода испытаний класса B выделяет двух специалистов, имеющих опыт работы в области финансового анализа, для работы за терминалами. Испытания в условиях минимальной конфигурации, описанной в разделе 2.3.3.1.1.4, проводятся на реальном оборудовании; аналогичным образом проверяется базовая конфигурация, за исключением контроллеров связи, линий связи и терминалов. Максимальная конфигурация должна содержать одно устройство типа M442, одно устройство типа M443, предположительно семь телефонных каналов и терминалы. Число контроллеров ограничено имеющейся аппаратурой, а число линий связи и терминалов — имеющимся персоналом.

Отдел испытаний программных средств разрабатывает имитатор терминалов, а отдел электронных интерфейсов — разветвленные каналы для работы с этой моделью. Для проверки изделия ASK одновременно имитируются до 144 устройств типа 1024 Telcoscope путем подключения имитатора на входы кана-

лов системы VSOS, а изделия ASK — на выходы каналов. Этот же режим имитации обеспечивает моделирование работы одного устройства типа 43-1 на каждое устройство типа 43.

2.6.3.2 Эталоны для сравнения

Определяются эталонные системы, относительно которых должно выполняться сравнение. Указываются характеристики данной системы в относительных единицах. Если эталона для сравнения нет, приводятся абсолютные значения характеристик.

Пример. Поскольку ASK является новым изделием, нет базы для корректировки ошибочных решений. Критериями при проведении испытаний класса B будут лишь документ, в котором описывается назначение изделия ASK и требования к нему, и внешняя спецификация изделия ASK.

2.6.4 Обеспечение поддержки

Для каждого подраздела указываются конкретные мероприятия; при этом делаются ссылки на план поддержки или констатируется, что соответствующие меры отсутствуют.

2.6.4.1 Мероприятия, обеспечивающие продвижение программного изделия на рынок

Могут указываться, например, экспозиции, которые следует подготовить для торговых выставок или для определенного заказчика.

2.6.4.2 Мероприятия, связанные с обучением

Описываются формы обучения различных категорий слушателей и относительное время его проведения (например, в фазе оценки). Указывается требуемый уровень начальной подготовки и требуемая квалификация.

2.6.4.3 Средства, обеспечивающие модернизацию программного изделия

Можно сослаться на раздел 2.3.(2,3).X.1.2.

2.7 Извещение об изменении календарных сроков

Пример.

Наименование проекта: Разработка изделия ASK Шифр проекта: C013. Шифр изделия: L301A. Наименование изделия: ASK				
Шифр этапа	Наименование этапа	Прежний срок	Новый срок	Примечания
П10	Утверждение бюджетной заявки	29.09.77		
Р10	Определение назначения изделия и требований к нему	03.11.77		
П20	Утверждение назначения и требований к изделию	15.12.77		
Р20	Составление внешней спецификации	09.01.78		
И10	Утверждение плана испытаний	09.02.78		
Р30	Утверждение внешней спецификации	15.03.78	06.02.78	
О10	Установка аппаратуры, необходимой для разработки	31.03.78		
Д10	Утверждение плана поддержки	02.03.78		
Р41	Демонстрация в действии	15.05.78		
И30	Начало испытаний класса В	03.07.78	08.05.78	
Д20	Рассылка рекламных материалов	15.07.78		
Д21	Подготовка учебных пособий	01.08.78		
С20	Подготовка спецификации сопровождения	15.08.78		
О20	Начало распространения изделия	01.09.78	17.07.78	
Подготовил С. Девис Утвердил Старая дата 06.01.78		Утвердил К. Андерсен Утвердил Новая дата 13.01.78		

3 НАПИСАНИЕ СПЕЦИФИКАЦИЙ

Написание спецификаций — цель первой части второй лабораторной работы. Также спецификации являются третьим разделом курсовой работы.

На этапе определения спецификаций осуществляется точное описание функций, реализуемых ЭВМ, а также задаются структуры входных и выходных данных, методы и средства их размещения. Определяются алгоритмы обработки данных.

Центральным вопросом определения спецификаций является проблема организации базы данных. При этом решается комплекс вопросов, имеющих отношение к структуре файлов, организации доступа к ним, модификации и удаления.

В случае, когда новая система создается на основе существующих, составной частью спецификаций является схема (алгоритм) приведения существующей базы данных к новому формату. Такое преобразование может потребовать разработку специальной программы, которая становится ненужной после ее первого и единственного использования.

Все эти вопросы должны быть отражены в функциональных спецификациях, которые представляют собой документ, являющийся основополагающим в процессе разработки системы, так как содержит конкретное описание последней. Чем подробней составлены спецификации, тем меньше вероятность возникновения ошибок.

В спецификациях должны быть представлены данные для тестирования элементов системы и системы в целом. Это требование является объективным и обязательным, так как на данном этапе на параметры тестирования не будет оказывать влияние конкретная реализация системы.

Так как функциональные спецификации описывают принятые решения в целом, данный документ можно использовать для начальных оценок временных затрат, числа специалистов и других ресурсов, необходимых для проведения работ.

В общем случае спецификации определяют те функции, которые должна выполнять система, не указывая, каким образом это достигается. Составление подробных алгоритмов на этом

этапе преждевременно и может вызвать нежелательные осложнения.

Пример. Далее приводится пример оформления спецификации на программу «Электронный каталог».

Внешняя спецификация:

```
main: procedure (File);
  declare 1 File;
            2 Name: string [20];
            2 Album: string [15];
            2 Genre: string [15];
            2 Year: string [4];

  if File not found then
  begin
    put ('ошибка открытия файла');
    call Create;
  end;
  if length (File)=0 then
    put ('файл не содержит записей');

  do case (кнопка)
    // Вывод содержимого файла на экран
    «Просмотр»: call View;
    // Поиск записи в файле
    «Поиск»: call Search;
    // Добавление записи
    «Добавить»: call Add;
    // Удаление записи из файла
    «Удалить»: call Del;
    // Выход из программы
    «Выход»: call Exit;
  end;
end main.
```

Внутренняя спецификация:

```
procedure View (File);
begin
  do while EOF(File)
  begin
    get (запись);
    put (запись);
```

```
  end;
end View;

procedure Search (File);
begin
  get (искомое значение);
  do while EOF(File)
  begin
    if (искомое значение)=true then
      put (Name, Album, Genre, Year);
  end;
end Search;

procedure Add (File);
begin
  get (запись);
  put (запись в файл);
end Add;

procedure Del (File);
begin
  get (запись);
  put (удаление записи из файла);
end Del;

procedure Create (File);
begin
  get (создание файла);
end Create;
```

4 ТЕСТИРОВАНИЕ

Проведение тестирования — цель второй части второй лабораторной работы. Также результаты тестирования являются четвертым разделом курсовой работы.

4.1 Общие принципы тестирования

Этап тестирования обычно в финансовых затратах составляет половину расходов на создание системы. Плохо спланированное тестирование приводит к существенному увеличению сроков разработки системы и является основной причиной срывов графиков разработки.

В процессе тестирования используются данные, характерные для системы в рабочем состоянии, т.е. данные для тестирования выбираются случайным образом. План проведения испытаний должен быть составлен заранее, обычно на этапе проектирования.

Тестирование подразумевает три стадии:

- автономное;
- комплексное;
- системное.

При *автономном* тестировании модуль проверяется с помощью данных, подготовленных программистом. При этом программная среда модуля имитируется с помощью программ управления тестированием, содержащих фиктивные программы вместо реальных подпрограмм, к которым имеется обращение из данного модуля (заглушки). Подобную процедуру называют программным тестированием, а программу тестирования — **UUT** (тестирующей программой). Модуль, прошедший автономное тестирование, подвергается комплексному тестированию.

В процессе *комплексного* тестирования проводится совместная проверка групп программных компонентов. В результате имеем полностью проверенную систему. На данном этапе тестирование обнаруживает ошибки, пропущенные на стадии автономного тестирования. Исправление этих ошибок может составлять до четверти от общих затрат.

Системное (или оценочное) тестирование — это завершающая стадия проверки системы, т.е. проверка системы в целом с помощью независимых тестов. Независимость тестов является главным требованием. Обычно Заказчик на стадии приемки работ настаивает на проведении собственного системного тестирования. Для случая, когда сравниваются характеристики нескольких систем (имеется альтернативная разработка), такая процедура известна как *сравнительное* тестирование.

В процессе тестирования для определения правильности выполнения программы вводится ряд критериев:

1) каждый оператор должен быть выполнен, по крайней мере, один раз для заданного набора тестов, и программа должна выдать правильный результат;

2) каждая ветвь программы должна быть опробована, и программа при этом должна выдать правильный результат;

3) каждый путь в программе должен быть испытан хотя бы один раз с использованием набора тестовых данных, и программа должна выдать правильный результат;

4) для каждой спецификации программы необходимо располагать набором тестовых данных, позволяющих установить, что программа правильно реализует данную спецификацию.

Хотя критерии п.п. 1 и 2 кажутся схожими, в действительности они сильно разнятся. Например, арифметический оператор **IF** в Fortran

IF (Выражение) N_1, N_2, N_3

Критерий п. 1 подразумевает, что **IF** должен быть выполнен, в то время как п. 2 подразумевает различные наборы данных, для того чтобы выполнились условия N_1, N_2, N_3 (т.е. передачу на эти метки).

В общем случае не существует единого программного критерия, определяющего «хорошо проверенную» программу.

Тесно связаны с тестированием понятия «верификация» и «испытание».

Испытание системы осуществляется посредством тестирования. Цель такой проверки — показать, что система функционирует в соответствии с разработанными на нее спецификациями.

Верификация заключается в выполнении доказательств, что программа удовлетворяет своим спецификациям.

Современный процесс разработки программ не позволяет реализовать обе указанные концепции. Для ситуаций, не контролируемых тестовыми данными, система, прошедшая испытания, может дать неверные результаты. После проведения верификации система работает правильно лишь относительно первоначальных спецификаций и допущений о поведении окружающей среды; формальные доказательства правильности программ весьма сложны и слабо разработаны.

Общий процесс создания правильных программ с помощью процедур испытания и верификации называется *аттестацией*.

Различаются три вида отклонения системы от нормальной работы.

Сбой системы — это явление, связанное с нарушением системой установленных на нее спецификаций.

Данные, при обработке которых правильными алгоритмами системы происходит сбой, называются *выбросом*. Исправление выброса можно предусмотреть в программе, так что не каждый выброс может приводить к сбою.

Ошибка — это алгоритмический дефект, который создает выброс (программная ошибка).

Различают понятия «правильная» и «надежная» программа. *Правильная* программа — это та, что удовлетворяет своим спецификациям. Что касается *надежной* программы, то она не обязательно является правильной, но выдает приемлемый результат даже в том случае, когда входные данные либо условия ее использования не удовлетворяют принятым допущениям. Естественно стремление иметь «живую» (robustness) систему, т.е. систему, способную воспринимать широкий спектр входных данных при неблагоприятных условиях.

Система является *правильной*, если в ней нет ошибок, а ее внутренние данные не содержат выбросов. Система называется *надежной*, если, несмотря на сбои, она продолжает удовлетворительно функционировать. Это особо видно на примере операционной системы (ОС), включающей систему обработки сбоев. При обнаружении выброса такая система прекращает работу с

сохранением текущей информации и возможности продолжения работы после устранения выброса.

4.2 Организация испытаний программных изделий

Под испытаниями понимают не отладку, призванную определить, почему в программе возникает та или иная ошибка и устранить ее причины, а процесс установления самого факта наличия дефектов и расхождения между истинными свойствами программного изделия и его спецификациями. Нельзя сказать, что испытания программного изделия гарантируют обеспечение его качества. Обеспечение качества программного изделия включает помимо испытаний еще целый ряд других процедур (анализ эксплуатационных характеристик, использование «стандартных» методов проектирования и программирования, восстанавливаемость после отказа, простота сопровождения, повторяемость результатов и др.) Однако испытания — важнейшая из этих процедур.

Группа испытаний оказывает значительное влияние на качественную сторону проектирования, используя такие воздействия, как технические ревизионные комиссии, соглашения о требованиях, спецификации и обзоры состояния проекта в различных фазах. Однако группа испытаний не может нести ответственность за качество изделия, так как она не управляет процессом создания отдельных компонентов программного обеспечения. В задачи группы испытаний входит:

- проведение испытаний;
- выработка оценок;
- участие в фазовых обзорах с целью влияния на ход разработок.

4.3 Виды испытаний программного изделия. Стадии испытаний

В общем случае, испытания проводятся в несколько стадий, разделенных по времени.

К первой стадии относятся испытания класса А, которые проводятся в конце фазы программирования, после того как будут отлажены и включены в систему все модули изделия. Этот процесс сопровождается *системной отладкой*, когда исправляются ошибки сопряжения модулей.

Ко второй стадии относятся испытания класса В, когда осуществляется независимая (от группы разработки) проверка компонентов законченного изделия как отдельно, так и во взаимодействии друг с другом. В идеальном случае испытания класса В начинаются после того, как разработчики объявляют, что изделие готово к передаче потребителю. В ходе испытаний класса В функционирование проверяется на соответствие требованиям, спецификациям, документации и цели.

Испытания класса С осуществляются после того, как группа испытаний рекомендует выпуск изделия и его распространение. Испытания класса С похожи на выборочный контроль в производстве, поскольку с полки случайным образом выбирают экземпляр программного изделия и выполняют прогон программ, бегло анализируя результаты.

Пример. Поскольку написание программы завершено, для проверки правильности работы программы выбран класс В (тестирование после разработки). Испытания класса А (тестирование в процессе разработки) отвергаются, а класс С (предпродажное тестирование) не может быть выбран потому, что тестируемая программа является учебной и не предназначена для продажи.

4.4 Режимы испытаний программ

Испытания различаются в зависимости от того, кто их проводит. Основная идея — независимость функции испытаний от функции разработки.

Режим I испытаний подразумевает полный цикл деятельности группы испытаний, включая планирование испытаний, разработку тестов, их прогон и анализ результатов. Обычно эта процедура является высшей и наиболее строгой формой контроля и используется для проверки универсальных программных изделий.

Режим II позволяет проводить ускоренные испытания изделия, поскольку в этом случае группа испытаний несет ответственность только за анализ результатов испытаний, а составление плана и спецификаций испытаний, построение тестов и их прогон поручаются разработчикам.

Режим III реализуется без участия группы испытаний. Этот режим используется лишь в случаях крайней необходимости, например при сильном нарушении сроков проектирования опытного образца, когда независимые испытания изделия или, по крайней мере, независимый контроль над испытаниями исключаются. Для гарантии успеха в этом неблагоприятном случае следует предусмотреть ввод в действие и поддержку такого изделия группой разработки. При этом качество программного изделия весьма сомнительно.

Пример. Тестирование будет выполняться в режиме II (выполняется разработчиком, а выводы делает независимая группа), так как режим I (тестирование в отдельной организации) недоступен, режим III (тесты и выводы делает разработчик) также отвергается.

4.5 Категории испытания программного изделия

Стадии испытания указывают на *время* проведения проверок, а режимы определяют тех, *кто проводит*. Категории испытаний устанавливают *характер* и *назначение* тестов. Продуманное деление испытаний изделий на категории облегчает общение между различными функциональными группами и степень их участия в работе. На практике выделяют следующие категории испытаний.

Демонстрация в действии. Во время демонстрации прогоняют специально подобранные тесты, обеспечивающие желаемый результат. Тесты обычно подбираются и выполняются в рамках функции разработки во время испытаний класса А, чтобы убедить руководителей всех заинтересованных функциональных групп в том, что изделие достигло определенного уровня завершенности.

Аттестация. Аттестация призвана гарантировать способность данного программного изделия правильно обрабатывать

реальные входные данные в условиях пользователя и давать верные результаты. Испытания этой категории проводятся для того, чтобы удовлетворить требования рынка сбыта и Заказчика, также для того, чтобы продемонстрировать совместимость или эксплуатационные качества изделия. Спецификация испытаний готовится группой поддержки, а аттестация проводится группой разработки по окончании испытаний класса А.

Полная функциональная проверка. Цель этой категории испытаний — показать, что изделие обладает всеми функциональными возможностями, указанными во внешних спецификациях, и работает правильно. Если объектом испытания является новая версия существующего изделия, проверке подвергаются как новые, так и старые функциональные возможности изделия, отдельно и во взаимодействии друг с другом. Испытания этой категории включаются в состав испытаний классов А и В.

Проверка новых свойств. Этим испытаниям подвергаются только новые версии существующих программных систем в целях оценки их новых функциональных качеств. Проверка новых свойств обычно проводится в рамках испытаний классов А и В и выполняется в тех случаях, когда изделие подвергается лишь незначительным изменениям.

Эксплуатационные испытания. В результате этой проверки оцениваются эксплуатационные характеристики программного изделия, такие, как скорость выполнения операций, объем занимаемой памяти, пропускная способность, скорость пересылки данных, время транслирования, компоновки или генерации, время реакции и условия взаимодействия с пользователем. Эксплуатационные свойства оцениваются в ходе испытаний классов А и В.

Испытания надежности. Во время этих испытаний изделие ставится в условия, позволяющие оценить его способность к устойчивой работе или восстановлению после отказа. Обычно в ходе этих испытаний преднамеренно вносятся искусственно созданные ошибки, испытывают изделие в условиях непрерывной работы в течение нескольких часов и проверяют все восстановительные процедуры. Испытания надежности входят в состав испытаний классов А и В.

Проверка устойчивости. Эти испытания призваны гарантировать правильность объединения программных изделий в систему. Они должны убедить всех в том, что взаимодействие различных программных средств не создает ошибочных ситуаций. В отношении отдельных изделий фиксируется среднее время между отказами. Проверка проводится в рамках испытаний классов А и В.

Возвратная проверка. В эту категорию испытаний входит проверка новой версии или редакции изделия, подтверждающая, что ранее замеченные дефекты исправлены и исправления не привели к появлению новых ошибок. Возвратная проверка входит в состав испытаний классов А и В.

Пусковые испытания. Эти испытания подтверждают, что ввод программного изделия в действие может быть осуществлен в полном соответствии с описанием, т.е. в отведенное для этого время, силами персонала, обученного соответствующим образом, с помощью технической документации и с помощью только тех средств, которые были предусмотрены в описании. Испытания проводятся на различных конфигурациях технических средств ЭВМ и обычно входят в состав испытаний классов А и В.

Конфигурационные испытания. Эти испытания призваны гарантировать, что изделие правильно функционирует на всех конфигурациях вычислительной техники, которые были предусмотрены проектом. В процессе этих испытаний создаются минимальные базовые конфигурации и имитируются максимальные. Конфигурационные испытания выполняются в рамках испытаний классов А и В.

Стадии, режимы и категории испытаний наглядно можно представить в табличной форме по аналогии с таблицей из раздела 2.6.3.1 СТ.

4.6 Технология тестирования, классы эквивалентности

Одним из способов изучения поставленного вопроса является исследование стратегии тестирования, называемой стратегией черного ящика, *тестированием с управлением по данным*

или *тестированием с управлением по входу-выходу*. При использовании этой стратегии программа рассматривается как черный ящик. Иными словами, такое тестирование имеет целью выяснение обстоятельств, в которых поведение программы не соответствует ее спецификации. При таком подходе обнаружение всех ошибок в программе является критерием *исчерпывающего входного тестирования*. Последнее может быть достигнуто, если в качестве тестовых наборов использовать все возможные наборы входных данных.

Стратегия *белого ящика*, или стратегия *тестирования, управляемого логикой программы*, позволяет исследовать внутреннюю структуру программы. В этом случае тестирующий получает тестовые данные путем анализа логики программы (к сожалению, здесь часто не используется спецификация программы).

Тестирование в данной курсовой работе должно выполняться по технологии черного ящика с использованием методологии эквивалентного разбиения. Разработка тестов методом эквивалентного разбиения осуществляется в два этапа:

1) выделение классов эквивалентности;

2) построение тестов.

Классы эквивалентности выделяются путем выбора каждого входного условия (обычно это предложение или фраза в спецификации) и разбиением его на две или более группы. Для проведения этой операции используют таблицу, подобную табл. 4.1.

Таблица 4.1 — Форма таблицы для перечисления классов эквивалентности

Входные условия	Классы эквивалентности	
	Правильные	Неправильные

Отметим, что различают два типа классов эквивалентности: правильные классы эквивалентности, представляющие правильные входные данные программы, и неправильные классы эквивалентности, представляющие все другие возможные состояния условий (т.е. ошибочные входные значения). Таким образом,

придерживаются одного из принципов необходимости сосредоточивать внимание на неправильных или неожиданных условиях.

Если задаться входными или внешними условиями, то выделение классов эквивалентности представляет собой в значительной степени эвристический процесс. При этом существует ряд правил:

1. Если входное условие описывает область значений (например, «целое данное может принимать значения от 1 до 999»), то определяются один правильный класс эквивалентности значений (от 1 до 999) и два неправильных (значения целого данного, меньшие 1, и значения целого данного, большие 999).

2. Если входное условие описывает число значений (например, «в автомобиле могут ехать от одного до шести человек»), то определяются один правильный класс эквивалентности и два неправильных (ни одного и более шести).

3. Если входное условие описывает множество входных значений и есть основание полагать, что каждое значение программа трактует особо (например, «известны способы передвижения на АВТОБУСЕ, ГРУЗОВИКЕ, ТАКСИ, ПЕШКОМ или на МОТОЦИКЛЕ»), то определяется правильный класс эквивалентности для каждого значения и один неправильный класс эквивалентности (например, «НА ПРИЦЕПЕ»).

4. Если входное условие описывает ситуацию «должно быть» (например, «первым символом идентификатора должна быть буква»), то определяется один правильный класс эквивалентности (первый символ — буква) и один неправильный (первый символ — не буква).

5. Если есть любое основание считать, что различные элементы класса эквивалентности трактуются программой неодинаково, то данный класс эквивалентности разбивается на меньшие классы эквивалентности.

4.7 Построение тестов

Процесс построения тестов включает в себя:

1) назначение каждому классу эквивалентности уникального номера;

2) проектирование новых тестов, каждый из которых покрывает как можно большее число непокрытых правильных классов эквивалентности до тех пор, пока все правильные классы эквивалентности не будут покрыты тестами (только не общими);

3) запись тестов, каждый из которых покрывает один и только один из непокрытых неправильных классов эквивалентности до тех пор, пока все неправильные классы эквивалентности не будут покрыты тестами.

Причина покрытия неправильных классов эквивалентности индивидуальными тестами состоит в том, что определенные проверки с ошибочными входами скрывают или заменяют другие проверки с ошибочными входами. Например, спецификация устанавливает «тип книги при поиске (ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА, ПРОГРАММИРОВАНИЕ или ОБЩИЙ) и количество (1—9999)». Тогда тест

XYZ 0

отображает два ошибочных условия (неправильный тип книги и количество) и, вероятно, не будет осуществлять проверку количества, так как программа может ответить «XYZ — НЕСУЩЕСТВУЮЩИЙ ТИП КНИГИ» и не проверять остальную часть входных данных.

Пример. Предположим, что при разработке компилятора для подмножества языка Фортран требуется протестировать синтаксическую проверку оператора DIMENSION. Спецификация приведена ниже. В спецификации элементы, написанные латинскими буквами, обозначают синтаксические единицы, которые в реальных операторах должны быть заменены соответствующими значениями, в квадратные скобки заключены необязательные элементы, многоточие показывает, что предшествующий ему элемент может быть повторен подряд несколько раз.

Оператор DIMENSION используется для определения массивов. Форма оператора DIMENSION:

DIMENSION *ad* [, *ad*] …,

где *ad* есть описатель массива в форме

n(*d*[,*d*]…),

где *n* — символическое имя массива, а *d* — индекс массива. Символические имена могут содержать от одного до шести символов — букв или цифр, причем первой должна быть буква. Допускается от одного до семи индексов. Форма индекса

[*lb*:]*ub*,

где *lb* и *ub* задают нижнюю и верхнюю границы индекса массива. Граница может быть либо константой, принимающей значения от –65534 до 65535, либо целой переменной (без индексов). Если *lb* не определена, то предполагается, что она равна единице. Значение *ub* должно быть больше или равно *lb*. Если *lb* определена, то она может иметь отрицательное, нулевое или положительное значение. Как и все операторы, оператор DIMENSION может быть продолжен на нескольких строках.

(Конец спецификации).

Первый шаг заключается в том, чтобы идентифицировать входные условия и по ним определить классы эквивалентности (табл. 4.2). Классы эквивалентности в таблице обозначены числами.

Таблица 4.2 — Классы эквивалентности

Входные условия	Классы эквивалентности	
	Правильные	Неправильные
Число описателей массивов	один (1), больше одного (2)	ни одного (3)
Длина имени массива	1—6 (4)	0 (5), больше 6 (6)
Имя массива	содержит буквы (7) и цифры (8)	содержит другие символы (9)
Имя массива начинается с буквы	да (10)	нет (11)
Число индексов	1—7 (12)	0 (13), больше 7 (14)
Верхняя граница	константа (15), целая переменная (16)	имя элемента массива (17), что-то иное (18)
Имя целой переменной	содержит буквы (19) и цифры (20)	состоит из других символов (21)
Целая переменная начинается с буквы	да (22)	нет (23)

Окончание табл. 4.2

Входные условия	Классы эквивалентности	
	Правильные	Неправильные
Константа	–65534..65535 (24)	< –65534 (25), > 65535 (26)
Нижняя граница определена	да (27), нет (28)	
Верхняя граница по отношению к нижней границе	больше (29), равна (30)	меньше (31)
Значение нижней границы	< 0 (32), 0 (33), > 0 (34)	
Нижняя граница	константа (35), целая переменная (36)	имя элемента массива (37), что-то иное (38)
Оператор расположен на нескольких строках	да (39), нет (40)	

Следующий шаг — построение теста, покрывающего один или более правильных классов эквивалентности. Например, тест

DIMENSION A(2)

покрывает классы 1, 4, 7, 10, 12, 15, 24, 28, 29 и 40. Далее определяются один или более тестов, покрывающих оставшиеся правильные классы эквивалентности. Так, тест

DIMENSION A 12345 (I, 9, J4XXXX, 65535, 1 , KLM, * 100),
BBB (–65534 : 100,0 : 1000,10 : 10,1: 65535)

покрывает оставшиеся классы. Перечислим неправильные классы эквивалентности и соответствующие им тесты:

(3) DIMENSION
(5) DIMENSION (10)
(6) DIMENSION A234567(2)
(9) DIMENSION A.I(2)
(11) DIMENSION 1A(10)
(13) DIMENSION B
(14) DIMENSION B (4,4,4,4,4,4,4,4)
(17) DIMENSION B(4,A(2))
(18) DIMENSION B(4„7)
(21) DIMENSION C(I.,10)
(23) DIMENSION C(10,1J)
(25) DIMENSION D(–65535:1)

(26) DIMENSION D(65536)
(31) DIMENSION D(4:3)
(37) DIMENSION D(A(2):4)
(38) DIMENSION D(.:4)

Эти классы эквивалентности покрываются 18 тестами. Хотя эквивалентное разбиение значительно лучше случайного выбора тестов, оно все же имеет недостатки (т.е. пропускает определенные типы высокоэффективных тестов). Следующие два метода — анализ граничных значений и использование функциональных диаграмм (диаграмм причинно-следственных связей cause-effect graphing) — свободны от многих недостатков, присущих эквивалентному разбиению.

Результаты тестирования приводятся в заключении пояснительной записки к курсовому проекту, также даются рекомендации по устранению ошибок, выявленных в результате тестирования.

5 РУКОВОДСТВО СИСТЕМНОГО ПРОГРАММИСТА

Составление руководства системного программиста — цель третьей части второй лабораторной работы. Также руководство системного программиста является пятым разделом курсовой работы.

Требования к оформлению руководства системного программиста излагаются в ГОСТ 19.503-79 «Руководство системного программиста. Требования к содержанию и оформлению».

5.1 ГОСТ 19.503-79

Настоящий стандарт устанавливает требования к содержанию и оформлению программного документа «Руководство системного программиста», определенного ГОСТ 19.101-77.

Стандарт полностью соответствует СТ СЭВ 2094-80.

5.1.1 Общие положения

1.1. Структуру и оформление документа устанавливают в соответствии с ГОСТ 19.105-78.

1.2. Руководство системного программиста должно содержать следующие разделы:

– общие сведения о программе;
– структура программы;
– настройка программы;
– проверка программы;
– дополнительные возможности;
– сообщения системному программисту.

В зависимости от особенностей документа допускается объединять отдельные разделы или вводить новые. В обоснованных случаях допускается раздел «Дополнительные возможности» не приводить, а в наименованиях разделов опускать слово «программа» или заменять его наименованием программы.

5.1.2 Содержание разделов

2.1. В разделе «Общие сведения о программе» должны быть указаны назначение и функции программы и сведения о технических и программных средствах, обеспечивающих выполнение данной программы.

2.2. В разделе «Структура программы» должны быть приведены сведения о структуре программы, ее составных частях, о связях между составными частями и о связях с другими программами.

2.3. В разделе «Настройка программы» должно быть приведено описание действий по настройке программы на условиях конкретного применения (настройка на состав технических средств, выбор функций и др.). При необходимости приводят поясняющие примеры.

2.4. В разделе «Проверка программы» должно быть приведено описание способов проверки, позволяющих дать общее заключение о работоспособности программы (контрольные примеры, методы прогона, результаты).

2.5. В разделе «Дополнительные возможности» должно быть приведено описание дополнительных разделов функциональных возможностей программы и способов их выбора.

2.6. В разделе «Сообщения системному программисту» должны быть указаны тексты сообщений, выдаваемых в ходе выполнения настройки, проверки программы, а также в ходе выполнения программы, описание их содержания и действий, которые необходимо предпринять по этим сообщениям.

2.7. В приложении к руководству системного программиста могут быть приведены дополнительные материалы (примеры, иллюстрации, таблицы, графики и т.п.).

5.2 Пример

5.2.1 Общие сведения о программе

Программа «Автоматизированное рабочее место читателя» предназначена для обеспечения простого и удобного доступа удаленных пользователей к ресурсам различных информационных служб. Удаленными пользователями могут являться любые

пользователи HTTP-сервера, на базе которого функционирует программа. Ресурсами информационных служб могут являться различные удаленные базы данных (библиографические, полнотекстовые, базы данных запросов на доставку документов, тезаурусы), доступные по протоколу ISO 23950. Таким образом, программа является промежуточным звеном многозвенной информационной системы, обеспечивающим пользователей следующими возможностями:

– поиск библиографической, полнотекстовой информации, информации о заказах, сведений о местонахождении и доступности документа в удаленных базах данных с возможностями устранения дублетности, навигации по поисковым индексам и автоматического расширения запроса с использованием тезаурусов и авторитетных файлов;

– заказ документа (копии) по найденному библиографическому описанию;

– проверка статуса заказа;

– вывод списка документов, полученных во временное пользование.

Другими словами, программа является Z39.50-клиентом с Web-интерфейсом пользователя. Поэтому для работы с программой пользователю необходимы лишь Web-агент (Netscape Navigator, MS Internet Explorer, Lynx и т.п.) и надежная связь с HTTP-сервером, на котором выполняется программа.

Программа может функционировать на любых технических средствах под управлением любой операционной системы, отвечающей стандарту ISO/IEC 9945-1 (POSIX) и спецификациям X/Open CAE specifications, Issue 4, July 1992 (XPG4). Обязательными требованиями для выполнения программы являются поддержка стека протоколов TCP/IP и наличие HTTP-сервера, поддерживающего CGI.

5.2.2 Структура программы

Программа «Автоматизированное рабочее место читателя» состоит из следующих компонентов:

1) zcon — приложение, реализующее функции Z39.50-клиента;

2) zgate — CGI-приложение, обеспечивающее передачу данных от HTTP-сервера Z39.50-клиенту и обратно;

3) zgate.cat — каталог сообщений, используемый двумя вышеуказанными приложениями для формирования выходных данных. Количество таких каталогов равняется количеству поддерживаемых программой языков сообщений (как минимум, два — русский и английский);

4) zgate.xsl — основная XSLT-программа визуального представления выходных данных. Может редактироваться администратором в целях изменения содержания и формы представления выходных данных;

5) usmarc.xsl — подпрограмма визуального представления записей в формате MARC21 (USMARC). Может редактироваться администратором в целях изменения содержания и формы представления выходных данных.

6) …

Примечание. Для правильного функционирования программы необходима установка следующих библиотек, поддерживающих операции над XML-документами и XSLT:

– xml2 — библиотека поддержки операций над XML-документами;

– xslt — библиотека поддержки XSLT;

– exslt — библиотека поддержки расширений XSLT.

В версию программы для операционных систем, не поддерживающих спецификации X/Open CAE specifications, Issue 4, July 1992 для интерфейсов catgets и iconv (например, Microsoft Windows), дополнительно включаются следующие библиотеки:

– catgets — интерфейс поддержки каталогов сообщений;

– iconv — интерфейс конвертора данных из одного кодированного набора символов в другой.

5.2.3 Настройка программы

5.2.3.1 Установка программы

В настоящем документе для именования файлов используется синтаксис, определенный ISO/IEC 9945-1. В тех операционных системах, которые не поддерживают указанный способ

именования файлов в приложениях, используемых при установке и настройке программы, следует использовать способ именования файлов, определяемый используемой операционной системой. Если, например, используется операционная система Microsoft Windows, то указанное в настоящем документе имя файла вида /a/b/c/f следует заменить на X:\a\b\c\f, где X: — название диска, на котором размещаются приложения zcon и zgate.

1. Разместить исполняемые файлы zcon и zgate в каталоге CGI-приложений HTTP-сервера.

2. Разместить файлы zgate.cat и ru_RU.UTF-8/zgate.cat в одном из системных каталогов сообщений, определяемых переменной окружения NLSPATH. Если, например, NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/dt/lib/nls/msg/%L/%N.cat, то необходимо поместить файл C/zgate.cat в каталог /usr/lib/nls/msg/C, а файл ru_RU.UTF-8/zgate.cat в каталог /usr/lib/nls/msg/ru_RU.UTF-8. Если определить местонахождение системного каталога сообщений невозможно, то следует разместить указанные файлы в каталоге /usr/lib/nls/msg и присвоить переменной окружения NLSPATH значение /usr/lib/nls/msg/%L/%N.

3.…

Примечание. Пользователь, устанавливающий программу, должен иметь полномочия, достаточные для записи указанных файлов в требуемые каталоги. В указанных каталогах должно быть достаточно свободного пространства для размещения файлов, в т.ч. и временных, размер которых зависит от размеров записей, извлекаемых пользователем.

5.2.3.2 Настройка программы

Обычно в данном подразделе приводятся несколько скриншотов рабочего окна программы с описанием (куда надо нажать и что ввести, чтобы получить результат) и описывается процедура настройки (если присутствует).

5.2.4 Проверка программы

Проверка программы осуществляется методом ее выполнения. В связи с тем, что конкретные условия применения про-

граммы (адреса Z39.50-серверов, названия баз данных, поддерживаемые точки доступа, форматы записей и т.п.) не могут быть известны заранее, можно ограничиться следующими рекомендациями:

1. При проверке программы следует использовать все возможные элементы управления — списки с точками доступа, операторами, форматами записей, выключатели и т.п. Например, можно последовательно осуществлять поиск по каждой из возможных точек доступа, затем получать записи в каждом из возможных форматов и т.д.

2. При получении диагностических и иных сообщений в ходе проверки программы следует обращаться к разделу «Сообщения системному программисту» данного руководства.

5.2.5 Дополнительные возможности

Дополнительной возможностью программы является возможность динамического управления формой представления записей при просмотре их в полном формате («Детальная информация») при помощи Web-агента, поддерживающего редактирование текущего URL. В этом случае URL имеет вид:

"http://"<адрес_сервера>":"<порт>"/"
<каталог_с_CGI-приложениями>"zgate?present+"
<идентификатор_соединения>"+"
<наименование_результирующего_множества>"+"
<номер_первой_записи_в_выдаваемой_порции>"+"
<количество_записей>"+"
<наименования_набора_элементов>"+"
<формат_записи>"+"
<код_языка_интерфейса_пользователя>".

5.2.6 Сообщения системному программисту

В таблице 5.1 представлены сообщения, которые могут получить системный программист в ходе выполнения настройки, проверки программы, а также пользователь в ходе выполнения программы. Описано содержание этих сообщений и действия

системного программиста, которые необходимо предпринять по этим сообщениям.

Таблица 5.1 — Сообщения системному программисту и пользователю

Сообщение	Описание	Действия системного программиста
В шлюз не включена форма <имя_файла>.	Отсутствует файл <имя_файла> с поисковой формой, указанный в одном из параметров настройки программы.	Проверить наличие файла с именем <имя_файла>. При необходимости создать файл с указанным именем либо изменить параметры настройки таким образом, чтобы включалась уже существующая форма.
В форме не указана переменная <название_переменной>	В форме отсутствует обязательный элемент управления <название_переменной>.	Добавить в форму элемент управления <название_переменной>.
Недостаточное количество параметров!	Точка входа для доступа к Z39.50-серверу не соответствует формату, указанному в п. 5.2.3.2 настоящего документа.	Привести точку входа в соответствие с требованиями, добавив необходимый параметр к значению атрибута href соответствующей гиперссылки.
…	…	…

СПИСОК ЛИТЕРАТУРЫ

Основная:

1. Калайда В.Т., Романенко В.В. Технология разработки программного обеспечения: Учебное пособие. — Томск: ТМЦДО, 2007. — 236 с.

2. Зельковиц М., Шоу А., Гэннон Дж. Принципы разработки программного обеспечения. — М.: Мир, 1982. — 368 с.

3. Гантер Р. Методы управления проектированием программного обеспечения. — М.: Мир, 1981. — 388 с.

4. Брукс Фредерик. Мифический человеко-месяц или как создаются программные системы. — СПб.: Символ, 2000. — 298 с.

5. Вендров А.М. CASE-технологии. Современные методы и средства проектирования информационных систем. — М.: Финансы и статистика, 1998. — 176 с.

6. Канер С., Фолк Дж., Нгуен Е.К. Тестирование программного обеспечения. — Киев: Диасофт, 2000. — 544 с.

Дополнительная:

1. Вольховер В.Г., Иванов Л.А. Производственные методы разработки программ. — М.: Финансы и статистика, 1983. — 236 с.

2. Боем Б. и др. Характеристики качества программного обеспечения. — М.: Мир, 1981. — 176 с.

3. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++. — М.: Бином; СПб.: Невский диалект, 1999. — 560 с.

ПРИЛОЖЕНИЕ А
Оформление курсового проекта

Федеральное агентство по образованию

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

НАЗВАНИЕ ПРОЕКТА

Пояснительная записка к курсовому проекту по дисциплине
«Теория разработки программного обеспечения»

Выполнил: ст. группы ХХХ

_____ Иванов И.И.

«____» _____ 2007

Проверил: доц. каф. АСУ

_____ Елизаров А.И.

«____» _____ 2007

Томск – 2007

(С новой страницы)

РЕФЕРАТ

Курсовой проект состоит из XX страниц, содержит XX таблиц и XX рисунков.

В результате проведенной работы была разработана техническая документация к программному проекту «НАЗВАНИЕ ПРОЕКТА».

Пояснительная записка к курсовому проекту выполнена в редакторе MS Word.

(С новой страницы)

СОДЕРЖАНИЕ

Примечание. Пункты 5 и 6 являются необязательными. В содержание также включаются, как минимум, пункты второго уровня.

ПРИЛОЖЕНИЕ Б
Пример выполнения курсового проекта № 1

1 Техническое задание

1.1 Введение

Наименование разрабатываемого программного обеспечения — переносимая программа трансляции данных по различным протоколам (Data Retranslation, DR).

Данное программное обеспечение применяется для перенаправления HTTP, FTP, SSL и других запросов и данных с клиентской машины через промежуточную машину на другие вышестоящие proxy-серверы. Выбор вышестоящего proxy-сервера осуществляется в соответствии с ранее определенными приоритетами.

Настоящее программное обеспечение может применяться в организациях, располагающих несколькими каналами выхода в Internet. DR должен позволять автоматическое переключение на резервный (менее приоритетный канал) в том случае, если основной канал является недоступным.

1.2 Основания для разработки

Разработка ведется на основании следующих документов:
1. Данное техническое задание

1.3 Назначение разработки

Функциональное и эксплуатационное назначение программы:

Данная программа призвана осуществлять перенаправления запросов от клиентов на вышестоящие proxy-серверы в соответствии с определенными для них приоритетами, а также доступностью или недоступностью того или иного сервера. Программа также позволяет достигать некоторой степени анонимности при работе в сети.

1.4 Технические требования к программе или программному изделию

1.4.1 Требования к функциональным характеристикам

– Программа должна полностью поддерживать стандарты передачи гипертекста (HTTP) версий 1.0 и 1.1, утвержденные World Wide

Web Consortium (W3C), а так же стандартные протоколы FTP, SSL, SMTP, POP3 и т.д.

– Программа должна обеспечивать переносимость в рамках операционных систем семейства Windows. Стандарт, предназначенный для достижения переносимости программного обеспечения на уровне исходных кодов.

– Программа должна работать по архитектуре «клиент-сервер», поддерживать несколько одновременных соединений.

– Программа должна считывать основные настройки из конфигурационного файла, осуществлять это во время работы, без остановки передачи данных.

– Конфигурационный файл должен быть легко читаем для человека, занимающегося администрированием proxy-сервера.

– Программа должна выбирать подходящий вышестоящий proxy-сервер, на который следует перенаправить запрос в соответствии с его приоритетом, определенным в конфигурационном файле, и его текущим статусом (доступен или недоступен).

– Программа должна осуществлять проверку вышестоящих proxy-серверов на работоспособность. Это должно осуществляться в фоновом процессе, без прерывания выполнения других операций передачи данных.

– Программа должна поддерживать передачу нескольких запросов в рамках одного соединения (pipelining).

– Программа должна вести журнал своей деятельности, куда будут сохраняться все сообщения об ошибках, нарушениях передачи и прочих проблемах.

1.4.2 Требования к надежности

– Программа должна при считывании конфигурационного файла корректно обрабатывать его отсутствие, поврежденность и некорректность введенных в него данных. В случае ошибки соответствующая запись должна быть создана в журнале работы программы и выведено предупреждение на экран.

– Программа должна обеспечивать устойчивое функционирование в течение минимум 48 часов.

1.4.3 Требования к эксплуатации

Никаких требований к условиям эксплуатации не выдвигается. Для обслуживания требуется один квалифицированный системный администратор.

1.4.4 Требования к составу и параметрам технических средств

– Для эксплуатации разрабатываемого программного обеспечения необходимы Windows-совместимая операционная система (Windows 98, WinNT 4.0, WinNT 5.0, WinNT 5.1) и компьютер архитектуры, поддерживаемой этой ОС.

– Необходим сетевой адаптер, обеспечивающий связь с Internet.

1.4.5 Требования к информационной и программной совместимости

Язык программирования — C или C++.

1.5 Требования к программной документации

– В дистрибутиве программного средства должно присутствовать полное описание процедуры установки программы.

– Необходимо также составить синтаксис описания конфигурационного файла, а также снабдить дистрибутив примером оформления этого файла.

1.6 Технико-экономические показатели

Программа является узкоспециализированной, более простой в использовании по сравнению с аналогами (WinGate, WinProxy), а также менее требовательной к системным ресурсам и времени.

Предполагается, что внедрение такой программы и обучение персонала обойдется в гораздо меньшую сумму, чем другие аналогичные некоммерческие и коммерческие разработки.

1.7 Стадии и этапы разработки

1. Ознакомление со стандартами и протоколами, анализ схожих существующих программных средств.

2. Разработка концептуальной модели функционирования будущей программы.

3. Разработка эскизного проекта программного средства и согласование его с заказчиком.

4. Непосредственная разработка законченного программного средства (рабочий проект).

5. Отладка и тестирование.
6. Внедрение.

2 Соглашения о требованиях

2.1 Описание программного изделия

2.1.1 Наименование и шифры изделия

2.1.1.1 Полное наименование изделия
Переносимая программа трансляции данных по различным протоколам (Data Retranslation, DR).

2.1.1.2 Сокращенные наименования
DR.

2.1.1.3 Шифры изделия
Отсутствуют.

2.1.1.4 Шифры проекта
Отсутствуют.

2.1.2 Краткое описание изделия

Данное программное обеспечение применяется для перенаправления HTTP, FTP, SSL и других запросов и данных с клиентской машины через промежуточную машину на другие вышестоящие proxy-серверы. Выбор вышестоящего proxy-сервера осуществляется в соответствии с ранее определенными приоритетами.

2.1.3 Сведения об авторском праве

Не требуются.

2.1.4 Результирующие компоненты изделия

Результирующие компоненты изделия перечислены в таблице 2.1.

Таблица 2.1 — Результирующие компоненты изделия

Обозначения:

Основное изделие — не используется для создания других изделий

Вспомогательное изделие — используется для создания других изделий

Уровень поддержки 1: удовлетворяются заявки на исправление дефектов; возможно сообщение об изменениях; принимаются заявки на расширение функциональных возможностей изделия

Уровень поддержки 2: удовлетворяются заявки на исправление дефектов; возможно сообщение об изменениях; заявки на расширение не принимаются

Уровень поддержки 3: удовлетворяются заявки на исправление дефектов

Р — группа разработки

	Формируется целиком	Модифицируется	Распространяется	Не распространяется	Ответственная группа
Спецификации					
Внешняя спецификация	X			X	Р
Внутренняя спецификация	X			X	Р
Спецификация испытаний (не надо)					
Спецификация сопровождения (не надо)					
Другие спецификации					
Документация					
Техническое описание системы					
Справочное руководство					
Справочный буклет					
Руководство оператора	X		X		Б
Указатель системных сообщений					
Информационный листок выпуска					
Другие печатные издания					
Рекламные материалы					

Тип изделия			Начальный уровень поддержки	
	Основное			
	Вспомогательное	X		
			1	X
			2	

Окончание табл. 2.1

	3							
			Программное обеспечение					
			Листинги					
			Исходные модули	X			X	P
			Объектные модули					
			Контрольные примеры	X			X	P
			Средства разработки					
			Прочие средства					

2.2 Цели

Поставленной задачей было написание программы представляющей собой переносимый многопоточный proxy-сервер, осуществляющий перенаправление любых запросов на другие вышестоящие proxy-серверы.

2.2.1 Согласование заявок на проверку

2.2.1.1 Отклоненные заявки
Отсутствуют.

2.2.1.2 Принятые заявки
Отсутствуют.

2.2.2 Согласование заявок на расширение функциональных возможностей изделия

2.2.2.1 Отклоненные заявки
Отсутствуют.

2.2.2.2 Принятые заявки
Отсутствуют.

2.2.3 Согласование заявок на внесение исправлений

2.2.3.1 Отклоненные заявки
Отсутствуют.

2.2.4 Согласование планов

2.2.4.1 Исключенные пункты плана
Отсутствуют.

2.2.4.2 Включенные пункты плана
Отсутствуют.

2.2.5 Перечень требований пользователя

Заказчику требуется:

– Программа, которая должна служить в качестве proxy-сервера, осуществляющего перенаправление запросов на другие вышестоящие proxy-серверы.

– Программа должна работать под любой операционной системой семейства Windows (начиная с версии 3.1).

– Программа должна поддерживать многопоточность, т.е. одновременно обрабатывать несколько запросов от пользователей.

– Программа должна поддерживать изменение пользовательских установок без остановки работы сервера, т.е. перечитывание конфигурационного файла без остановки уже запущенных передач файлов.

2.2.6 Рассмотренные альтернативы

Среди операционных систем семейства Windows наибольшее распространение получили два proxy-сервера: WinGate и WinProxy. Они были отклонены вследствие следующих причин:

1) Эти продукты являются универсальными, вследствие чего громоздки, сложны в использовании и требовательны к системным ресурсам и времени, что влечет за собой большие затраты на внедрение такой программы и обучение персонала.

2) Эти программы используют не совсем корректный алгоритм для определения работоспособности того или иного вышестоящего сервера, вследствие чего все запросы могут уходить в никуда, т.к. не-

работоспособный proxy-сервер имеет более высокий приоритет, нежели другой, работоспособный.

2.2.7 Окупаемость капиталовложений

Капиталовложений нет.

2.3 Стратегии

2.3.1 Соглашения относительно представления материала

2.3.1.1 Обозначения
В данном документе не используется никаких специальных обозначений.

2.3.1.2 Терминология
Вся специальная терминология определяется в контексте данного документа.

2.3.2 Генерируемое программное обеспечение

Не используется.

2.3.3 Системное программное обеспечение

Программа состоит из конфигурационного блока и блока обработки данных (рис. 2.1).

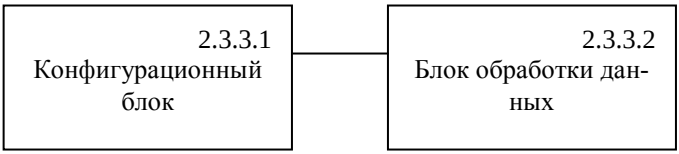

Рис. 2.1 — Функциональные модули программы

2.3.3.1 Характеристики конфигурационного блока
2.3.3.1.1 Внешние ограничения
2.3.3.1.1.1 Действующие стандарты
Не используются.
2.3.3.1.1.2 Ограничения на совместимость

Не существует программных изделий, совместимых с данным программным продуктом.

2.3.3.1.1.3 Программные ограничения

Данная программа работает с любыми операционными системами семейства Windows.

2.3.3.1.1.4 Аппаратные ограничения

Для эксплуатации разрабатываемого программного обеспечения необходимы Windows-совместимая операционная система и компьютер архитектуры, поддерживаемой этой ОС.

Необходим сетевой адаптер, обеспечивающий связь с Internet.

2.3.3.1.2 Внешние характеристики

2.3.3.1.2.1 Результаты работы конфигурационного блока

Проверка конфигурационного файла на противоречивость и передача обработанных данных блоку обработки данных.

2.3.3.1.2.2 Процессы конфигурационного блока

Конфигурационный блок производит лексический разбор данных из конфигурационного файла, проверку их корректности и осуществляет безопасную синхронизацию этими данными между блоком обработки и собой.

2.3.3.1.2.3 Входы конфигурационного блока

Конфигурационный файл, оформленный в соответствии с синтаксисом описания конфигурационного файла, приведенным в руководстве оператора.

2.3.3.1.3 Эргономические характеристики

2.3.3.1.3.1 Безопасность и секретность

Изменение конфигурационного файла возможно только с правами администратора данной ЭВМ.

Никакие входные данные (будь то некорректный конфигурационный файл или некорректный запрос пользователя) не должны приводить к сбою системы.

2.3.3.1.3.2 Надежность

Любой набор входных данных (описанный выше) не должен приводить к сбою системы. В случае некорректности конфигурационного файла продолжает действовать старая конфигурация (если имеется). В случае некорректности запроса, пользователю отправляется соответствующее сообщение, вместо ответа от вышестоящего proxy-сервера.

Реализуется контроль типа вводимых пользователем данных и пороговый контроль их значений.

2.3.3.1.3.3 Рестарт

В программе отсутствует информация, которую необходимо периодически сохранять для возобновления работы после отказа блока

питания. Все необходимые для работы данные содержатся в конфигурационном файле, который из самой программы не изменяется.

2.3.3.1.3.4 Соответствие требованиям заказчика

Система должна соответствовать требованиям технического задания.

2.3.3.1.3.5 Рабочие характеристики

Изделие не накладывает никаких ограничений на конфигурацию, помимо ограничений, определяемых оборудованием.

2.3.3.1.3.6 Обеспечение эксплуатации

Никаких требований к условиям эксплуатации не выдвигается.

Для обслуживания требуется один квалифицированный системный администратор.

Работа программного изделия начинается с запуска исполняемого файла. По умолчанию программа работает в режиме сервиса (системная служба). Всю информацию о работе программы можно посмотреть, воспользовавшись службой syslog. Пример настройки syslog можно посмотреть в руководстве оператора.

Для завершения работы программы необходимо послать соответствующий сигнал сервису из командной строки. Например: TASKKILL /PID 1230 /T. При этом все установленные соединения будут разорваны.

2.3.3.1.3.7 Мобильность

Данный программный продукт распространяется с открытым исходным кодом.

2.3.3.1.4 Внутренние характеристики

2.3.3.1.4.1 Удобство сопровождения

Программа может быть скомпилирована с ключом DEBUG (-DDEBUG), что дает разработчику или человеку, занимающемуся сопровождением данного ПО, достаточно полную информацию о том, чем занимается в данный момент программа.

2.3.3.1.3.2 Алгоритмы

Подлежат описанию во внутренней спецификации.

2.3.3.2 Характеристики блока обработки данных

Все пропущенные пункты см. в п. 2.3.3.1.

2.3.3.2.2 Внешние характеристики

2.3.3.2.2.1 Результаты работы блока обработки данных

Обработка запросов от пользователей, перенаправление запроса на один из вышестоящих proxy-серверов и передача полученной информации обратно пользователю.

2.3.3.2.2.2 Процессы блока обработки данных

Блок обработки данных представляет собой три процесса:

1) Первый процесс занимается поддержанием небольшого пула потоков. Ограниченное количество потоков (задаваемое в конфигурационном файле) постоянно является активным, чтобы минимизировать временные задержки при обработке поступающих запросов.

2) Второй процесс осуществляет прием соединений от пользователей и назначение им того или иного потока из пула.

3) Третий процесс также работает параллельно с первыми двумя и проводит фоновую проверку работоспособности вышестоящих proxy-серверов.

После создания нового потока для принятого соединения идет передача информации между клиентом и наиболее приоритетным вышестоящим proxy-сервером.

В рамках одного соединения может быть передано несколько запросов.

2.3.3.2.2.3 Входы блока обработки данных

1) Данные, полученные от конфигурационного блока.

2) Данные об установленном соединении, передаваемые программе от операционной системы.

3) Текст запросов, получаемый от клиентов.

4) Текст ответов, получаемый от вышестоящих proxy-серверов.

2.3.4 Внутренние ограничения

Единственные ограничения, которые накладываются на данное ПО, связаны с ограничениями операционной системы (максимальное количество сокетов, потоков, количество открытых файлов и т.д.), а также возможностями данной аппаратной системы (количество оперативной памяти).

Программа может иметь более одной своей копии в оперативной памяти только в том случае, если каждая из них слушает свой порт (имеется в виду номер TCP-порта), при этом при запуске каждой копии программы указывается свой конфигурационный файл (в качестве ключа командной строки).

2.4 Используемые материалы

2.4.1 Справочные документы

– Стандарты передачи гипертекста (HTTP) версий 1.0 и 1.1, утвержденные World Wide Web Consortium (W3C), Стандарты передачи файлов (FTP) версий 1.0 и 1.1, утвержденные World Wide Web Consortium (W3C), SSL и т.д.

– Стандарт, предназначенный для достижения переносимости программного обеспечения на уровне исходных кодов. Windows-стандарт стандартизирован ANSI (American National Standards Institute) и ISO (International Standards Organisation).

2.5 Передача заказчику и ввод в действие

2.5.1 Ресурсы, обеспечивающие ввод в действие

Любой квалифицированный системный администратор Windows, имеющий опыт работы более 6 месяцев (или при эквивалентном обучении), сможет осуществить ввод в действие DR.

Для генерации ПО требуется Windows-совместимая операционная система и любой Windows-совместимый компилятор C++.

2.5.2 Носители информации

В качестве носителей информации используется дискета емкостью 1.44 Мб.

2.6 Тактика

2.6.1 Взаимосвязи

2.6.1.1 Требуемые взаимосвязи
Не выдвигаются требования другим изделиям.
2.6.1.2 Обеспечения взаимосвязи
Требования других изделий к программе отсутствуют.

2.6.2 Техническая ревизионная комиссия (ТРК)

Создание ревизионной комиссии не требуется.

2.6.3 Проверка изделия

2.6.3.1 Уровни испытаний
Уровни испытаний приведены в таблице 2.2.

Таблица 2.2 — Уровни испытаний

Категория испытаний	Класс испытаний		
	A	B	C
Демонстрация в действии		Р	
Аттестация			
Полная функциональная проверка		И	
Проверка новых свойств			
Эксплуатационные испытания		И	
Испытания надежности		И	
Проверка устойчивости			
Возвратная проверка			
Пусковые испытания			
Испытания конфигураций		Р	
Режимы испытаний: I — проводятся группой испытаний ( ) II — контролируются группой испытаний (X) III — группа испытаний не участвует ( )			
Подразделения, проводящие испытания: Р — группа разработки И — группа испытаний О — группа обслуживания / — испытания исключены			

2.6.3.2 Эталоны для сравнения

Отсутствуют.

2.6.4 Обеспечение внедрения

2.6.4.1 Мероприятия, обеспечивающие продвижение программного изделия на рынок

Не предусматриваются.

2.6.4.2 Мероприятия, связанные с обучением

Обслуживающий персонал должен быть знаком с операционными системами семейства Windows, существующими средствами сборки ПО (компиляторами), общепринятой структурой конфигурационных файлов Windows. Специальное обучение не проводится. В случае возникновения каких-либо вопросов, они могут быть разрешены посредством online-консультаций или очных встреч с разработчиками ПО.

2.6.4.3 Средства, обеспечивающие модернизацию программного изделия

Не предусматриваются.

2.7 Календарный план

Календарный план работы представлен в таблице 2.3.

Таблица 2.3 — Календарный план

Этапы работы	Дата начала	Дата окончания	Финансирование
Анализ требований	18.09.06	25.09.06	3000 руб. (15%)
Спецификации	26.09.06	07.10.06	2000 руб. (10%)
Проектирование	08.10.06	15.10.06	2000 руб. (10%)
Кодирование	16.10.06	15.11.06	8000 руб. (40%)
Тестирование	16.11.06	30.11.06	5000 руб. (25%)

3 Спецификации

3.1 Внешняя спецификация

```
main: procedure
  do steps:
    «Открыть, прочитать настройки»:
      ReadSettings();
    «Выполнение»:
      Start();
    do case:
      «Пауза»: Suspend();
      «Завершение выполнения»: Stop();
    end;
  end;
end main.
```

3.2 Внутренняя спецификация

```
/* Обработка подключения */
Function StartUserThread (указатель на
  настройки, указатель на текущий сокет)
  declare Result число
  if проверка имени пользователя и пароля
     прошла успешно then do
  begin
    Внести в список текущий сокет, IP-адрес,
      логин и пароль как цепочку.
```

```
    Продолжение работы с пользователем,
      обработка его запросов.
  else do Сообщение об ошибке доступа.
  Set result as Correct Connection;
End;

/* Ожидание подключений */
procedure WaitForConnection()
  declare HTHREAD HANDLE;
  Ожидание подключений пользователей.
  if соединение успешно then
  begin
    Set HTHREAD =CreateThread(StartThread).
    Внести HTHREAD в список.
End;

/* Приостановка всех потоков */
BOOL Function SuspendThreads()
  if список потоков не пустой then
    for all thread in list do SuspendThread();
/* Продолжение всех потоков */
BOOL Function ResumeThreads()
  if список потоков не пустой then
    for all thread in list do ResumeThread();

/* Остановка всех потоков */
BOOL Function TerminateThreads()
  if список потоков не пустой then
    for all thread in list do
      TerminateThread();
```

4 Тестирование

Для проведения тестирования программы «Переносимая программа транслирования данных по различным протоколам» составим классы эквивалентности входных данных (табл. 4.1).

Таблица 4.1 — Классы эквивалентности входных данных

Входные условия	Классы эквивалентности	
	Правильные	Неправильные
Host name, IP-address	Тип string(20) («*.*.*.*» или «*») (1)	Не удовлетворяет маске ввода (2)
Port	Тип Word (3)	Не принадлежит типу Word (4)
Start	Тип Boolean (5)	Не принадлежит типу Boolean (6)

Классы 1,3,5: вводимые данные

«192.168.0.2», 3129, TRUE

Создан процесс «транслятор», главная форма программы свернута в Tray.

Ожидаемый результат: тот же.

Класс 2: вводимые данные

93, 3129, TRUE

Результат: программа выводит сообщение «Error read data. Try another name» и закрывается.

Ожидаемый результат: тот же.

Класс 4: вводимые данные

«192.168.0.2», 124123123, TRUE

Результат: программа выводит сообщение «Error. This value is not WORD value» и закрывается.

Ожидаемый результат: тот же.

Класс 6: вводимые данные

«192.168.0.2», 3129, 3

Результат: программа выводит сообщение «Error. This not a valid Boolean value» и закрывается.

Ожидаемый результат: тот же.

Вывод. Результаты тестирования программы не выявили ошибок, по результатам тестирования делаем вывод, что программа работает корректно.

5 Руководство системного программиста

5.1 Общие сведения о программе

Данное программное обеспечение применяется для перенаправления HTTP, FTP, SSL и других запросов и данных с клиентской машины через промежуточную машину на другие вышестоящие proxy-серверы. Выбор вышестоящего proxy-сервера осуществляется в соответствии с ранее определенными приоритетами.

Это консольное приложение может функционировать на технических средствах под управлением операционной системы семейств Windows NT. Минимальными требованиями для выполнения программы являются система, поддерживающая Windows NT, 1,5 Мб места на жёстком диске, клавиатура. Программа поставляется в архиве, содержащем программу, исходный код на языке C++.

5.2 Структура программы

Переносимая программа транслирования данных по различным протоколам состоит из следующих компонентов:

1) kblc_proxy.exe — исполняемый модуль;

2) kblc_proxy.cpp — исходный код программы на языке C++.

Данная программа не требует установки каких-либо дополнительных библиотек или приложений.

5.3 Настройка программы

5.3.1 Установка программы

Распакуйте архив в выбранную папку.

5.3.2 Настройка программы

Не нуждается в настройке.

5.4 Проверка программы

Проверка программы производится в следующем порядке:

1. Запуск приложения.
2. Ввод необходимых параметров для начала работы.

Пример: «192.168.0.2», 3129, TRUE

3. Результат: программа запускается и сворачивается в Tray.

4. Проверка полученных результатов: в браузере Internet Explorer устанавливается proxy-сервер с параметрами «192.168.0.2:3129». После обращения к доступному до выбора данных настроек ресурсу вы увидите тот же ресурс.

5. Если результаты работы программы Internet Explorer до настройки и после совпадают, то проверка считается успешно завершенной.

6. При получении диагностических и иных сообщений в ходе проверки программы следует обращаться к разделу «Сообщения системному программисту» данного руководства.

5.5 Дополнительные возможности

Программа не обладает дополнительными возможностями.

5.6 Сообщения системному программисту

В таблице 5.1 представлены сообщения, которые может получить системный программист в ходе выполнения настройки, проверки программы, а также пользователь в ходе выполнения программы. Описаны содержание этих сообщений и действия системного программиста, которые необходимо предпринять по этим сообщениям.

Табл. 5.1 — Сообщения системному программисту и пользователю

Сообщение	Описание	Действия системного программиста
Error read data. Try another Name	Нет такого имени компьютера или оно введено неправильно	Ввести другое имя компьютера
Error. This value is not WORD value	Номер порта вышел за пределы диапазона 1..65535	Введите другое значение порта
Error. This not a valid Boolean value	Параметр START введён неправильно (требуется тип BOOLEAN)	Ввести правильное значение параметра START
Error. Host is unreachable	В ходе работы программы последующий узел связи стал недоступен	Проверить подключения до следующего узла

ПРИЛОЖЕНИЕ В
Пример выполнения курсового проекта № 2

1 Техническое задание

1.1 Введение

Программа «День рождения» служит для реализации функций системы управления данных и служит для выполнения задач накопления, хранения, навигации и обработки информации, занесенной в базу. Таковой информацией является дата происшествия события, описание события, телефон и адрес (опционально), дата внесения события в базу. Программа может быть использована на персональных компьютерах в качестве альтернативного органайзера или как дополнение к стандартным программам-органайзерам.

1.2 Основания для разработки

Разработка программы велась на основе задания выданного доцентом кафедры АСУ Горитовым А.Н. на курсовое проектирование по дисциплине «Структуры и алгоритмы обработки данных» в октябре 2004 г.

1.3 Назначение разработки

1.3.1 Функциональное назначение программы

Программа должна реализовывать следующие возможности:

– Работать под управлением операционной системы MS-DOS.

– Корректно отображать информацию о предстоящих событиях в соответствии с определенными в программе правилами.

– Заносить в базу информацию о событии (дата происхождения события, краткое описание события, адрес человека, связанного с событием (опционально), телефон этого человека (опционально), дата внесения события в базу).

– Позволять редактировать любое поле данных.

– Сортировать поля по различным ключам по убыванию или возрастанию.

– Производить поиск данных по содержимому любого из полей.

– Осуществлять удаление любого поля.

– Иметь интуитивный интерфейс.

– Иметь возможность настраивать параметры работы программы.

1.3.2 Эксплуатационное назначение программы

Программа предназначена для напоминания пользователю о важных для него событиях, которые должны произойти в интервале времени от текущего дня до заданного в программе значения (7 дней). Кроме того, программа имеет базу данных, в которой хранится информация о важных событиях (см. выше). Разрешается добавлять, редактировать, удалять информацию о событиях и, кроме того, осуществлять сортировку и поиск. Программа может использоваться в качестве оригинального или альтернативного органайзера на персональных компьютерах.

1.4 Требования к программе или программному изделию

1.4.1 Требования к функциональным характеристикам

Входная информация в программе представлена в виде файла, содержащего в себе некоторое количество одинаковых записей, каждая из которых состоит из следующих полей:

– дата события;
– краткое описание события;
– адрес;
– телефон;
– дата внесения события в базу.

Поля записей представлены следующими типами данных и должны удовлетворять нижеописанным условиям:

1. Дата события: состоит из трех чисел:
– День — целое число в интервале от 1 до 31;
– Месяц — целое число в интервале от 1 до 12;
– Год — целое число в интервале от 1900 до 2100.

2. Краткое описание события — строковая переменная длиной от 1 до 40 символов. Нулевая длина этой строки считается ошибочной.

3. Адрес — строковая переменная длиной от 0 до 40 символов. Нулевая длина соответствует отсутствию адреса и заменяется соответствующим значением.

4. Телефон — строковый параметр длиной от 0 до 13 символов. Нулевая длина параметра соответствует отсутствию телефона и заменяется соответствующим значением.

5. Дата внесения события в базу — состоит из трех чисел типа integer, значения которых берутся из системных часов.

Программа выполняет следующие функции:

– Открытие файла базы данных.

– Сверка значений поля «Дата события» каждой записи с текущей датой и отображение поля «Описание события» для записей, удовлетворяющих условию отображения.

– Просмотр записей, содержащихся в базе. Навигация осуществляется с помощью клавиш Up, Down, Page Up, Page Down, Home, End.

– Добавление новой записи в базу (клавиша Insert).

– Удаление выбранной записи/записей из базы данных (клавиша Delete).

– Сортировка записей в базе данных по различным ключам (клавиша F2).

– Поиск записи по значению какого-либо ключа (клавиша F3).

– Краткая справка, описывающая принципы работы с программой (клавиша F1).

Выходная информация в программе представлена в виде набора строк на экране монитора, характеризующих предупреждение о предстоящих событиях, а также отображающих содержимое базы данных событий. Кроме того, к входной информации относятся предупреждения и сообщения об ошибках, выдаваемые программой (например, сообщение о невозможности открыть файл, о невозможности записать информацию в файл или предупреждение о неправильном значении одного из полей при заполнении формы добавления записи в базу).

1.4.2 Требования к надежности

Программа реализует анализ входных данных для предотвращения ввода заведомо ложных значений, что может привести к сбоям в работе. Имеется анализатор ошибок, который выдает описание ошибки при проблемах с записью в файл базы данных или чтением из него. При сбое, файл имеющейся на момент открытия базы данных не повреждается. Теряются только не сохраненные записи. Новые записи сохраняются при выходе из программы.

1.4.3 Условия эксплуатации

Условия эксплуатации должны соответствовать типовым условиям эксплуатации персональных компьютеров. Пользователь должен

иметь навык работы с компьютером. Никаких специальных навыков от пользователя не требуется.

1.4.4 Требования к составу и параметрам технических средств

Для корректной работы программы достаточно компьютера следующей конфигурации:

– Процессор Intel Pentium 66 или эквивалентный ему.

– Не менее 100 Кб свободного места на жестком диске для самой программы, а также место для файла базы данных (количество записей в файле * длину записи, где длина одной записи ~ 600 б).

– Устройства ввода (клавиатура, мышь).

1.4.5 Требования к информационной и программной совместимости

Программа работает под управлением операционной системы MS-DOS, Windows 95/98/Me/NT/2000/XP.

Язык написания программы — Pascal. Компилятор — Borland Turbo Pascal 7.0.

Защита информации, хранящейся в базе данных, не производилась по причине отсутствия необходимости.

1.4.6 Требования к программной документации

Программа сопровождается кратким руководством пользователя, поясняющим принципы работы с программой. Руководство доступно для просмотра как из программы, так и извне ее, с помощью любого текстового редактора.

1.4.7 Стадии и этапы разработки

Разработка описываемой программы состояла из следующих этапов:

– Определение общих методов решения задачи.

– Построение предметной области.

– Реализация предметной области программными средствами.

– Работы по устранению ошибок и отладке программы.

Выполнение этих этапов было произведено в октябре — декабре 2004 г.

2 Соглашение о требованиях

2.1 Описание программного изделия

2.1.1 Наименования и шифры изделия

2.1.1.1 Полное наименование изделия.
Программа «День рождения».
2.1.1.2 Сокращенное наименование изделия
ДР.
2.1.1.3 Шифры изделия
Отсутствуют.
2.1.1.4 Шифры проекта.
Отсутствуют.

2.1.2 Краткое описание изделия

Программа предназначена для напоминания пользователю о важных для него событиях, которые должны произойти в интервале времени от текущего дня до заданного в программе значения (7 дней). Кроме того, программа имеет базу данных, в которой хранится информация о важных событиях. Разрешается добавлять, редактировать, удалять информацию о событиях и, кроме того, осуществлять сортировку и поиск. Программа может использоваться в качестве оригинального или альтернативного органайзера на персональных компьютерах.

2.1.3 Сведения об авторском праве

Отсутствуют.

2.1.4 Результирующие компоненты изделия

Результирующие компоненты изделия приведены в таблице 2.1.

Таблица 2.1 — Результирующие компоненты изделия

Обозначения: Основное изделие — не используется для создания других изделий Вспомогательное изделие — используется для создания других изделий Уровень поддержки 1: удовлетворяются заявки на исправление дефектов; возможно сообщение об изменениях; принимаются заявки на расширение функциональных возможностей изделия Уровень поддержки 2: удовлетворяются заявки на исправление дефектов; возможно сообщение об изменениях; заявки на расширение не принимаются Уровень поддержки 3: удовлетворяются заявки на исправление дефектов Р — группа разработки						Формируется целиком	Модифицируется	Распространяется	Не распространяется	Ответственная группа
					Спецификации					
					Внешняя спецификация	X			X	Р
					Внутренняя спецификация	X			X	Р
					Спецификация испытаний (не надо)					
					Спецификация сопровождения (не надо)					
					Другие спецификации					
					Документация					
					Техническое описание системы					
					Справочное руководство					
					Справочный буклет					
					Руководство оператора	X		X		Б
Тип изделия	Основное	X	Начальный уровень поддержки		Указатель системных сообщений					
	Вспомогательное				Информационный листок выпуска					
			1	X	Другие печатные издания					
			2		Рекламные материалы					

Окончание табл. 2.1

	3							
			Программное обеспечение	X		X		P
			Листинги	X			X	P
			Исходные модули	X			X	P
			Объектные модули					
			Контрольные примеры	X			X	P И
			Средства разработки					
			Прочие средства					

2.2 Цели

Данная программа была выполнена в рамках курсового проектирования по дисциплине «Системы и алгоритмы обработки данных» в октябре—декабре 2004 г. Требовалось написать программу, реализующую функции системы управления базой данных.

2.2.1 Согласование заявок на проверку

2.2.1.1 Отклоненные заявки
Отсутствуют.
2.2.1.2 Принятые заявки
Отсутствуют.

2.2.2 Согласование заявок на расширение функциональных возможностей изделия

2.2.2.1 Отклоненные заявки
Отсутствуют.
2.2.2.2 Принятые заявки
Отсутствуют.

2.2.3 Согласование заявок на внесение исправлений

2.2.3.1 Отклоненные заявки
Отсутствуют.

2.2.4 Согласование планов

2.2.4.1 Исключенные пункты плана
Отсутствуют.
2.2.4.2 Включенные пункты плана
Отсутствуют.

2.2.5 Требования заказчика

Заказчиком является доцент кафедры АСУ А.Н. Горитов. Заказчику требуется обеспечение следующих функций:

– Представление данных внутри программы должно реализовываться динамическими структурами (списками).
– Программа должна быть написана на языке программирования Pascal.
– Программа должна работать под управлением операционной системы MS-DOS.
– Программа должна обрабатывать стандартные ошибки и обеспечить проверку входных данных.

На изготовление программы отводится 3 месяца.

2.2.6 Рассмотренные альтернативы

Отсутствуют.

2.6.7 Окупаемость капиталовложений

Программа создавалась в учебных целях, поэтому капиталовложения в нее отсутствуют.

2.3 Стратегия

2.3.1 Стратегия относительно предоставляемого материала

2.3.1.1 Обозначения
В данном документе не используется никаких специальных обозначений.
2.3.1.2 Терминология
В данном документе используется общепринятая терминология. Никаких специальных терминов не используется.

2.3.2 Генерируемое программное обеспечение

Не используется.

2.3.3 Системное программное обеспечение

Программа состоит из трех частей: интерфейс, блок обработки данных (БОД), блок записи/чтения данных (БЗЧ) — см. рис. 2.1.

Рис. 2.1 — Функциональные модули программы

2.3.3.1 Общие характеристики функции «Интерфейс»

2.3.3.1.1 Внешние ограничения

2.3.3.1.1.1 Действующие стандарты

ЕСПД и совместимый с ним стандарт предприятия ОС ТУСУР 6.1 97.

2.3.3.1.1.2 Ограничения на совместимость

Не существует программных изделий или баз данных, совместимых с программой «ДР». Файлы баз данных, генерируемые программой «ДР», не могут использоваться другими программами.

2.3.3.1.1.3 Программные ограничения

Программа «ДР» способна работать под управлением операционных систем MS-DOS, Windows 9x/NT/2000/XP.

2.3.3.1.1.4 Аппаратные ограничения

– Процессор Intel Pentium 66 или эквивалентный ему.

– Не менее 100 Кб свободного места на жестком диске для самой программы, а также место для файла базы данных (количество записей в файле * длину записи, где длина одной записи ~ 600 б).

– Устройства ввода (клавиатура, мышь).

См. таблицу 2.2.

Таблица 2.2 — Аппаратные требования

Наименование	Минимальное количество	Оптимальное количество	Максимальное количество
Место на НЖМД	100 Кб + N*R	Не ограничено	Не ограничено
Мышь	0	1	1
Клавиатура	1	1	1
Процессор	Pentium 66 МГц	Pentium II 300 МГц	Не ограничено
ОЗУ	8 Мб	24 Мб	Не ограничено

Здесь:

– N — количество записей в файле базы данных;

– R — длина одной записи.

2.3.3.1.2 Внешние характеристики

2.3.3.1.2.1 Результаты

Результатами работы блока «Интерфейс» являются следующие данные:

– Отображение на экране монитора в виде строковых параметров результаты работы БОД.

– Проверка полей новых записей, вводимых с клавиатуры, на корректность и передача верных значений БОД.

– Выдача предупреждения при вводе некорректных значений полей.

2.3.3.1.2.2 Процессы

Блок «Интерфейс» выполняет следующие процессы:

– Преобразование в удобное для понимания представление (набор строковых значений, выводимых на экран монитора) данных, обработанных БОД.

– Считывание значений полей новых записей, вводимых пользователем с клавиатуры, их преобразование, проверка на корректность и передача в БОД.

– Обработка нажатия «горячих» клавиш и клавиш навигации.

2.3.3.1.2.3 Входы

Блок «Интерфейс» имеет следующие входы:

– Значения полей записей, поступающих из БОД. Значения могут иметь тип string или integer.

– Значения типа char, считанные с клавиатуры.

2.3.3.1.3 Эргономические характеристики

2.3.3.1.3.1 Безопасность и секретность системы

Работа с записями, хранящимися в файле базы данных, возможна только из программы «ДР». Записи, введенные с клавиатуры, записываются в ОЗУ и сохраняются на диск по завершению работы с программой «ДР».

Никаких шагов по обеспечению секретности не осуществлялось, так как это не является целью работы.

2.3.3.1.3.2 Надежность

При вводе значений с клавиатуры реализуется проверка их корректности. В случае ввода некорректного значения поля на экран выводится предупреждение, а введенная информация не передается в БОД, что исключает возможность внутренней ошибки в БОД при работе с блоком «Интерфейс».

В случае возникновения сбоя, не связанного с программой, будут потеряны все не сохраненные записи.

2.3.3.1.3.3 Рестарт

При сбое все несохраненные на жесткий диск данные будут потеряны. Поэтому, если работа с программой не была завершена или была завершена неверно, ввод данных придется повторить снова.

2.3.3.1.3.4 Соответствие требованиям заказчика

Программа «ДР» должна удовлетворять требованиям, описанным в техническом задании.

2.3.3.1.3.5 Рабочие характеристики

Программа «ДР» не накладывает никаких ограничений на конфигурацию, помимо ограничений, определяемых аппаратной и программной частью.

2.3.3.1.3.6 Удобство эксплуатации

Работать с программой может любой пользователь, имеющий любой уровень квалификации.

Взаимодействие программы и пользователя происходит в диалоговом режиме. Для удобства ввода, редактирования, сортировки данных используются «горячие» клавиши.

Для начала работы с программой достаточно запустить исполняемый модуль. Завершение работы выполняется горячей клавишей, после нажатия на которую программа сама выполнит необходимые операции.

2.3.3.1.3.7 Мобильность

Программа «ДР» без модификации способна работать с операционными системами MS-DOS и Windows любой версии вплоть до XP.

2.3.3.1.4 Внутренние характеристики

2.3.3.1.4.1 Удобство сопровождения

Дополнительных модулей для установки данного комплекса на любом ЭВМ с Intel-совместимым процессором не требуется.

2.3.3.1.4.2 Алгоритмы.

Подлежат описанию во внутренней спецификации.

2.3.3.2 Общие характеристики функций «Блок обработки данных»

Для всех пропущенных разделов см. соответствующие разделы п. 2.3.3.1.

2.3.3.2.1 Внешние характеристики

2.3.3.2.1.1 Результаты работы блока обработки данных

Линейный динамический двусвязный список, состоящий из записей, содержащихся в файле базы данных.

2.3.3.2.1.2 Процессы

– присваивание списка по значениям записей, поступившим из БЧЗ;

– добавление элемента по информации, поступившей из блока «Интерфейс» в список;

– поиск элемента в списке по ключу, поступившему из блока «Интерфейс»;

– сортировка элемента списка по ключу и флагу, поступившим из блока «Интерфейс»;

– удаление элемента из списка, по команде из блока «Интерфейс»;

– изменение значения элемента по данным, поступившим из блока «Интерфейс».

2.3.3.2.1.3 Входы

– записи, прочтенные БЧЗ из файла базы данных;

– новые записи, полученные из блока «Интерфейс»;

– команды на выполнение какого-либо процесса, полученные из блока «Интерфейс».

2.3.3.2.1.4 Внутренние ограничения

1. Отсутствие возможности распечатки.

2. Ограничения на размерность данных:

– день, месяц — целое от 0 до 255 (byte);

– год — целое от 0 до 65535 (integer);

– краткое описание события, адрес — строка длиной от 1 до 40 символов (string[40]);

– телефон — строка длиной от 0 до 13 символов (string[13]).

2.3.3.3 Общие характеристики функций «Блок чтения/записи данных»

Для всех пропущенных разделов см. соответствующие разделы п. 2.3.1.1.

2.3.3.3.1 Внешние характеристики

2.3.3.3.1.1 Результаты работы

– запись, прочитанная из файла базы данных;

– файл базы данных, представляющий собой набор записей, несущих в себе информацию о событиях.

2.3.3.3.1.2 Процессы

– чтение записей из файла базы данных;

– сохранений записей в файл базы данных с перезаписью.

2.3.3.3.1.3 Входы

– файл, состоящий из набора записей, описывающих события;

– элементы линейного списка, поступающие из блока обработки данных.

2.3.3.3.1.4 Внутренние ограничения

Записи, прочитанные из файла, должны соответствовать типу записи, определенной в данной программе.

2.3.3.3.3 Эргономические характеристики

2.3.3.3.3.2 Надежность

При открытии файла проверяется соответствие типа записи типу, определенному в программе «ДР». При несоответствии типов выводится сообщение об ошибке. Сообщение об ошибке выводится также при отсутствии файла. Это позволяет избежать внутренней ошибки в БОД.

2.4 Используемые материалы

2.4.1 Справочные документы

Отсутствуют.

2.5 Передача заказчику и ввод в действие

2.5.1 Средства защиты прав собственности на изделие

Соблюдение прав собственности не требуется.

2.5.2 Ресурсы, обеспечивающие ввод в действие

Допустимая квалификация для ввода программы «ДР» в действие — низкая. Не требуется никаких специальных навыков.

2.5.3 Носители информации

В качестве носителя информации используется дискета емкостью 1.44 Мб.

2.6 Тактика

2.6.1 Взаимосвязи

2.6.1.1. Требуемые взаимосвязи

Не предъявляется никаких требований к другим программным изделиям.

2.6.1.2 Обеспечиваемые взаимосвязи

Никакое другое программное обеспечение не накладывает требований на программу «ДР».

2.6.2 Техническая ревизионная комиссия

Создание ревизионной комиссии не требуется.

2.6.3 Проверка изделия

2.6.3.1 Уровни испытаний

См. таблицу 2.3.

Таблица 2.3 — Уровни испытаний

Категория испытаний	Класс испытаний		
	A	B	C
Демонстрация в действии		P	
Аттестация			
Полная функциональная проверка			
Проверка новых свойств			
Эксплуатационные испытания		P	
Испытания надежности		P	
Проверка устойчивости		P	
Возвратная проверка			
Пусковые испытания			
Испытания конфигураций			
Режимы испытаний: I — проводятся группой испытаний ( ) II — контролируются группой испытаний (X) III — группа испытаний не участвует ( )			

Окончание табл. 2.3

Подразделения, проводящие испытания:
Р — группа разработки
И — группа испытаний
О — группа обслуживания
/ — испытания исключены

2.6.4 Обеспечение поддержки

2.6.4.1 Мероприятия, обеспечивающие продвижение программного обеспечения на рынок

Не производятся.

2.6.4.2 Мероприятия, связанные с обучением

Не производятся.

3 Спецификации

3.1 Внешние спецификации

```
main: procedure
  declare char string;
  char = Выбранная команда;
  do Case(char)
  /* Подпрограмма создания нового элемента
     списка */
    «Создать элемент»: call AddItem;
  /* Подпрограмма редактирования имеющегося
     элемента списка */
    «Редактировать элемент»: call EditItem;
  /* Подпрограмма удаления имеющегося элемента
     списка */
    «Удалить элемент»: call DelItem;
  /* Подпрограмма поиска элемента списка
     по ключу */
    «Поиск»: call FindItem;
  /* Подпрограмма сортировки элементов списка
     по событию */
    «Сортировка по названию события»:
      call SortItemsByEvents;
  /* Подпрограмма сортировки элементов списка
     по времени события */
    «Сортировка по дате события»:
```

```
      call SortItemsByDoE;
  /* Подпрограмма сортировки элементов списка
     по времени добавления в базу */
    «Сортировка по дате внесения в базу»:
     call SortItemsByDoA;
  /* Вызов краткого руководства по функциям
     базы данных */
    «Подсказка»: call About;
  /* Подпрограмма завершения работы */
    «Выход»: call FinWork;
  end case;
end main.
```

3.2 Внутренние спецификации

```
/* Процедура, выполняющая добавление нового
   элемента в список */
Procedure AddItem;

/* Процедура, выполняющая редактирование
   элемента списка, находящегося под
   курсором */
Procedure EditItem;

/* Процедура, выполняющая удаление элемента
   списка, находящегося под курсором */
DelItem;

/* Процедура, выполняющая поиск элемента
   списка по заданному ключу */
FindItem;

/* Процедура, выполняющая сортировку элементов
   списка по названию события */
SortItemsByEvents;

/* Процедура, выполняющая сортировку элементов
   списка по дате события */
SortItemsByDoE;

/* Процедура, выполняющая сортировку элементов
   списка по дате их внесения в базу */
```

```
  SortItemsByDoA;

  /* Процедура, выводящая на экран краткую
     подсказку по функциям программы
     и информацию о разработчиках */
  About;

  /* Процедура, осуществляющая сохранение списка
     в файл базы данных и завершающая работу с
     программой */
FinWork;
```

4 Тестирование

4.1 Обоснование уровня испытаний

Для проведения тестирования было решено провести испытания класса B. Испытания класса B выполняются независимо от группы разработки и начинаются после того, как разработчики объявляют, что изделие готово к передаче потребителю. Испытания данного класса были выбраны потому, что программа была написана задолго до проведения тестирования и, следовательно, проведение испытаний класса A было невозможно. Так как программа не предназначена на продажу, то проводить испытания класса C также не представляется возможным.

Режим испытаний — II, так как группу испытаний интересует только анализ результатов испытаний, а составление плана, спецификации испытаний, построение тестов и их прогонки поручаются разработчику.

4.1.1 Чтение записей из файла и составление списка

В каталоге, содержащем загрузочный модуль программы «День рождения», должен находиться файл «name.dat», содержащий в себе записи базы данных. В случае отсутствия этого файла при запуске программы на экран выводится сообщение об ошибке. Иначе из записей, хранящихся в файле, создается динамический двусвязный список.

4.1.2 Добавление записи

При добавлении новой записи на экран выводится форма, содержащая ряд полей. Обязательные поля:

1. Событие — строковая переменная, состоящая из любых символов и имеющая длину от 1 до 40 символов.

2. Дата события в следующем формате:

– день — целое число в интервале от 1 до 31;
– месяц — целое число в интервале от 1 до 12;
– год — целое число в интервале от 1900 до 2100.

Необязательные поля:

1. Адрес — строковая переменная, состоящая из любых символов и имеющая длину от 1 до 40 символов или пустая. Пустое значение автоматически заменяется строкой «Нет данных».

2. Телефон — строковая переменная, состоящая из любых символов и имеющая длину от 1 до 13 символов или пустая. Пустое значение автоматически преобразуется в значение «Нет данных».

4.1.3 Правка полей записи, находящейся под курсором

При исправлении одного или нескольких полей записи на экран выводится форма, содержащая ряд полей. Набор полей идентичен набору полей в форме добавления новой записи (см. Добавление записи). Возможна правка значений всех полей формы.

4.1.4 Поиск записи по ключу

При поиске элемента на экран выводится форма, содержащая ряд полей. Набор полей идентичен набору полей в форме добавления нового элемента (см. «Добавление записи»). Поиск записи может производиться по значению одного из полей или по любой их совокупности.

4.6 Классы эквивалентности

Правильные классы эквивалентности:

1. Файл name.dat существует (1).
2. Значение поля «Событие» введено (2).
3. Значение поля «События» — строковая переменная длиной от 1 до 40 символов (3).
4. Значение поля «День» введено (4).
5. Значение поля «День» — целое число в интервале от 1 до 31 (5).
6. Значение поля «Месяц» введено (6).
7. Значение поля «Месяц» — целое число в интервале от 1 до 12 (7).
8. Значение поля «Год» введено (8).

9. Значение поля «Год» — целое число в интервале от 1900 до 2100 (9).

10. Значение поля «Адрес» — строковая переменная длиной от 1 до 40 символов или пустое значение (10).

11. Значение поля «Телефон» — строковая переменная длиной от 1 до 13 символов или пустое значение (11).

Неправильные классы эквивалентности

1. Файл name.dat не существует (12).

2. Значение поля «Событие» не введено (13).

3. Значение поля «Событие» — строковая переменная длиной более 40 символов (14).

4. Значение поля «День» не введено (15).

5. Значение поля «День» — число, меньшее 1 (16).

6. Значение поля «День» — число, большее 31 (17).

7. Значение поля «День» — строковая переменная (18).

8. Значение поля «День» — дробное число (19).

9. Значение поля «Месяц» не введено (20).

10. Значение поля «Месяц» — число, меньшее 1 (21).

11. Значение поля «Месяц» — число, большее 12 (22).

12. Значение поля «Месяц» — строковая переменная (23).

13. Значение поля «Месяц» — дробное число (24).

14. Значение поля «Год» не введено (25).

15. Значение поля «Год» — число, меньшее 1900 (26).

16. Значение поля «Год» — число, большее 2100 (27).

17. Значение поля «Год» — строковая переменная (28).

18. Значение поля «Год» — дробное число (29).

19. Значение поля «Адрес» — строковая переменная длиной более 40 символов (30).

20. Значение поля «Телефон» — строковая переменная длиной более 13 символов (31).

4.7 Тесты

4.7.1Тест для правильных классов эквивалентности

Заполнение полей формы правильными значениями (охватывает классы 1—11):

- Событие: День рождения Ивана Ивановича
- День: 15
- Месяц: 5
- Год: 1968

– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: поля заносятся в запись, запись добавляется к списку, происходит отображение обновленной информации на экране.

4.7.2 Тесты для неправильных классов эквивалентности

1. В директории, содержащей исполняемый модуль, отсутствует файл name.dat (класс 12).

Результат: Выводится системное сообщение об ошибке «File Not Found». Программа завершает работу.

2. Поле «Событие» не введено (класс 13):

– Событие:
– День: 15
– Месяц: 5
– Год: 1968
– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: ошибка; поле «Событие» принимает значение «Нет данных».

3. Значение поля «Событие» — строковая переменная длиной более 40 символов (класс 14):

– Событие: День рождения Владимира Ильича Ленина — верного друга всех октябрят
– День: 15
– Месяц: 5
– Год: 1968
– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: в поле «Событие» будут содержаться лишь первые сорок символов введенной фразы, благодаря проверке на длину строки, которая отсечет лишние символы.

4. Значение поля «День» не введено (класс 15):

– Событие: День рождения Ивана Ивановича
– День:
– Месяц: 5
– Год: 1968
– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: После ввода пустого значения не произошел переход курсора на другое поле, так как введено неправильное значение. Ожидается ввод корректного значения.

5. Значение поля «День» — число, меньшее 1 (класс 16):

– Событие: День рождения Ивана Ивановича
– День: 0
– Месяц: 5
– Год: 1968
– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: после ввода некорректного значения произведется очистка поля, и курсор остается в текущей позиции, ожидая ввода правильного значения.

6. Значение поля «День» — число, большее 31 (класс 17):

– Событие: День рождения Ивана Ивановича
– День: 55
– Месяц: 5
– Год: 1968
– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: после ввода некорректного значения произведется очистка поля, и курсор остается в текущей позиции, ожидая ввода правильного значения.

7. Значение поля «День» — строковая переменная (класс 18):

– Событие: День рождения Ивана Ивановича
– День: три
– Месяц: 5
– Год: 1968
– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: неправильное значение не будет введено, так как содержимым этого поля не могут быть нецифровые символы.

8. Значение поля «День» — дробное число (класс 19):

– Событие: День рождения Ивана Ивановича
– День: 2.5
– Месяц: 5
– Год: 1968
– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: неправильное значение не будет введено, так как содержимым этого поля не могут быть нецифровые символы.

9. Значение поля «Месяц» не введено (класс 20):

– Событие: День рождения Ивана Ивановича
– День: 15
– Месяц:
– Год: 1968
– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: После ввода пустого значения не произошел переход курсора на другое поле, так как введено неправильное значение. Ожидается ввод корректного значения.

10. Значение поля «Месяц» — число, меньшее 1 (класс 21):

– Событие: День рождения Ивана Ивановича
– День: 15
– Месяц: 0
– Год: 1968
– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: после ввода некорректного значения произведется очистка поля, и курсор остается в текущей позиции, ожидая ввода правильного значения.

11. Значение поля «Месяц» — число, большее 12 (класс 22):

– Событие: День рождения Ивана Ивановича
– День: 15
– Месяц: 64
– Год: 1968
– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: после ввода некорректного значения произведется очистка поля, и курсор остается в текущей позиции, ожидая ввода правильного значения.

12. Значение поля «Месяц» — строковая переменная (класс 23):

– Событие: День рождения Ивана Ивановича
– День: 15
– Месяц: май
– Год: 1968
– Адрес: ул. Ленина, 25-54
– Телефон: 33-33-33

Результат: неправильное значение не будет введено, так как содержимым этого поля не могут быть нецифровые символы.

13. Значение поля «Месяц» — дробное число (класс 24):
- Событие: День рождения Ивана Ивановича
- День: 15
- Месяц: 5,5
- Год: 1968
- Адрес: ул. Ленина, 25-54
- Телефон: 33-33-33

Результат: неправильное значение не будет введено, так как содержимым этого поля не могут быть нецифровые символы.

14. Значение поля «Год» не введено (класс 25):
- Событие: День рождения Ивана Ивановича
- День: 15
- Месяц: 5
- Год:
- Адрес: ул. Ленина, 25-54
- Телефон: 33-33-33

Результат: После ввода пустого значения не произошел переход курсора на другое поле, так как введено неправильное значение. Ожидается ввод корректного значения.

15. Значение поля «Год» — число, меньшее 1900 (26)
- Событие: День рождения Ивана Ивановича
- День: 15
- Месяц: 5
- Год: 19
- Адрес: ул. Ленина, 25-54
- Телефон: 33-33-33

Результат: после ввода некорректного значения произведется очистка поля, и курсор остается в текущей позиции, ожидая ввода правильного значения.

16. Значение поля «Год» — число, большее 2100 (класс 27):
- Событие: День рождения Ивана Ивановича
- День: 15
- Месяц: 5
- Год: 35000
- Адрес: ул. Ленина, 25-54
- Телефон: 33-33-33

Результат: после ввода некорректного значения произведется очистка поля, и курсор остается в текущей позиции, ожидая ввода правильного значения.

17. Значение поля «Год» — строковая переменная (класс 28):
- Событие: День рождения Ивана Ивановича
- День: 15
- Месяц: 5
- Год: Вася
- Адрес: ул. Ленина, 25-54
- Телефон: 33-33-33

Результат: неправильное значение не будет введено, так как содержимым этого поля не могут быть нецифровые символы.

18. Значение поля «Год» — дробное число (класс 29):
- Событие: День рождения Ивана Ивановича
- День: 15
- Месяц: 5
- Год: 19.68
- Адрес: ул. Ленина, 25-54
- Телефон: 33-33-33

Результат: неправильное значение не будет введено, так как содержимым этого поля не могут быть нецифровые символы.

19. Значение поля «Адрес» — строковая переменная длиной более 40 символов (класс 30):
- Событие: День рождения Ивана Ивановича
- День: 15
- Месяц: 5
- Год: 1968
- Адрес: ул. Христофора Бонифатиевича Колумба, дом 8
- Телефон: 33-33-33

Результат: в поле «Адрес» будут содержаться лишь первые сорок символов введенной строки, благодаря проверке на длину, которая отсечет лишние символы.

20. Значение поля «Телефон» — строковая переменная длиной более 13 символов (класс 31):
- Событие: День рождения Ивана Ивановича
- День: 15
- Месяц: 5
- Год: 1968
- Адрес: ул. Ленина, 25-54
- Телефон: Please, call me!

Результат: в поле «Телефон» будут содержаться лишь первые сорок символов введенной строки, благодаря проверке на длину, которая отсечет лишние символы.

4.7.3 Результаты тестирования

В ходе тестирования были выявлены следующие недостатки программы:

1. Поле «Событие» может содержать пустое значение, что недопустимо. Для устранения этой проблемы необходимо проверять длину введенного сообщения не только на максимальное, но и на минимальное вводимое количество символов.

2. Отсутствует взаимодействие между различными компонентами даты. В частности, возможно значение даты вида 31.02.1968. Способ устранения: проверять граничные условия не только исходя из значений конкретного числа, но и всей даты в целом.

3. При отсутствии файла работа с программой невозможна. Вариант исправления: при отсутствии файла name.dat в указанной выше папке производить создание нового пустого файла.

4. Ряд полей может содержать значения, не соответствующие действительности из-за ограниченности длины этих полей. Возможное решение: увеличить длину поля, вводить значения, удовлетворяющие указанным выше ограничениям.

5 Руководство системного программиста

5.1 Общие сведения о программе

Программа «День рождения» предназначена для реализации функций системы управления данных и для выполнения задач накопления, хранения, навигации и обработки информации, занесенной в базу. Основными функциями программы является: отображение корректной информации о событии, занесение в базу информации о событии, редактирование любого поля данных, сортировка полей, поиск данных, удаление любого поля, возможность настраивать параметры работы программы.

Технические и программные средства, обеспечивающие выполнение данной программы: процессор Intel Pentium 66 или эквивалентный ему, не менее 100 Кб на жестком диске для самой программы, а также место для файла базы данных, устройства ввода (клавиатура,

мышь). Программа работает под управлением операционной системы MS-DOS, Windows 95/98/Me/NT/ 2000/XP.

5.2 Структура программы

Программа «День рождения» состоит из следующих компонентов:

1. Birthday.pas — исходный код программы;
2. Birthday.exe — загрузочный модуль программы;
3. Name.dat — файл, содержащий в себе записи базы данных.

Данная программа не требует установки каких-либо дополнительных библиотек или приложений.

5.3 Настройка программы

5.3.1 Установка программы

Если в одном каталоге с загрузочным модулем находится файл Name.dat, то запустить файл Birthday.exe.

5.3.2 Настройка программы

Не требует никаких предварительных настроек.

5.4 Проверка программы

Проверка программы производится в следующем порядке:

1) запуск приложения;
2) заполнение полей записи;
3) поиск по ключу;
4) если работа программы закончилась успешно и получено соответствующее сообщение, то проверка считается успешно завершенной и следует выйти из программы;
5) если работа программы завершилась не успешно, то следует ввести другую входную информацию.

При получении диагностических и иных сообщений в ходе проверки программы следует обращаться к разделу «Сообщения системному программисту» данного руководства.

5.5 Дополнительные возможности

Программа не обладает дополнительными возможностями.

5.6 Сообщения системному программисту

Программа выполняет проверку введенных данных в автоматизированном режиме и, если это необходимо, сама исправляет ошибки информации, введенной пользователем.