

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего образования

**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Факультет дистанционного обучения (ФДО)

В. В. Романенко, А. И. Елизаров

ОСНОВЫ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Учебное методическое пособие

Томск 2021

Корректор: А. Н. Миронова

Романенко В. В., Елизаров А. И.

Основы разработки программного обеспечения : учебное методическое пособие / В. В. Романенко, А. И. Елизаров. – Томск : ФДО, ТУСУР, 2021. – 91 с.

© Романенко В. В.
Елизаров А. И., 2021
© Оформление.
ФДО, ТУСУР, 2021

ОГЛАВЛЕНИЕ

Введение.....	6
1 Техническое задание.....	8
1.1 Введение.....	8
1.2 Основания для разработки	9
1.3 Назначение разработки.....	9
1.4 Требования к программе или программному изделию	9
1.4.1 Требования к функциональным характеристикам	10
1.4.2 Требования к надежности	11
1.4.3 Условия эксплуатации.....	11
1.4.4 Требования к составу и параметрам технических средств.....	11
1.4.5 Требования к информационной и программной совместимости.....	12
1.5 Требования к программной документации	12
1.6 Техничко-экономические показатели	12
1.7 Стадии и этапы разработки	13
2 Соглашение о требованиях.....	14
2.1 Описание программного изделия	15
2.1.1 Наименование и шифры изделия	15
2.1.2 Краткое описание изделия	16
2.1.3 Сведения об авторском праве	16
2.1.4 Результирующие компоненты изделия.....	16
2.2 Цели	19
2.2.1 Согласование заявок на проверку	20
2.2.2 Согласование заявок на расширение	20
2.2.3 Согласование заявок на внесение исправлений.....	21
2.2.4 Согласование планов	22
2.2.5 Перечень требований пользователя	23
2.2.6 Рассмотренные альтернативы.....	23
2.2.7 Окупаемость капиталовложений.....	24

2.3 Стратегия.....	24
2.3.1 Соглашения относительно представления материала	24
2.3.2 Генерируемое программное обеспечение	25
2.3.3 Системное программное обеспечение	25
2.3.4 Внутренние ограничения	46
2.4 Используемые материалы	47
2.4.1 Справочные документы.....	47
2.5 Передача заказчику и ввод в действие	48
2.5.1 Средства защиты права собственности на изделие.....	48
2.5.2 Ресурсы, обеспечивающие ввод в действие.....	48
2.5.3 Носители информации.....	49
2.6 Тактика	49
2.6.1 Взаимосвязи.....	50
2.6.2 Техническая ревизионная комиссия	51
2.6.3 Проверка изделия.....	51
2.6.4 Обеспечение поддержки	53
2.7 Извещение об изменении календарных сроков	54
3 Написание спецификаций	56
3.1 Внешняя спецификация.....	57
3.2 Внутренняя спецификация	58
4 Тестирование	60
4.1 Общие принципы тестирования	60
4.2 Организация испытаний программных изделий.....	63
4.3 Виды испытаний программного изделия. Стадии испытаний	64
4.4 Режимы испытаний программ	65
4.5 Категории испытания программного изделия	66
4.6 Технология тестирования, классы эквивалентности.....	68
4.7 Построение тестов.....	71

5 Руководство системного программиста.....	76
5.1 ГОСТ 19.503–79.....	76
5.1.1 Общие положения	76
5.1.2 Содержание разделов	77
5.2 Пример.....	78
5.2.1 Общие сведения о программе	78
5.2.2 Структура программы	79
5.2.3 Настройка программы	80
5.2.4 Проверка программы	81
5.2.5 Дополнительные возможности.....	82
5.2.6 Сообщения системному программисту	82
Литература	84
Приложение А (справочное) Оформление отчета по лабораторной работе.....	86
Приложение Б (справочное) Оформление пояснительной записки	88

ВВЕДЕНИЕ

Выполнение лабораторных работ и курсового проекта (если он предусмотрен учебным планом) является завершающим этапом изучения дисциплины «Основы разработки программного обеспечения». В этот период закрепляются теоретические знания и приобретаются практические навыки разработки программного обеспечения (ПО) и программной документации [1].

Основной задачей практической части дисциплины является разработка программной документации. Комплект отчетных материалов должен включать оттестированное программное обеспечение, соответствующее техническому заданию, и следующие разделы программной документации:

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

Разрабатываемое ПО должно содержать не менее 300 строк кода (не включая код, автоматически генерируемый средой разработки, а также пустые строки и комментарии), работать в многооконном графическом режиме и поддерживать работу клавиатуры и манипулятора типа «мышь». Программа должна обладать достаточным количеством входных данных с проверкой корректности ввода, а также процедурами и функциями для их обработки.

За основу для выполнения лабораторных работ или курсового проекта рекомендуется взять уже готовую программу (результат выполнения контрольной, лабораторной или курсовой работы по ранее изученной дисциплине, рабочий проект и т. п.), удовлетворяющую минимальным требованиям. При этом писать разделы программной документации 1–3 необходимо так, как будто ПО еще только проектируется (т. е. так, как документация

выглядела бы при написании в момент проектирования указанного ПО). Либо можно написать новую программу.

Программная документация должна соответствовать требованиям стандартов Единой системы программной документации (ЕСПД), образовательного стандарта ОС ТУСУР 01–2013 [2] и Положения о проверке самостоятельности выполнения письменных работ бакалавров, специалистов и магистров в ТУСУР [3].

1 ТЕХНИЧЕСКОЕ ЗАДАНИЕ

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.

1.1 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 устройств. Минимизация пути режущего инструмента обеспечивает существенную экономию энергии и трудозатрат.

Наименование: система минимизации пути режущего инструмента при раскрое листовых материалов (далее – минимизатор).

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

1.2 Основания 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. Основанием для разработки является данное техническое задание, выданное 15 сентября 2017 г. заместителем директора Балтийского сталепрокатного завода И. И. Ивановым. Наименование темы разработки – «Минимизатор».

1.3 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.

1.4 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.

1.4.1 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.

1.4.2 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.

1.4.3 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.

1.4.4 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 архитектуры x86 с частотой не ниже 1 ГГц;

- объем ОЗУ не менее 128 Мб;
- не менее 10 Мб свободного места на жестком диске;
- графический адаптер SVGA;
- клавиатура, манипулятор типа «мышь».

1.4.5 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 версии не ниже, чем Microsoft Windows 7. Требование информационной совместимости должно быть обеспечено работой с файлами геометрической информации определенной структуры в качестве входной и выходной информации.

1.5 Требования к программной документации

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

1.6 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.

1.7 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

Ниже дается описание разделов, которые должны присутствовать в любом соглашении о требованиях (СТ) согласно ГОСТ. Все последующие упоминания о соглашении о требованиях имеют в виду СТ, 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 весь пример 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

Кратко и в общих понятиях описываются 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 © 2018 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.

Таблица 2.1 – Результирующие компоненты изделия

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

Окончание таблицы 2.1

Тип изде- лия	Основ- ное	X	Началь- ный уро- вень под- держки	Информационный листок выпуска						
	Вспомо- гатель- ное			Другие печатные издания						
				Рекламные матери- алы						
			1	X	Программное обес- печение					
			2		Листинги	X			X	P
			3		Исходные модули	X			X	P
					Объектные модули	X				P
					Контрольные при- меры	X			X	P И
					Средства разра- ботки				X	
					Прочие средства					

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» указывается «Р», «И» и др. – по ключу на бланке. Если предполагается выпуск нестандартных изделий, этот факт отмечается в графе «Другие спецификации» и описание соответствующих документов дается сразу после таблицы.

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 сделать 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 Telcoscore 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 либо цитируется определенное место из каждого 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 также приводится предполагаемый срок использования изделия. Обычно это 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 предназначена для специалистов по финансовому анализу или других лиц с аналогичными аналитическими задачами, таких 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 программ, поэтому всегда следует 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 (используются понятия, соответствующие целевому назначению организации).

Пример. Фирма ABC Services ожидает, что объем сбыта в финансовой сфере увеличится на 10% в течение 3 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, $(a_1, b_1) \dots (a_2, b_2)$, 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 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).

Пример:

2.3.N Генерируемое (системное) программное обеспечение

Программа состоит из трех модулей (рис. 2.1):

- общие функции ASK;
- интерфейс системы;
- ядро системы.

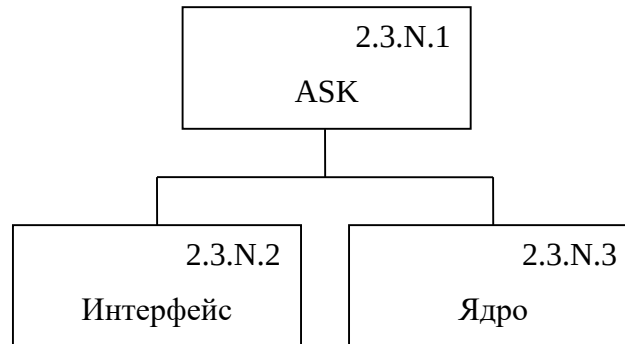


Рисунок 2.1 – Структурная схема из соглашения о требованиях для изделия ASK

2.3.N.1 Общие характеристики функций ASK

2.3.N.1.1 Внешние ограничения ASK

И т. д. (здесь N = 2 или 3).

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. Однако, 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 ABC 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 наиболее 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	не ограничено
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 аварийного 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. Также указывается, 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 программы; 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

Для всех пропущенных подразделов см. соответствующие подразделы раздела 2.3.3.1.

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 автоматическая и ручная процедуры рестарта на основе использования этих данных.

1. Автоматический рестарт. При использовании вспомогательного блока питания M107 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 обработана 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 Telcoscore 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. Требуется 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	1	2
M107 Резервный блок питания	1	1	1
M1100 Модуль памяти	1	2	не ограничено
M3100 Дисковый модуль	1	3	не ограничено
M210 Пульт	1	1	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, откуда заимствуется максимально возможное количество программ. Процессор корректировок

должен работать с не меньшей скоростью, чем программа UPDATE, в случае сравнимых операций.

2.3.3.3.6 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 то, 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, 13.06.2017 (проект), разд. 5;

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

в) Спецификация содержания и формата DATABASE, 1230711, редакция 07.02.2017;

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, характеризующаяся 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 диагностическими испытательными средствами, такими как файл ошибок или средства профилактического контроля в режиме онлайн.

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		
	А	В	С
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 В 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 каналов 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 В 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 от-носительное 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	Примеча- ния
П10	Утверждение бюджетной за- явки	29.09.2017		
P10	Определение назначения из- делия и требований к нему	03.11.2017		
П20	Утверждение назначения и требований к изделию	15.12.2017		
P20	Составление внешней специ- фикации	09.01.2018		
И10	Утверждение плана испыта- ний	09.02.2018		
P30	Утверждение внешней специ- фикации	15.03.2018	06.02.2018	
O10	Установка аппаратуры, необ- ходимой для разработки	31.03.2018		
Д10	Утверждение плана под- держки	02.03.2018		

P41	Демонстрация в действии	15.05.2018		
И30	Начало испытаний класса В	03.07.2018	08.05.2018	
Д20	Рассылка рекламных материалов	15.07.2018		
Д21	Подготовка учебных пособий	01.08.2018		
С20	Подготовка спецификации сопровождения	15.08.2018		
О20	Начало распространения изделия	01.09.2018	17.07.2018	
Подготовил С. Девис Утвердил Старая дата 15.09.2017		Утвердил К. Андерсен Утвердил Новая дата 13.01.2018		

3 НАПИСАНИЕ СПЕЦИФИКАЦИЙ

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 разработки 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.

3.1 Внешняя спецификация

Пример. Далее приводится пример оформления внешних спецификаций на программу «Электронный каталог» на языке PDL [1].

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

  if FileName не найден then
  begin
    put('Ошибка открытия файла');
    call Create(File);
  end;
  else
    call Open(FileName, File);

  if length(File)=0 then
    put('Файл не содержит записей');

  do while (кнопка≠«Выход»)
  do case (кнопка)
    // Вывод содержимого файла на экран
    «Просмотр»: call View(File);
    // Поиск записи в файле
    «Поиск»: call Search(File);
    // Добавление записи
```

```

    «Добавить»: call Add(File);
    // Удаление записи из файла
    «Удалить»: call Del(File);
    // Выход из программы
    «Выход»: call Exit;
end;
end;
end main.

```

3.2 Внутренняя спецификация

Пример. Далее приводится пример оформления внутренней спецификаций на программу «Электронный каталог» на языке PDL [1].

```

procedure Create(File);
begin
    File = создание нового файла;
end Create;

procedure Open(FileName, File);
begin
    File = чтение файла FileName;
end Create;

procedure View(File);
begin
    do while not EOF(File)
    begin
        get(File, запись);
        put(запись);
    end;
end View;

procedure Search(File);
begin

```

```
get(искемое значение);  
do while not EOF(File)  
begin  
    get(File, запись);  
    if запись = искоемое значение then  
        put(Name, Album, Genre, Year);  
    end;  
end Search;
```

```
procedure Add(File);  
begin  
    get(Name);  
    get(Album);  
    get(Genre);  
    get(Year);  
    put(File, запись);  
end Add;
```

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

```
procedure Exit;  
begin  
    выход из программы;  
end Del;
```

4 ТЕСТИРОВАНИЕ

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

4.1 Общие принципы тестирования

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

IF (Выражение) N₁, N₂, N₃

Критерий п. 1 подразумевает, что **IF** должен быть выполнен, в то время как п. 2 подразумевает различные наборы данных, для того чтобы выполнялись условия N₁, N₂, N₃ (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. Естественное 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 хорошо 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 серьезном 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 их 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, а 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 программного обеспечения можно выполнять по технологии черного ящика с использованием методологии эквивалентного разбиения [1]. 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 групп. Для проведения этой операции используют таблицу, подобную таблице 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 число, может принимать значения от 1 до 999»), то определяются один правильный класс эквивалентности значений (от 1 до 999) и несколько неправильных (значения целого числа, меньшие 1, значения целого числа, большие 999, и т. д.).

2. Если входное условие описывает строку определенной длины (например, «строка длиной от 1 до 999 символов»), то определяются один правильный класс эквивалентности значений (строка длиной от 1 до 999 символов) и два неправильных (пустая строка, строка длиннее 999 символов).

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

4. 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 как минимум один неправильный класс эквивалентности (например, «Неизвестный способ передвижения»).

5. 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).

6. 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:

$$\text{DIMENSION } arr [, arr...],$$

где arr 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)
Нижняя граница определена	да (15), нет (16)	
Нижняя граница (НГ)	константа (17), целая переменная (18)	имя элемента массива (19), что-то иное (20)
Имя целой переменной, задающей НГ	содержит буквы (21) и цифры (22)	состоит из других символов (23)
Целая переменная, задающая НГ, начинается с буквы	да (24)	нет (25)
Константа, задающая НГ	–65534..65535 (26)	< –65534 (27), > 65535 (28)
Значение константы, задающей НГ	< 0 (29), 0 (30), > 0 (31)	
Верхняя граница (ВГ)	константа (32), целая переменная (33)	имя элемента массива (34), что-то иное (35)
Имя целой переменной, задающей ВГ	содержит буквы (36) и цифры (37)	состоит из других символов (38)
Целая переменная, задающая ВГ, начинается с буквы	да (39)	нет (40)
Константа, задающая ВГ	–65534..65535 (41)	< –65534 (42), > 65535 (43)
Значение константы, задающей ВГ	< 0 (44), 0 (45), > 0 (46)	

Окончание таблицы 4.2

Входные условия	Классы эквивалентности	
	Правильные	Неправильные
Верхняя граница по отношению к нижней границе	больше (47), равна (48)	меньше (49)
Оператор расположен на нескольких строках	да (50), нет (51)	

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

```
DIMENSION A(2)
```

покрывает классы 1, 4, 7, 10, 12, 16, 32, 41, 46, 47 и 51. Далее определяются один или более тестов, покрывающих оставшиеся правильные классы эквивалентности. Так, тест

```
DIMENSION A12345(I:9, J4XXXX:65535, KLM, 0:N2),
B(-65534:100, 0:100, 10:65535, 10:10, -100:0, -100:-1, 0:0),
C(65535:65535, -65534:-65534)
```

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

- (3) DIMENSION
- (5) DIMENSION (10)
- (6) DIMENSION A123456(10)
- (9) DIMENSION A.I(10)
- (11) DIMENSION 1A(10)
- (13) DIMENSION A
- (14) DIMENSION A(10,10,10,10,10,10,10,10)
- (19) DIMENSION A(B(2):10)
- (20) DIMENSION A(.:10)
- (23) DIMENSION A(I.J:10)
- (25) DIMENSION A(1J:10)
- (27) DIMENSION A(-65535:10)
- (28) DIMENSION A(65536:10)

- (34) DIMENSION A(10,B(2))
- (35) DIMENSION A(10,.)
- (38) DIMENSION A(I.J,10)
- (40) DIMENSION A(10,1J)
- (42) DIMENSION A(-65535)
- (43) DIMENSION A(65536)
- (49) DIMENSION A(10:1)

Эти классы эквивалентности покрываются 22 тестами. Хотя эквивалентное разбиение значительно лучше случайного выбора тестов, оно все же имеет недостатки (т. е. пропускает определенные типы высокоэффективных тестов). Другие два метода, описанные в учебном пособии – анализ граничных значений и использование функциональных диаграмм (диаграмм причинно-следственных связей cause-effect graphing) – свободны от многих недостатков, присущих эквивалентному разбиению. Их также можно использовать для тестирования написанной программы.

В отчете или пояснительной записке приводится описание классов эквивалентности, тестирование правильных и неправильных классов, результаты тестирования, а также даются рекомендации по устранению ошибок, выявленных в результате тестирования.

5 РУКОВОДСТВО СИСТЕМНОГО ПРОГРАММИСТА

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 (Microsoft Internet Explorer, Google Chrome, Mozilla Firefox, Apple Safari 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 том числе 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 программы (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_результатирующего_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 может 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

Окончание таблицы 5.1

Сообщение	Описание	Действия системного программиста
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
...	...	...

ЛИТЕРАТУРА

Основная литература

1. Калайда, В. Т. Технология разработки программного обеспечения : учеб. пособие / В. Т. Калайда, В. В. Романенко. – Томск : ТМЦДО, 2007. – 236 с.
2. Образовательный стандарт вуза ОС ТУСУР 01–2013. Работы студенческие по направлениям подготовки и специальностям технического профиля. Общие требования и правила оформления [Электронный ресурс]. – Режим доступа: <https://regulations.tusur.ru/documents/70> (дата обращения: 21.05.2020).
3. Положение о проверке самостоятельности выполнения письменных работ бакалавров, специалистов и магистров в ТУСУР [Электронный ресурс]. – Режим доступа: <https://regulations.tusur.ru/documents/81> (дата обращения: 21.05.2020).

Дополнительная литература

1. Боэм, Б. Характеристики качества программного обеспечения / Б. Боэм, Дж. Браун, Х. Каспар, М. Липов, Г. Мак-Леод, М. Меррит. – М. : Мир, 1981. – 208 с.
2. Брукс, Ф. Мифический человеко-месяц, или Как создаются программные системы / Ф. Брукс, Х. Чапел. – СПб. : Символ-Плюс, 2010. – 304 с.
3. Буч, Г. Объектно-ориентированный анализ и проектирование с примерами приложений / Г. Буч, Р. А. Максимчук, М. У. Энгл, Б. Дж. Янг, Д. Коналлен, К. А. Хьюстон. – М. : Вильямс, 2017. – 720 с.
4. Вендров, А. М. CASE-технологии. Современные методы и средства проектирования информационных систем / А. М. Вендров. – М. : Финансы и статистика, 1998. – 176 с.

5. Вольховер, В. Г. Производственные методы разработки программ / В. Г. Вольховер, Л. А. Иванов. – М. : Финансы и статистика, 1983. – 236 с.
6. Гантер, Р. Методы управления проектированием программного обеспечения / Р. Гантер. – М. : Мир, 1981. – 392 с.
7. Зелковиц, М. В. Принципы разработки программного обеспечения / М. В. Зелковиц, А. С. Шоу, Дж. Д. Гэннон. – М. : Мир, 1982. – 364 с.
8. Канер, С. Тестирование программного обеспечения / С. Канер, Дж. Фолк, Е. К. Нгуен. – Киев : Диасофт, 2000. – 544 с.

ПРИЛОЖЕНИЕ А**(справочное)****Оформление отчета по лабораторной работе**

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра автоматизированных систем управления (АСУ)

ТЕМА ПРОЕКТАОтчет по лабораторной работе №__ по дисциплине
«Основы разработки программного обеспечения»

Обучающийся гр. _____

_____ И. О. Фамилия

« ____ » _____ 20__ г.

Руководитель: канд. техн. наук,
и.о. зав. каф. АСУ

_____ В. В. Романенко

« ____ » _____ 20__ г.

Томск 20__

Оглавление

1 Техническое задание	X
2 Соглашение о требованиях.....	X
3 Спецификации	X
4 Тестирование	X
5 Руководство системного программиста.....	X
6 Заключение	X
Список использованных источников	X

Примечания:

1. Если учебным планом предусмотрена только одна лабораторная работа, ее номер на титульном листе допускается не указывать.

2. Если учебным планом предусмотрены две лабораторные работы, разделы 1–5 распределяются между ними (рекомендуется включать разделы 1–2 в отчет по лабораторной работе № 1, разделы 3–5 – в отчет по лабораторной работе № 2).

3. В оглавление также включаются разделы отчета 2–3-го уровней.

ПРИЛОЖЕНИЕ Б**(справочное)****Оформление пояснительной записки**

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования

**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра автоматизированных систем управления (АСУ)

ТЕМА ПРОЕКТА

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

Обучающийся гр. _____

_____ И. О. Фамилия

« ____ » _____ 20__ г.

Руководитель: канд. техн. наук,
и.о. зав. каф. АСУ

_____ В. В. Романенко

« ____ » _____ 20__ г.

Томск 20__

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования

**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра автоматизированных систем управления (АСУ)

ЗАДАНИЕ

на курсовое проектирование по дисциплине

«Основы разработки программного обеспечения»

Студенту _____

группы _____ факультета _____

1. Тема курсового проекта: _____

2. Срок сдачи студентом законченного проекта: «____» _____ 20__ г.

3. Цель проекта: _____

4. Задачи проекта: _____

5. Исходные данные: _____

6. Дата выдачи задания: «____» _____ 20__ г.

Задание выдал:

к.т.н., и.о. зав. каф. АСУ _____

В. В. Романенко

Задание принял к исполнению: «____» _____ 20__ г.

студент гр. _____

И. О. Фамилия

Реферат

Пояснительная записка к курсовому проекту состоит из XX страниц, содержит XX таблиц, XX рисунков, XX источников.

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

Пояснительная записка к курсовому проекту выполнена в (*указать название и версию программы*).

Оглавление

1 Техническое задание	X
2 Соглашение о требованиях.....	X
3 Спецификации	X
4 Тестирование	X
5 Руководство системного программиста.....	X
6 Заключение	X
Список использованных источников	X

Примечания:

1. Курсовой проект выполняется только в том случае, если это предусмотрено учебным планом.

2. Если наряду с курсовым проектом в учебном плане предусмотрены лабораторные работы, в пояснительную записку включаются результаты выполнения лабораторных работ и устраняются оставшиеся замечания. Также в этом случае допускается раздел «5 Руководство системного программиста» не включать в отчет по лабораторной работе № 2.

3. В оглавление также включаются разделы пояснительной записки 2–3-го уровней.