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)

М. Ю. Катаев

УЧЕБНО-ИССЛЕДОВАТЕЛЬСКАЯ РАБОТА

**Учебно-методическое пособие для студентов
направления подготовки 09.03.01 «Информатика
и вычислительная техника» (уровень бакалавриата), профиль
«Программное обеспечение средств вычислительной техники
и автоматизированных систем», обучающихся с применением
дистанционных образовательных технологий**

Томск 2021

Корректор: А. Н. Миронова

Катаев М. Ю.

Учебно-исследовательская работа : учебно-методическое пособие для студентов направления подготовки 09.03.01 «Информатика и вычислительная техника» (уровень бакалавриата), профиль «Программное обеспечение средств вычислительной техники и автоматизированных систем» обучающихся с применением дистанционных образовательных технологий / М. Ю. Катаев. – Томск : ФДО, ТУСУР, 2021. – 309 с.

© Катаев М. Ю., 2021
© Оформление.
ФДО, ТУСУР, 2021

ОГЛАВЛЕНИЕ

Введение..... 6
1 Подготовка к проведению научного исследования..... 11
1.1 Объектная область, объект и предмет 11
1.2 Тема, проблема и актуальность исследования..... 12
1.3 Изучение научной литературы и уточнение темы 14
1.4 Элементы исследований..... 17
1.5 Определение методов исследования..... 19
2 Проведение научного исследования 24
2.1 Этапы научного исследования..... 24
2.2 Информационная проработка темы 27
2.3 Виды и методика проведения информационного поиска..... 32
3 Оформление научно-исследовательской работы..... 34
3.1 Оформление результатов исследовательской работы 34
3.2 Правила оформления списка литературы 37
3.2.1 Схема библиографического описания документа для списка литературы 38
3.2.2 Схема аналитического описания документа для списка литературы 42
4 Защита результатов исследования 57
4.1 Доклад 57
4.2 Подготовка мультимедийных презентаций 61
5 Программный проект..... 65
5.1 Виды программного обеспечения и их характеристики..... 65
5.2 Жизненный цикл программного средства..... 77
5.3 Техническая документация..... 82
5.3.1 Подготовка документации на программные средства в соответствии с ГОСТом..... 84
5.3.2 Стандарты комплекса ГОСТ 34..... 96

6 Универсальный язык моделирования UML 102
6.1 Общие сведения об UML 102
6.2 Краткое описание UML 113
6.3 UML-диаграммы 118
6.3.1 Основные понятия диаграмм классов UML 120
6.3.2 Связи-ассоциации 125
6.3.3 Сущности 130
6.3.4 Отношения 134
6.4 Краткое описание работы в Umbrello UML Modeller 136
6.4.1 Пользовательский интерфейс 136
6.4.2 Создание, загрузка и сохранение моделей 138
6.4.3 Импорт и генерация кода 146
7 Разработка технического задания 148
7.1 Общие сведения о техническом задании 148
7.2 Рекомендации по оформлению ТЗ 153
7.3 Примеры технических заданий 172
7.3.1 Техническое задание на разработку программного продукта построения графика функций одного аргумента 172
7.3.2 Техническое задание на разработку программного продукта учета успеваемости студентов 174
7.3.3 Пример разработки и внедрения информационно-аналитической системы 178
8 Принципы проектирования программных систем 184
8.1 Общие сведения о проектировании программных систем 184
8.2 Понятие архитектуры программного обеспечения 186
8.3 Объектная модель 188
8.4 Анализ и проектирование программного обеспечения 191
8.5 Технологические аспекты проектирования программного обеспечения 201
8.6 Объектная декомпозиция 205

9 Методические указания к выполнению учебно-исследовательской работы 211
9.1 Задания для выполнения учебно-исследовательской работы 213
9.1.1 Задания для УПД-1 213
9.1.2 Задания для УПД-2 214
9.1.3 Задания для УПД-3 215
9.1.4 Задания для УПД-4 233
9.2 Примеры заданий по проекту УИР 234
Литература 242
Приложение А ГОСТ 19.701–90 (ИСО 5807–85) Схемы алгоритмов, программ, данных и систем, обозначения условные и правила выполнения 243
Приложение Б ГОСТ 34.601–90 Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания 265
Приложение В ГОСТ 19.201–78 Техническое задание. Требования к содержанию и оформлению 271
Приложение Г ГОСТ 7.32–2001 Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления 274
Приложение Д ГОСТ Р (ИСО/МЭК 12119–2000) Информационная технология «Пакеты программ». Требования к качеству и тестирование 290
Приложение Е (справочное) Определения из других нормативных документов 307

ВВЕДЕНИЕ

Ни одна сфера духовной культуры не оказала столь существенного и динамичного влияния на общество, как наука. Наука разрушила многие наши представления, идущие из далекого прошлого, существенно изменила состояние нашего сознания.

Понятие «наука» имеет несколько определений:

– сфера человеческой деятельности, функция которой – выработка и теоретическая систематизация объективных знаний о действительности;

– одна из форм общественного сознания;

– деятельность по получению нового знания для определения научной картины мира;

– особый тип знаний об объекте, который в конечном счете дан эмпирически.

Особенность научных знаний человека заключается в глубоком проникновении в суть явлений и в их теоретическом описании. Буквальное значение слова «наука» – знание (Краткий словарь по философии). Однако не любое знание может быть научным. Научное знание начинается только тогда, когда за совокупностью фактов осознается закономерность – всеобщая и необходимая связь между ними, что позволяет объяснить, почему данное явление протекает так, а не иначе, предсказать дальнейшее его развитие.

Исследования и наука объединены в одно целое, неразрывное состояние. При исследовательской деятельности определяющим является подход, а не число источников информации (книги, статьи, интернет-страницы, программы, ГОСТы и др.), на основании которых выполнена работа. На основании информационных источников можно выполнить реферативную, исследовательскую и практическую работу. Самое главное, что эти три составляющие являются неразрывными и определяющими само исследование, как целое.

Реферат ни в коем случае не должен отражать субъективных взглядов студента, его готовившего, а также давать оценку изученной информации. Суть реферативной работы – в поиске первоисточников, наиболее полно освещающих выбранную исследовательскую проблему. Специфика реферата – в том, что в нем нет развернутых доказательств, рассуждений, он отвечает на вопрос о том, что нового, существенного содержится в направлении выбранных исследований, приводится сравнительный анализ различных подходов к решению проблемы, определение цели и задач исследований. Реферат может рассматриваться как один из этапов исследовательской работы, но не заменитель ее.

Суть исследовательской работы состоит в сопоставлении данных первоисточников, их творческом анализе и проводимых на его основании новых выводов, полученных в ходе изучения определенной заранее предметной области. Под исследовательской деятельностью в целом понимается такая форма организации работы, которая связана с решением учащимся исследовательской задачи с неизвестным заранее решением. Учебно-исследовательская деятельность – это деятельность, главной целью которой является образовательный результат, она направлена на обучение учащихся, развитие у них исследовательского типа мышления и практических навыков. Главное здесь не овладение новыми, ранее совсем неизвестными фактами, а понимание и изучение алгоритма ведения исследования, определение навыков, которые могут быть затем использованы в исследовании любой сложной задачи.

В настоящем пособии сделана попытка рассмотреть весь процесс подготовки учебной исследовательской работы, начиная с выбора темы исследования до подготовки материалов, ее представляющих.

В результате изучения курса будут получены компетенции:

1. Аналитические:

– умение ставить исследовательские задачи, определять объект и предмет исследования, выделять и уточнять исследовательскую проблему;

– формулировать цели и задачи предполагаемого проекта, планировать работу с ним;

– умение выбирать понятия для описания и решения исследовательской проблемы и работать с ними;

– освоение алгоритма разработки программы исследования;

– навыки написания и редактирования научных текстов.

2. Системные:

– способность к самостоятельной работе, сбору необходимой информации (в том числе – поиску источников информации);

– способность к анализу и критической оценке существующей литературы;

– способность к выработке оптимальных исследовательских стратегий с учетом имеющихся ограничений;

– базовые навыки написания научных текстов и подготовки устных презентаций, в том числе – умение правильно оформить научно-справочный аппарат.

3. Коммуникационные:

– развитие навыков профессиональной коммуникации в контексте подготовки реферата, доклада, презентации,

– способность представить программу собственного исследовательского проекта в устной и письменной формах.

Этапы учебно-исследовательской деятельности учащихся предполагают получение ими определенных знаний и умений:

1. В ходе исследования организуется поиск в какой-то области, формулируются отдельные характеристики итогов работ. В этом случае и отрицательный результат есть тоже результат. Логика построения исследовательской деятельности включает формулировку проблемы исследования, выдвижение гипотезы (для решения этой проблемы) и последующую экспериментальную или модельную проверку выдвинутых предположений.

2. Умение видеть проблему приравнивается к проблемной ситуации и понимается как возникновение трудностей в решении проблемы при отсутствии необходимых знаний и средств; умение ставить вопросы можно рассматривать как вариант, компонент умения видеть проблему; выдвижение гипотезы – это формулирование возможного варианта решения проблемы, который проверяется в ходе проведения исследования; структурирование текстов является частью умения работать с текстом, которое включает достаточно большой набор операций; умение давать определение понятиям – это логическая операция, которая направлена на раскрытие сущности понятия либо установление значения термина.

3. Умение выделять из материала сущностное, которое будет использовано в исследовании; определять параметры (показателей количественных и качественных); формулировать вопросы, предлагаемые для решения, обсуждения и пр.; наблюдать; проводить эксперименты; делать выводы и умозаключения; организовать наблюдения, планирование и проведение простейших опытов для нахождения необходимой информации и проверки гипотез; использовать разные источники информации; обсуждать и оценивать полученные результаты и применять их к новым ситуациям; умение делать выводы и заключения; умение классифицировать; структурировать материал; объяснять и доказывать, защищать результаты проделанной работы, принять стороннюю оценку полученных результатов.

В книге [1] написано: «Программист, подобно поэту, работает почти непосредственно с чистой мыслью. Он строит свои замки в воздухе и из воздуха, творя силой воображения. Трудно найти другой материал, используемый в творчестве, который столь же гибок, прост для шлифовки или переработки и доступен для воплощения грандиозных замыслов». Некоторые психологи, которые работают с программистами, идут дальше и даже утверждают, что программирование – это высшая форма творчества [2].

Творчество в программировании начинается с определения целей программы и заканчивается только тогда, когда в ее коде, написанном на каком-либо языке программирования, поставлена последняя точка с запятой. Попытки разделять программистов на творческую элиту, архитекторов и проектировщиков, и нетворческих программистов-кодеров не имеют под собой объективных оснований. Даже если алгоритм программы строго определен математически, два разных программиста его закодируют по-разному, и полученная программа будет иметь разные потребительские качества.

Разработка программных средств имеет ряд специфических особенностей:

– разработка программы содержит определенные этапы формализации на всех этапах жизни программы (использование ГОСТов и др. регламентных документов);

– разработка программного обеспечения это творческая, исследовательская работа, которая, с одной стороны, сугубо индивидуальна, а с другой стороны, четко регламентирована;

– абстракции сопутствуют программисту на всех уровнях разработки программы от описания ее целей до исполняемого машинного кода, в зависимости от сложности программы этих уровней могут быть единицы, десятки или сотни и на каждом уровне абстракций их деталей становится все больше и больше;

– обязательным шагом в каждом процессе (этапе) разработки ПС должна быть проверка правильности принятых решений;

– разработка программы заканчивается получением продукта, который передается пользователям.

На большинство из указанных выше особенностей разработки программного обеспечения указывается в данном учебном пособии. Задачей курса является пройти основные этапы разработки программного обеспечения, от зарождения идеи до написания текста программы и документации.

1 ПОДГОТОВКА К ПРОВЕДЕНИЮ НАУЧНОГО ИССЛЕДОВАНИЯ

Подготовка к проведению научного исследования традиционно предполагает наличие нескольких этапов.

1.1 Объектная область, объект и предмет

Исследование, в отличие от повседневного опытного познания, носит систематический и целенаправленный характер. Поэтому важной задачей является четкое определение сферы научно-исследовательской деятельности – ее объекта и предмета, своеобразной «системы координат» исследования. Работа над любым исследованием начинается с определения названной «системы». Ее составляют три элемента: «объектная область», «объект» и «предмет» исследования. Этот этап предшествует выбору темы исследования.

Дадим краткие определения каждого из элементов «системы»:

- Объектная область исследования – это сфера науки и практики, в которой находится объект исследования.
- Объект исследования – это определенный процесс или явление, порождающее проблемную ситуацию. Объект – это своеобразный носитель проблемы – то, на что направлена исследовательская деятельность. С понятием объекта тесно связано понятие предмета исследования.
- Предмет исследования – это конкретная часть объекта, внутри которой ведется поиск. Предметом исследования могут быть явления в целом, отдельные их стороны, аспекты и отношения между отдельными сторонами и целым (совокупность элементов, связей, отношений в конкретной области объекта). Именно предмет исследования определяет тему работы.

Границы между объектной областью, объектом, предметом условны, подвижны. То, что в одном случае является объектом исследования, в другом – может стать объектной областью; то, что было в данном случае объектом, в ином случае предстает в качестве предмета исследования.

1.2 Тема, проблема и актуальность исследования

Тема – еще более узкая сфера исследования в рамках изучаемого предмета. Выбор темы для многих является весьма трудным этапом. Часто учащиеся выбирают слишком масштабные, сложные темы или, наоборот, очень простые. Сложные темы могут оказаться непосильными для их раскрытия в рамках учебного исследования. Возможен и такой случай, когда учащийся в силу тех или иных причин выбирает тему, давно ставшую решенной проблемой. Тема – ракурс, в котором рассматривается проблема. Она представляет объект изучения в определенном аспекте, характерном для данной работы.

Чтобы облегчить процесс выбора темы, попытаемся выделить основные критерии:

– желательно, чтобы тема представляла интерес для учащегося не только на данный, текущий момент, но и вписывалась в общую перспективу его профессионального развития, т. е. имела непосредственное отношение к предварительно выбранной им будущей специальности;

– тема также должна быть реализуема в имеющихся условиях, в определенные учебным планом сроки. Это значит, что по выбранной теме должны быть доступны оборудование и литература. Примером реализуемой темы может служить тема «Особенности компиляторов C++». Заявленная тема не требует много времени, специального оборудования и сложных условий.

Важно с самого начала правильно сформулировать тему. Ведь тема – это своего рода визитная карточка исследования. Такая формулировка будет носить не окончательный, а предварительный характер. Здесь также целесообразно вспомнить о некоторых традиционных требованиях: тема должна быть сформулирована по возможности лаконично, а используемые при ее

формулировке понятия должны быть логически взаимосвязаны. Формулировка темы отражает сосуществование в науке уже известного и еще не исследованного, т. е. процесс развития научного познания. Вследствие этой причины очень ответственным этапом в подготовке исследования становится этап обоснования актуальности темы.

Обосновать актуальность – значит, объяснить необходимость данной темы в контексте общего процесса научного познания. Определение актуальности исследования – обязательное требование к любой работе. Актуальность может состоять в необходимости получения новых данных и необходимости проверки новых методов и т. п. Освещение актуальности, как и формулировка темы, не должно быть многословным. Не нужно начинать ее описание издалека. Одной страницы, чтобы показать главное, достаточно. Обосновывая актуальность выбранной темы, следует указать, почему именно она и именно на данный момент является актуальной. Здесь желательно кратко осветить причины, по которым изучение этой темы стало необходимым, и что мешало ее раскрытию раньше, в предыдущих исследованиях.

Несомненным показателем актуальности является наличие проблемы данной области исследования. Когда и почему возникает проблема? Как правило, ее появление связано с тем, что существующее научное знание уже не позволяет решать новые задачи, познавать новые явления, объяснять ранее неизвестные факты или выявлять несовершенство прежних способов объяснения фактов и эмпирических закономерностей. Таким образом, можно представить проблему как некую противоречивую ситуацию, требующую своего разрешения. Разрешение этого противоречия самым непосредственным образом связано с практической необходимостью. Это значит, что, обращаясь к той или иной проблеме, исследователю нужно четко представить, на какие вопросы практики могут дать ответ результаты его работы.

Правильная постановка и ясная формулировка новых проблем в исследовании очень важна. Она определяет стратегию исследования направление научного поиска. На данном этапе работы не всегда можно точно определить тему исследования, пути и способы ее разработки и осуществления. Для этого необходимо изучить научную литературу по вопросу. После чего тема обычно уточняется, изменяется.

1.3 Изучение научной литературы и уточнение темы

Поначалу может сложиться впечатление, что литература по теме – это некое безграничное пространство, в котором невозможно обнаружить какие-либо ориентиры. Это затруднение будет достаточно легко преодолено, если вы выберете верный метод ознакомления с источниками. Освоенный алгоритм работы позволит вам в дальнейшем свободно ориентироваться в литературе по избранной вами теме. Начинать целесообразно с самостоятельной работы по составлению библиографического списка источников по теме.

В составлении списка необходимой для изучения литературы обязательно приобрести навык самостоятельной работы в библиотеке: знакомства с системой библиотечных, в том числе электронных, каталогов, с приемами правильного оформления библиографических данных, поиска в Интернете релевантной информации.

Сначала можно привести ряд общих рекомендаций. Изучение научных публикаций желательно проводить по этапам. Лучше начинать с работ так называемого общего характера, т. е. таких работ, из которых можно получить представление об основных вопросах, к которым примыкает избранная тема, а затем уже вести поиск узкоспециального материала. Данные каждого издания следует заносить в файл, чтобы не потерять расположение и смысл информации. В первую очередь следует ознакомиться с традиционными вузовскими учебниками по соответствующей теме работы дисциплине,

где собрана и обобщена базовая информация по вопросу. В конце глав учебников обычно публикуются ориентировочные списки литературы, что может помочь в поиске и составлении собственного списка по вашей конкретной теме. Кроме этого, в учебниках указываются основные монографии по вопросу, знакомство с которыми станет следующим этапом изучения литературы.

Работая с литературой по теме, учащийся должен владеть различными типами чтения, предполагающими различную степень глубины проникновения в материал.

Просмотровое чтение желательно использовать в тех случаях, когда требуется познакомиться с общим содержанием книги, ее глав или параграфов, автором произведения. При этом обычно читается титульный лист, оглавление, аннотация, отдельные абзацы и предложения.

Совсем не обязательно тщательно изучать весь предварительный список литературы, среди которой будут монографии, статьи, тезисы, сборники, научные журналы. Как правило, это значительно замедляет процесс освоения текстов и тормозит исследование на его начальном этапе. Поэтому важно научиться свободно оперировать различными приемами работы с текстом. Здесь желательно ориентироваться в структуре издания. Вот лишь ее некоторые характерные элементы с соответствующими им функциями:

1) заголовок в научной литературе указывает на тему;

2) аннотация расположена на обороте титульного листа и представляет содержание работы;

3) оглавление содержит план изложения темы, является своего рода путеводителем по книге. Оно знакомит с проблематикой работы, её общей структурой и дает возможность быстрого поиска информации;

4) предисловие излагает задачи, поставленные автором; более подробно характеризует структуру издания и ориентирует в ней читателя. Оно

предваряет изложение основного материала и дает установку на его восприятие;

5) послесловие подводит итог, сообщает краткие выводы исследования;

6) справочный материал дает комментарий к понятиям, терминам, фактам, которые нуждаются в пояснении. Уже этой информации будет достаточно, чтобы решить, насколько необходим для конкретной работы тот или иной текст.

Ознакомительное (выборочное) чтение поможет в поиске ответа на определенные вопросы по нескольким источникам и для сравнения и сопоставления найденной информации, выработки своей собственной точки зрения.

Изучающее чтение – это активный вид подробного чтения. Он предполагает, что вы читаете внимательно, останавливаясь и обдумывая информацию. Данный вид чтения требует последовательного изучения материала по параграфам, главам, частям. Заключительный этап содержит в себе и обучающий момент: здесь формируется умение критически воспринимать информацию. Цель изучающего чтения – получение необходимой информации; понимание логики доказательств; поиск ответов на поставленные перед вами вопросы.

Какие приемы можно рекомендовать для фиксирования нужной вам информации? Целесообразно посоветовать создавать своеобразный «банк данных» по теме своей работы. Важно делать выписки всего, что может вам пригодиться в вашей научной работе: интересные мысли, факты, цифры, различные точки зрения. Это можно делать либо в форме отдельного файла, либо отдельной директории с набором файлов. Здесь можно использовать самые различные методы работы. Их существует множество. Следует лишь верно выбрать именно тот, который лучше всего соответствует вашим индивидуальным особенностям, темпу мышления, объему памяти, широте

ассоциативных связей. Может случиться и так, что не вся выписанная информация окажется необходимой. При этом ни в коем случае не нужно пытаться включить в исследование весь имеющийся материал, как бы звучны ни были иные имена и цитаты, – это может лишь повредить целостности и логичности исследования. Если доказательство того или иного положения строится преимущественно на цитатах, то это, как правило, производит неблагоприятное впечатление. Исключением является изложение концепции произведения в полемических целях.

При изучении литературы не следует стремиться к однозначному заимствованию материала. Будет правильнее сопоставить, проанализировать найденную информацию, пропустить «через себя». Ведь основой для получения нового знания должны служить не чьи-то, а собственные мысли, пусть и возникшие в ходе знакомства с чужими работами как отклик на них.

1.4 Элементы исследований

Определение гипотезы. Уточнив тему в результате изучения специальной литературы, исследователь может приступать к выработке гипотезы. Это один из самых ответственных моментов работы над исследованием. Сначала обратимся к определению самого понятия. Гипотеза должна удовлетворять ряду требований:

- быть проверяемой;
- содержать предположение;
- быть логически непротиворечивой;
- соответствовать фактам.

В переводе с древнегреческого языка гипотеза – «основание, предположение». В современной научной практике гипотеза определяется как научно-обоснованное предположение о непосредственно наблюдаемом явлении. При формулировке гипотезы обычно используются словесные конструкции типа: «если..., то...»; «так..., как...»; «при условии, что...»,

т. е. такие, которые направляют внимание исследователя на раскрытие сущности явления, установление причинно-следственных связей. Процесс формулирования гипотезы не является одномоментным актом. Вначале лучше составить ее рабочий вариант – как первичное, временное предположение, служащее систематизации материала. После накопления значительного количества фактического материала рабочий вариант гипотезы уточняется, видоизменяется и приобретает вид окончательной научной гипотезы.

Вслед за выработкой гипотезы начинается следующий этап подготовки к исследованию – определение его цели и задач. Точнее, не начинает, а продолжается, так как выработка цели и задач происходит уже в ходе разработки гипотезы. Заметим, что любое деление на этапы достаточно условно, особенно в практической деятельности, какой является и деятельность научно-исследовательская. Тем не менее это деление необходимо в чисто учебных, объяснительных целях для того, чтобы максимально ясно обозначить все составляющие той или иной деятельности. На практике же названные этапы могут протекать параллельно, перекрещиваться и даже меняться местами в зависимости от конкретной ситуации исследования. Важно лишь все их учитывать, как необходимые элементы данного вида деятельности. Именно этим оправдывается предпринятое нами структурирование. Но вернемся к определению понятий целей и задач в контексте подготовки к исследованию.

Цель и задачи исследования. В общем виде цель и задачи должны уточнить направления, по которым пойдет доказательство гипотезы. Цель исследования – это конечный результат, которого хотел бы достичь исследователь при завершении своей работы. Выделим наиболее типичные цели. Ими может быть определение характеристик явлений, не изученных ранее; выявление взаимосвязей неких явлений; изучение развития явлений; описание нового явления; обобщение, выявление общих закономерностей; создание классификаций.

Формулировку цели исследования также можно представить различными способами – традиционно употребляемыми в научной речи клише. Приведем примеры некоторых из них. Можно поставить целью:

– выявить;
– установить;
– обосновать;
– уточнить;
– разработать.

Формулировать задачи необходимо очень тщательно, так как описание их решения в дальнейшем составит содержание глав. Заголовки глав рождаются именно из формулировок задач. Предложим одно из определений понятия «задача». Задача исследования – это выбор путей и средств для достижения цели в соответствии с выдвинутой гипотезой. Задачи лучше всего формулировать в виде утверждения того, что необходимо сделать, чтобы цель была достигнута. Постановка задач основывается на дроблении цели исследования на подцели. Перечисление задач строится по принципу от наиболее простых к наиболее сложным, трудоемким, а их количество определяется глубиной исследования.

Цель – идеальное видение результата, который направляет деятельность человека. Исследователь для достижения поставленной цели и проверки положений сформулированной им гипотезы выделяет конкретные задачи исследования.

После формулирования гипотезы, целей и задач исследования следует этап определения методов.

1.5 Определение методов исследования

Как показывает учебная практика, на первых порах овладения навыками научной работы учащимся, прежде всего, недостает как опыта для ее организации, так и опыта использования различных методов научного

познания и применения традиционных для научной практики логических законов и правил.

Что заключает в себе понятие «метод»? Метод – это способ достижения цели исследования. Уже отсюда очевидна решающая роль метода в успехе той или иной исследовательской работы. Ясно, что от выбора метода зависит сама возможность реализации исследования – его проведения и получения определенного результата.

Методы научного познания традиционно делятся на общие и специальные. Применения специальных методов решения требует большинство специальных проблем конкретных наук. Они определяются характером исследуемого объекта, никогда не бывают произвольными. Как правило, их применение требует от исследователя уже значительной подготовленности.

Теоретические методы

Моделирование позволяет применять экспериментальный метод к объектам, непосредственное действие с которыми затруднительно или невозможно. Оно предполагает мыслительные или практические действия с «заместителем» этого объекта – моделью.

Абстрагирование состоит в мысленном отвлечении от всего несущественного и фиксировании одной или нескольких интересующих исследователя сторон предметов. Следует различать процесс абстрагирования и его результат – абстракцию. Процесс абстрагирования – это совокупность действий, ведущих к получению такого результата (абстракции).

Анализ и синтез. Анализ – это метод исследования путем разложения предмета на составные части. Синтез, напротив, представляет соединение полученных при анализе частей в нечто целое. Нужно помнить, что методы анализа и синтеза ни в коем случае не изолированы друг от друга, а сосуществуют, друг друга дополняя. Методами анализа и синтеза проводится,

в частности, начальный этап исследования – изучение специальной литературы по теории вопроса.

Восхождение от абстрактного к конкретному предполагает два условно самостоятельных этапа. На первом этапе единый объект расчленяется, описывается при помощи множества понятий и суждений, на втором этапе восстанавливается исходная целостность объекта, воспроизводится во всей многогранности – но уже в мышлении.

Помимо специальных методов, характерных для определённых областей научного знания, существуют общие методы научного познания. В отличие от специальных они используются в самых различных по предмету науках – от литературы до химии, математики. К ним относятся: теоретические методы, эмпирические методы, математические методы.

Эмпирические методы

Наблюдение представляет собой активный познавательный процесс, который опирается на работу органов чувств человека и его предметную деятельность. Это наиболее элементарный метод познания. Наблюдения должны приводить к результатам, которые не зависят от воли, чувств, желаний человека. Это предполагает изначальную объективность: наблюдения должны информировать нас о свойствах и отношениях реально существующих предметов и явлений.

Сравнение – один из наиболее распространенных методов познания. Недаром говорится, что все познается в сравнении. Сравнение позволяет установить сходство и различие предметов и явлений. Выявление общего, повторяющегося в явлениях – это серьезный шаг к познанию закономерностей и законов окружающего нас мира.

Эксперимент предполагает вмешательство в естественные условия существования предметов и явлений или воспроизведение их определенных

сторон в специально созданных условиях с целью их изучения. Экспериментальное изучение объектов по сравнению с наблюдением имеет ряд преимуществ:

– в процессе эксперимента возможно изучение явления в «чистом виде», т. е. объективно;

– эксперимент позволяет исследовать свойства объектов в экстремальных условиях;

– достоинством эксперимента является его повторяемость, т. е. возможность проверки и перепроверки полученной информации.

Измерение – представляет собой процедуру определения численного значения величины посредством проведения эксперимента. Ценность этого метода заключается в том, что он дает точные, количественно определенные сведения об окружающем мире.

Математические методы

– Статистические методы;

– методы и модели теории графов и сетевого моделирования;

– методы и модели динамического программирования;

– методы и модели массового обслуживания;

– метод визуализации данных (функции, графики и т. п.).

Методы исследования можно подразделить на методы теоретического и эмпирического исследования, фундаментальные и прикладные, количественные и качественные и т. д. Каждый метод имеет три основных аспекта:

– объективно-содержательный,

– операциональный,

– праксиологический.

Первый аспект выражает обусловленность метода предметом исследования через теорию. Операционный аспект фиксирует зависимость содержания метода не столько от объекта, сколько от субъекта, его компетентности,

способности перевести соответствующую теорию в систему правил, принципов, приемов, которые в своей совокупности и образуют метод. Праксиологический аспект метода составляют такие его свойства, как эффективность, надежность, ясность, конструктивность и т. д.

Для овладения основными методами, которые будут применены в исследовании, необходимо пройти подготовку, например, выполняя специальные упражнения. Данные упражнения желательно вписать в схему общей подготовки к проведению исследования. Подготовка к проведению может проходить как в форме спецкурса, так и в форме выполнения индивидуальных заданий. Этот этап предшествует собственно практической работе и является его необходимой предпосылкой.

2 ПРОВЕДЕНИЕ НАУЧНОГО ИССЛЕДОВАНИЯ

2.1 Этапы научного исследования

Проведение исследования включает в себя два последовательных этапа: собственно, проведение (так называемый технологический этап) и аналитический, рефлективный этап.

Чтобы четко уяснить себе последовательность проведения исследования, желательно составить рабочий план. В рабочей программе исследования обязательно рассматриваются действия по подготовке и проведению экспериментов. С учетом специфики творческого процесса такой план должен предусматривать все, что можно предвидеть уже в самом начале исследовательской работы. Конечно, в науке возможны и случайные открытия, но нельзя строить научное исследование, ориентируясь на случайности. Только плановое исследование может позволить надежно шаг за шагом познать новые факты и закономерности.

В рабочем плане необходимо указать цель планируемых экспериментов; перечислить необходимый для их проведения инвентарь; формы записей в черновых тетрадях. В рабочий план также включается первичная обработка и анализ результатов практических действий, этапы их проверки. По существу, в рабочий план включаются все элементы, обозначенные в части подготовки проведения исследования.

В первом блоке они представляют собой содержание теоретической работы, где формируются понятийный аппарат, основы исследовательской деятельности. Включенные в рабочий план те же элементы обозначают этапы непосредственно практики проведения исследования – от определения его объекта и предмета до выбора метода. Перечень этих действий составляет первый блок рабочего плана.

Во втором блоке описывается собственно экспериментальная часть работы. Содержание экспериментальной части зависит от объектной области исследования, темы работы, в соответствии с ним и определяется специфика.

Вслед за проведением эксперимента, технологического этапа работы необходимо отрефлексировать полученные результаты: проанализировать, насколько они позволяют подтвердить выдвинутую в начале исследования гипотезу, уточнить их соответствие поставленным целям. Только после проведения рефлексивной части можно приступать к планированию следующего блока работы, включающего в себя оформление результатов исследования.

Третий блок включает в себя оформление результатов исследования.

На следующем этапе прописывается способ экспертизы и представления результатов исследования – от рецензии до обсуждения в группе учащихся выступления на конференции при очном обучении и контроле руководителя при обучении с применением дистанционных образовательных технологий. Заметим, что чем чаще результаты работы подвергаются обсуждению в разных по составу аудиториях, тем лучше для ее автора. Особенно продуктивны обсуждения в группах, где несколько учащихся или студентов работали над исследованиями близкой тематики. Здесь обнаруживаются наибольшие возможности для продуктивной дискуссии.

На заключительном этапе целесообразно продумать способ представления результатов своего исследования на конференцию, отработать формы представления в виде статьи и тезисов, осмыслить возможные рекомендации по практическому применению результатов, т. е. спланировать внедренческий этап исследования.

На более поздних стадиях работы составляют план-проспект, т. е. такой план, который представляет собой реферативное, более подробное изложение вопросов, по которым в дальнейшем будет систематизироваться весь собранный фактический материал.

План-проспект служит основой для последующей оценки преподавателем соответствия работы учащегося целям и задачам проводимого исследования. По этому плану уже можно будет судить об основных положениях содержания будущей исследовательской работы, принципах раскрытия темы, о построении и соотношении объемов отдельных ее частей.

Практически план-проспект – это уже черновое оглавление работы с реферативным раскрытием содержания ее глав и параграфов. Удобство составления предварительного плана-проспекта очевидно: путем систематического включения в такой план все новых и новых данных его можно довести до окончательного объема работы.

Кроме этого, наличие плана-проспекта, представляющего всю работу целиком, позволит предварительно провести анализ ее результатов, проверить их соответствие намеченной цели и при необходимости внести коррективы в ту или иную часть работы. Отметим, что исследовательская деятельность представляет собой достаточно длительный процесс, который включает и этап предварительного обучения учащихся (при очной форме обучения), и практику проведения, и анализ, и оформление результатов, и их представление на конференции. Поэтому вопрос о распределении времени по подготовке и проведению исследования – один из самых важных. Если планируется представление результатов работы на конференцию, то сроки ее проведения будут определять последовательность и примерные сроки всех предшествующих этапов. Обычно подготовка и проведение исследовательской работы занимают от одного до двух семестров.

Исследовательская работа представляется на экспертизу для выяснения ее актуальности, новизны, полноты исследований, соответствия названия выполненной работе и др. Автор работы может (желательно выполнить эту работу) опубликовать результаты своих исследований на конференции. Конечно, выступление на конференции лишь условно можно назвать заключительным этапом исследовательской работы. На самом деле, это некая заметная веха, позволяющая продолжить исследование уже на более высоком уровне – с обновленными в обсуждении результатами, обогащенными замечаниями коллег-исследователей и специалистов, со знанием о результатах других работ близкой тематики.

2.2 Информационная проработка темы

Любой этап научного исследования начинается с информационной проработки темы, которая включает в себя:

– сбор материала по теме исследования;

– работу с иностранными источниками, их перевод;

– формирование литературного обзора по теме исследования.

Как результат этой деятельности – обзор литературы по теме исследования. Но прежде выясним, какими бывают источники информации. Основные источники информации можно условно подразделить на первичные и вторичные. Перечень основных источников информации представлен на рис. 2.1.



Рис. 2.1 – Основные источники информации

Первичные источники информации

Опубликованные источники информации

Книгой называется непериодическое издание (произведение печати) в виде нескольких сброшюрованных листов печатного материала, объемом

более 50 страниц, как правило, в обложке или переплете, прошедшее редакционно-издательскую обработку.

Брошюра – произведение печати, объемом от 5 до 50 страниц.

Периодическое издание – произведение печати, выпускаемое отдельными, неповторяющимися по содержанию выпусками, под одним названием, которое регулярно выходят через определенные или неопределенные промежутки времени, причем каждый выпуск имеет порядковый номер или дату.

Журнал – периодическое издание, выходящее не реже двух раз в год и не чаще одного раза в неделю, подчиненное интересам определенного круга читателей, имеющее постоянное название, одинаковое оформление и ежегодную сквозную нумерацию.

Первый в мире журнал Journal des Scavants вышел во Франции в январе 1665 г. Спустя три месяца в Лондоне увидел свет первый номер второго в мире журнала Philosophical transactions of the Royal Societes, а через три года в 1667 г. в Голландии появился третий журнал Collegium privitum amsclodamense. Всего с 1665 по 1730 г. появилось 30 новых журналов. Первый в России журнал – приложение к газете «Санкт-Петербургские ведомости» – «Месячные исторические, генеалогические и географические примечания в Ведомостях» издан в 1728 г. В течение первых 100 лет своего существования журнал служил источником информации о новых научных книгах. Позднее в журналах стали появляться отдельные оригинальные статьи в виде писем ученых друг к другу, а в XIX в. журнал стал активным пропагандистом научных и технических достижений.

Важная роль журнала в научных коммуникациях состоит в следующем:

– журнал выполняет одновременно функции текущего оповещения и публичного архива, обеспечивая оперативное доведение информации;

– является средством апробации результатов научных исследований;

– фиксирует приоритет научных и прикладных задач, создавая условия для признания авторов как ученых.

Журнальные публикации составляют примерно 70% всех научных документов. Примерно 80% специалистов различных уровней считают научный журнал основным источником научно-технической информации.

Препринт-оттиск опубликованной статьи, которое издательство рассылает по своей корреспондентской сети. Кроме того, многие научные центры, вузы получили право печатать препринты в виде брошюр.

Преимущества препринта:

– публикуется то, что, возможно, не могло быть опубликовано в журнале и никогда потом не будет опубликовано;

– публикуется материал, считающийся сырым, спорным, не апробированным, не требующий такого рецензирования, как журнальная статья;

– нет ограничений по объему;

– рассылка по списку с участием автора в руки специалистов этой области.

Неопубликованные источники информации

Диссертация – квалификационная научная работа в определенной области наук, содержащая совокупность научных результатов и положений, выдвигаемых автором для публичной защиты и свидетельствующая о личном вкладе автора в науку и о его качествах как ученого. Основу диссертации составляют выполненные и опубликованные научные работы, открытия или крупные изобретения, внедренные в производство технологические процессы и др.

Депонированные рукописи – переданные на хранение в орган-депозитарий научные работы, выполненные индивидуально или в соавторстве и рассчитанные на ограниченный круг потребителей. Депонирование сокращает

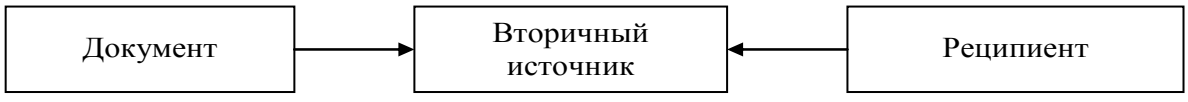
срок публикации, обеспечивает доступ исследователей к непубликуемым материалам. Депонирование осуществляется централизованно. Сведения о депонированных рукописях отражаются в реферативных журналах (РЖ) и библиографических указателях. Все заинтересованные потребители могут заказать копию любой депонированной рукописи. Депонирование было введено с 1971 г. с целью ознакомления ученых и специалистов с рукописями статей и обзоров узкоспециального характера, которые нецелесообразно публиковать обычными способами печати. Рукописи разных научных направлений депонируются разными органами-депозитариями: по естественным, точным наукам и технике – в ВИНИТИ, по общественным наукам – в ИНИОН и т. д.

Депонированные рукописи приравниваются к публикациям. Авторы сохраняют право на публикацию статей в научных изданиях. Ограничения по объему депонированных рукописей значительно меньше, чем для публикаций, что позволяет автору более полно представить результаты своей работы.

Научный отчет – отчет научной организации о проведенном исследовании, доступен организациям и частным лицам, хранится в фонде ВНТИЦентра.

Вторичные источники информации

Вторичные источники служат посредником между документами и реципиентами (читателями, зрителями, слушателями).



Вторичные источники являются конечным результатом аналитико-синтетической переработки информации (АСПИ). Различаются следующие виды АСПИ: библиографическое описание произведений печати и других

документов, аннотирование, реферирование, научно-технический перевод, систематизация и обобщение научно-технических данных, составление обзоров научно-технической литературы. Результатом каждого вида АСПИ является определенный вид вторичного документа: библиографическое описание, аннотация, перевод, реферат и т. д.

Резюме – отвечает на вопрос «Что?» об источнике информации.

Аннотация – отвечает на вопрос «Что?» и «О чем?» об источнике информации.

Реферат – получается при объединении аннотации и резюме. Состоит из трех частей: аннотационной, фактографической и резюмирующей. Реферат – это сокращенное изложение содержания первичного документа (или его части) с основными фактическими сведениями и выводами.

К идее реферата пришли в прошлом веке двумя путями: первый путь открыли библиотекари, которые долгое время гордились тем, что знали все книги «в лицо». Но затем книг стало очень много, и были придуманы каталоги. Книги расписывались на библиографические карточки и публиковалась библиотечная библиография, дополненная краткой аннотацией, которая была необходима, когда фамилия автора и название книги ни о чем не говорили. Такая библиография стала носить периодический характер, называлась аннотированной и превратилась в реферативный журнал (РЖ).

Второй путь проложили журналисты. На страницах журналов регулярно помещались пересказы интересных книг, которые из подробных конспектов превратились в краткие конспекты, подробные и краткие рефераты. Эти рефераты помещали в специальные разделы и наиболее важные переходили в реферативные сборники: разовые и периодические. Так идея реферата переросла в идею реферативного журнала.

РЖ является важным, хотя и не единственным, элементом информационной культуры. Для плодотворной работы с источниками информации Вам необходимо найти свой РЖ. Не страшно, если бесполезной информации

окажется 50–70% (нужным оказывается каждый второй, третий реферат). Ненужное легко опускается, а нужное все искупает.

В вузовском понимании реферат –это реферативный обзор источников информации по данной теме или информационный анализ источников. Реферирование подразумевает анализ литературы по проблеме, т. е. систематизированное изложение чужих мыслей с указанием на первоисточник и в обязательном порядке с собственной оценкой изложенного. Если у студента не хватает подготовки на то, чтобы выразить согласие или несогласие с той или иной точкой зрения, то он должен констатировать совпадение или расхождение мнений по тому или иному вопросу разных авторов примерно следующим образом: «Как видно из изложенного, мнения авторов по проблеме существенно расходятся в том-то и том-то».

2.3 Виды и методика проведения информационного поиска

Различают следующие виды информационного поиска:

- по ключевым словам,
- по тематическим рубрикам,
- по фамилии автора (или авторов),
- нумерационный поиск (по номеру документа),
- ретроспективный,
- текущий поиск.

Поиск, по ключевым словам, осуществляют при наличии во вторичных изданиях предметного указателя. При этом находят нужное ключевое слово и номера рефератов источников в данном сборнике, в которых встречается это слово или которые отвечают тематике поиска.

Поиск по тематическим рубрикам осуществляется путем просмотра всей рубрики или раздела, в которых исследователь надеется найти отвечающие его запросу источники.

В случае если исследователю известны фамилии авторов, которые работают в интересующей области, возможно осуществление поиска по фамилии автора. В этом случае пользуются авторским указателем во вторичном издании, найдя нужную фамилию в указателе и номера тех рефератов в данном сборнике, которые относятся к данной фамилии. По этим номерам находят рефераты, которые, возможно, отвечают запросу.

Если известен номер документа (в случае поиска диссертаций, научных отчетов, депонированных рукописей и др.), можно осуществить нумерационный поиск. Для этого в нумерационном указателе вторичного издания ищете нужный номер. Рядом с ним находят номер страницы, на которой расположен реферат нужного документа в данном сборнике.

В некоторых случаях в исследовательских целях необходимо провести ретроспективный поиск с глубиной 5, 10, 20 лет или больше, т. е. просмотреть всю информацию за последние 5 лет или больше. В этом случае просматриваются или конкретные источники, или указатели вторичных изданий за указанный период с целью поиска информации по интересующей теме. Текущий поиск проводят по тем изданиям, которые появляются в текущем году.

3 ОФОРМЛЕНИЕ НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЫ

3.1 Оформление результатов исследовательской работы

Принято считать, что оформление – незначительный, чисто формальный этап создания рукописи научного исследования. На самом деле, это не так. Оформление результатов исследования – один из самых трудоемких этапов работы. Существует несколько основных форм представления результатов научной работы:

– текст научного сочинения;

– статья, тезисы;

– доклад, сообщение;

– отчет и т. д.

Определению формы научного произведения сопутствует составление плана исследования. Именно тогда выявляются контуры будущей работы, намечается характер и объем иллюстративного материала, складывается круг источников. Это говорит о том, что поиск оптимальной формы научной работы идет на каждой стадии исследования.

Каждой из форм присущи свои особенности написания. Тезисы всегда выявляют суть содержания исследования и позволяют обобщить имеющийся материал. Это основные положения (мысли) текста, которые объясняются, доказываются, поясняются в тексте. Основные требования к содержанию тезисов:

– обоснование актуальности темы;

– главный тезис;

– аргументация, доказательства и факты, подтверждающие выдвинутый тезис;

– основные выводы.

Обратимся теперь к характеристикам статьи. Статья представляет собой самостоятельный научный текст, где исследователь излагает собственные мысли по проблеме. Структура статьи схожа со структурой текста научного сочинения, но представляет его как бы в миниатюре. В начале статьи выдвигается ее главный тезис, который затем подвергается аргументированному доказательству в основной части. В заключение статьи помещаются выводы, подтверждающие либо опровергающие все вышесказанное. Заметим, что обе эти формы – и статья, и тезисы – создаются на основе текста собственно научного сочинения, где подробно рассматривается весь ход исследования и описываются его результаты. Поэтому особое внимание мы уделим именно этому главному, фундаментальному варианту оформления итогов научной работы.

Начинается оно с компоновки подготовленных текстов по главам в соответствии с примерной структурой работы. После того как главы сформированы, следует их внимательно прочитать и отредактировать как с точки зрения орфографии и синтаксиса, так и по содержанию (сверить цифры и факты, сноски, цитаты и т. п.). Сразу же после прочтения каждой главы и осуществления правки приступают к написанию выводов к соответствующей главе. Вывод по главе обычно содержит изложение сущности вопроса, разбираемого в ней, и обобщение результатов проделанного анализа.

Далее составляется заключение по всей работе. Только после этого приступают к составлению окончательного варианта введения к работе. Представим этот процесс: первоначальный вариант введения ко всей работе, компоновка текстов, редактирование текста, выводы к каждой главе, составление библиографического списка, общее заключение, редактирование и составление окончательного варианта введения.

Библиографический список представляет собой перечень книг и статей в периодических изданиях, расположенных в алфавитном порядке по фамилиям авторов или названиям коллективных трудов без указания авторов на титульном листе.

Рассмотрим более детально правила оформления основных структурных элементов научно-исследовательской работы. Титульный лист является первой страницей научной работы и заполняется по определенным правилам. Они предполагают указание автора работы, название темы работы, фамилии, имени, отчества и должности, научной степени и звания научного руководителя. Оглавление следует за титульным листом. Оно включает в себя указание на основные элементы работы: введение, главы, параграфы, заключение, список литературы, приложения.

Введение представляет собой наиболее ответственную часть научной работы, так как содержит в сжатой форме все основные, фундаментальные положения, обоснованию и проверке которых посвящено исследование. Введение должно включать в себя: формулировку темы, описание актуальности исследования, проблемы исследования, объекта, предмета, цели и задач исследования, гипотезы и методов исследования, этапов и структуры исследования. Во введении даётся характеристика работы – относится ли она к теоретическим исследованиям или к прикладным; сообщается, в чем заключается практическая значимость и (или) прикладная ценность полученных результатов; приводится характеристика источников для написания работы и краткий обзор имеющейся по данной теме литературы.

Объем введения по отношению ко всей работе небольшой и обычно составляет 2–3 страницы. Текст научно-исследовательской работы делится на крупные главы и мелкие параграфы, части. Существует еще один, простейший, способ рубрикации внутри текста: с помощью абзацев – отступов вправо в строке при начале новой смысловой части. Абзацы – это своеобразный композиционный прием, позволяющий более зримо обозначить логические акценты в тексте.

Основная (содержательная) часть работы может содержать 2–3 главы. (Название этой части как основной скорее связано с ее большим, чем

у остальных частей, объемом, нежели со значением, так как, например, введение является ничуть не менее значимой частью работы.) Глава 1 обычно содержит итоги анализа специальной литературы, теоретическое обоснование темы исследования; главы 2–3 описывают практические этапы работы, интерпретацию данных, выявление определенных закономерностей в изучаемых явлениях в ходе эксперимента. Каждая глава завершается выводами. Содержание основной части должно точно соответствовать теме работы и полностью её раскрывать.

Заключение обычно составляет не больше 1–2 страниц. Основное требование к заключению: оно не должно дословно повторять выводы по главам. В заключении формулируются наиболее общие выводы по результатам исследования и предлагаются рекомендации. Отмечается степень достижения цели, обозначаются перспективы дальнейших исследований. При этом должна быть подчёркнута самостоятельность выводов, новизна, теоретическое и (или) практическое (прикладное) значение полученных результатов.

Особой точности требует составление библиографического списка (списка использованной литературы). В переводе с древнегреческого языка библиография – значит «описание книги». Библиографический список – это список изученной по теме литературы, представленный специальным образом. Наиболее удобен в исследовательской работе учащихся алфавитный (по алфавиту фамилий авторов или заглавий) способ группировки литературных источников. В список литературы входят все использованные в работе источники. Сведения о книгах (монографиях, учебниках, справочниках и т. д.) должны включать следующие необходимые элементы: фамилию, инициалы автора; заглавие; данные о последующих изданиях; место издания, издательство; год издания и объем в страницах.

3.2 Правила оформления списка литературы

Оформление результатов учебной или научной работы (реферат, курсовая работа, дипломная работа, отчет по учебно-исследовательской работе,

научная статья, доклад, диссертация) является одним из важнейших этапов научно-исследовательской деятельности. Этап оформления библиографической части рукописи работы включает:

– использование цитат и ссылок;

– оформление списка литературы;

– библиографическое описание документов в этом списке.

Список литературы – органическая часть любой научной работы. Список включает в себя цитируемые в данной работе, просмотренные произведения, архивный материал, имеющий отношение к теме. Варианты расположения литературы в списке:

– алфавитное;

– по типам документов;

– систематическое;

– по мере использования (по главам и разделам);

– хронологическое и др.

3.2.1 Схема библиографического описания документа для списка литературы

Библиографическое описание на книгу или любой другой документ составляется по определенным правилам и регламентируется:

– ГОСТ 7.1–2003 «Библиографическая запись. Библиографическое описание. Общие требования и правила составления»;

– ГОСТ 7.82–2001 «Библиографическая запись. Библиографическое описание электронных ресурсов»;

– ГОСТ 7.12–93 «Библиографическая запись. Сокращение слов на русском языке. Общие требования и правила»;

– ГОСТ 7.11–2004 «Сокращение слов и словосочетаний на иностранных европейских языках в библиографическом описании».

В состав библиографического описания входят следующие области:

1) область заглавия и сведений об ответственности;

2) область издания;

3) область специфических сведений;

4) область выходных данных;

5) область физической характеристики;

6) область серии;

7) область примечания;

8) область стандартного номера (ISBN) и условий доступности.

Область стандартного номера (ISBN) при написании курсовых и дипломных работ является необязательной.

Пунктуация в библиографическом описании выполняет две функции – обычных грамматических знаков препинания и знаков предписанной пунктуации, т. е. знаков, имеющих опознавательный характер для областей и элементов библиографического описания. Предписанная пунктуация предшествует элементам и областям или заключает их. Ее употребление не связано с нормами языка.

В качестве предписанной пунктуации выступают знаки препинания и математические знаки:

. – точка и тире

. точка

, запятая

: двоеточие

; точка с запятой

… многоточие

/ косая черта

// две косые черты

( ) круглые скобки

[ ] квадратные скобки

+ знак плюс

= знак равенства

В конце библиографического описания ставится точка. Каждой области описания, кроме первой, предшествует знак точка и тире, который ставится перед первым элементом области. Для более четкого разделения областей и элементов, а также для различения предписанной и грамматической пунктуации применяют пробелы в один печатный знак до и после предписанного знака. Исключение составляют точка и запятая – пробелы оставляют только после них.

Краткая схема библиографического описания (описание состоит из обязательных элементов) схематично может быть представлена так:

Заголовок описания. Основное заглавие : сведения, относящиеся к заглавию / Сведения об ответственности. – Сведения об издании. – Место издания, дата издания. – Объем. – ISBN.

Заголовок может включать имя лица (имя лица – условно применяемое понятие, включающее фамилию, инициалы или имя и отчество, псевдоним, личное имя или прозвище в качестве фамилии), наименование организации, унифицированное заглавие произведения, обозначение документа, географическое название, иные сведения. Заголовок применяют при составлении записи на произведение одного, двух (трех авторов).

Произведения одного, двух (трех авторов)

Кириллов, В. И. Логика : учебник для юрид. вузов / В. И. Кириллов ; Моск. гос. юрид. академия. – 6-е изд., перераб. и доп. – М. : Проспект, 2009. – 233 с. – ISBN 978-5-392-00353-2.

Грушевицкая, Т. Г. Культурология : учебник для вузов / Т. Г. Грушевицкая, А. П. Садохин. – 3-е изд., перераб. и доп. – М. : ЮНИТИ-ДАНА, 2008. – 687 с. – (Cogito ergo sum). – ISBN 978-5-238-01058-8.

Произведения четырех авторов и более

В случае наличия четырех и более авторов запись составляют под заглавием произведения, а в сведениях об ответственности ограничиваются указанием первого автора с добавлением в квадратных скобках сокращения «и другие» [и др.] или его эквивалента на латинском языке [et al.].

Философия : учебник / Н. С. Савкин [и др.] ; отв. ред. Н. С. Савкин. – 3-е изд., испр. – Саранск : Морд. ун-т, 2004. – 355 с. – (Учебники Мордовского университета). – ISBN 5-7103-0712-2.

Многотомные издания. Документ в целом

Казьмин, В. Д. Справочник домашнего врача : в 3 т. / В. Казьмин. – М. : АСТ : Астрель, 2001.

Многотомные издания. Отдельный том

Казьмин, В. Д. Справочник домашнего врача : в 3 т. / В. Д. Казьмин. – М. : АСТ : Астрель, 2002. – Т. 2 : Детские болезни. – 303 с.

Основным заглавием является заглавие книги или статьи, а сведением, относящимся к заглавию, – пояснение жанра, типа издания, например сборник статей, учебное пособие и т. п. (Философия : учебник…).

Сведения об ответственности – это сведения о соавторах, переводчиках, редакторах и/или о той организации, которая принимает на себя ответственность за данную публикацию.

Первым сведениям об ответственности предшествует знак косая черта; последующие группы сведений отделяют друг от друга точкой с запятой. Однородные сведения внутри группы отделяют запятыми.

Русский традиционный костюм : ил. энциклопедия / авт.-сост. Н. Соснина, И. Шангина ; предисл. И. Шангиной…

Сведения об издании включают качественную и количественную характеристику документа – переработанное, стереотипное, 2-е и т. п.

Место издания – наименование города. Москва, Санкт-Петербург, Ростов на Дону сокращаются (М., СПб., Ростов н/Д), все остальные города пишутся полностью (Новосибирск, Киев).

Объем – это количество страниц или страницы, на которых опубликована статья в журнале или сборнике.

Унифицированные формы сокращений, применяемые в отдельных положениях, приводят на русском либо латинском языках:

и другие (et alii) – и др. (et al.);

и так далее (et cetera) – и т. д. (etc.);

то есть (id est) – т. е. (i.e.);

без места (sine loco) – б. м. (s.l.);

без издателя (sine nomine) – б. и. (s.n.)

3.2.2 Схема аналитического описания документа для списка литературы

Аналитическим описание источников называется потому, что оно основано на принципе разделения источника на части, например журнал может быть разделен на статьи, а книга – на главы.

Схема аналитического описания такова:

Сведения о составной части документа // Сведения о документе, в котором помещена составная часть.

Примеры

Произведения одного, двух (трех авторов)

Казаков, Н. А. Запоздалое признание : повесть / Н. Казаков // На боевом посту. – 2000. – № 9. – С. 64–76.

Баталов, А. Л. Сакральная топография средневекового города / А. Л. Баталов, Л. А. Беляев // Известия Института христианской культуры средневековья. – М., 1998. – Т. 1. – С. 13–22.

Боголюбов, А. Н. О вещественных резонансах в волноводе с неоднородным заполнением / А. Н. Боголюбов, А. Л. Делицын, М. Д. Малых // Вестник Московского университета. Сер. 3, Физика. Астрономия. – 2001. – № 5. – С. 23–25. – ISSN 0201-7385.

Произведения четырех авторов и более

Мониторинг состояния оборудования систем связи в трубопроводном транспорте нефти / Л. И. Григорьев [и др.] // Автоматизация, телемеханизация и связь в нефтяной промышленности. – 2007. – № 5. – С. 3–8.

Раздел, глава

Малый, А. И. Введение в законодательство Европейского сообщества / А. И. Малый // Институты Европейского союза : учеб. пособие / А. И. Малый, Дж. Кемпбелл, М. О'Нейл. – Архангельск : Правда Севера, 2002. – Разд. 1. – С. 7–26.

Материалы конференции

Моисеева, М. В. Японская мультипликация и ее восприятие в России / М. В. Моисеева, Е. В. Кудряшева // Социокультурная миссия университета в современном обществе : сб. тр. участников I Всерос. научно-практической конф. «Человек, культура, образование» (Ульяновск, 2–4 июля 2005 г.). – Ульяновск, 2006. – С. 142–145.

Законодательные акты. Запись под заголовком

Российская Федерация. Конституция (1993). Конституция Российской Федерации [Текст] : офиц. текст. – М. : Маркетинг, 2001. – 39 с. – ISBN 5-94462-025-0.

Законодательные акты. Запись под заглавием

Конституция Российской Федерации [Текст]. – М. : Маркетинг, 2001. – 39 с. – ISBN 5-94462-025-0.

Российская Федерация. Законы. Семейный кодекс Российской Федерации [Текст] : [федер. закон : принят Гос. Думой 8 дек. 1995 г. : по состоянию на 3 янв. 2001 г.]. – СПб. : Victory : Стаун-кантри, 2001. – 94 с.

Патентные документы

В области специфических сведений указывают регистрационный номер заявки на патентный документ; дату ее подачи (поступления); дату публикации и (или) сведения об официальном издании, в котором опубликованы сведения о патентном документе; сведения о конвенционном приоритете: дату подачи заявки, номер и название страны конвенционного приоритета.

Приемопередающее устройство [Текст] : пат. 2187888 Российская Федерация : МПК H 04 B 1/38, H 04 J 13/00 / Чугаева В. И. ; заявитель и патентообладатель Воронеж. науч.-ислед. ин-т связи. – № 2000131736/09 ; заявл. 18.12.00 ; опубл. 20.08.02, Бюл. № 23 (II ч.). – 3 с.

или:

Пат. 2187888 Российская Федерация, МПК7 H 04 B 1/38, H 04 J 13/00. Приемопередающее устройство [Текст] / Чугаева В. И. ; заявитель и патентообладатель Воронеж. науч.-ислед. ин-т связи. – № 2000131736/09 ; заявл. 18.12.00 ; опубл. 20.08.02, Бюл. № 23 (II ч.). – 3 с.

ГОСТЫ

Стекло безопасное для наземного транспорта. Общие технические условия [Текст] : ГОСТ 5727–88. – Изд. сент. 2001 с Изм. 1, 2, 3 (ИУС. 1992. № 2; ИУС. 1999. № 1; ИУС. 2001. № 11). – Взамен ГОСТ 5727–83 ; введ. 01.01.90.

или:

ГОСТ Р 517721–2001. Аппаратура радиоэлектронная бытовая. Входные и выходные параметры и типы соединений. Технические требования [Текст]. – Введ. 2002-01-01. – М. : Изд-во стандартов, 2001. – IV, 27 с. : ил. ; 29 см.

К настоящему времени в России уже существует практика включения ссылок на ресурсы Интернет в научных и исследовательских работах студентов. При этом стоит придерживаться несложных правил, связанных со спецификой сетевых документов.

Первое правило состоит в том, что нет необходимости ссылаться на электронный документ, в случае существования его печатного аналога.

Второе правило заключается в необходимости сохранения на своем компьютере или даже распечатке копий электронных документов.

Третье правило связано с умением вычленять из огромного массива сетевых документов именно те, на которые имеет смысл ссылаться.

Примеры библиографических записей электронных ресурсов

Ресурсы локального доступа

Сидыганов, В. У. Модель Москвы [Электронный ресурс] : электрон, карта Москвы и Подмосковья / В. У. Сидыганов, С. Ю. Толмачев, Ю. Э. Цыганков. – Версия 2.0. – Электрон, дан. и прогр. – М. : FORMOZA, 1998. – 1 электрон, опт. диск (CD-ROM).

Атлас-98 [Электронный ресурс] : 3D., 1998. – 1 электрон, опт. диск (CD-ROM).

Сетевые ресурсы

Исследовано в России [Электронный ресурс] : многопредмет. науч. журн. / Моск. физ.-техн. ин-т. – Электрон. журн. – Режим доступа: http://zhumal.mipt. rssi.ru

Электронный каталог ГПНТБ России [Электронный ресурс] : база данных. – Режим доступа : http://www.gpntb.ru/win/search/help/ el-cat.html

Библиографические ссылки (цитаты)

Современное научное творчество, даже при наличии явной научной или художественной творческой новизны, «стоит на плечах гигантов». Учебные работы и диссертационные исследования в еще большей степени требуют компиляции чужого или его опровержения. Это требует грамотного оформления цитат и ссылок.

ГОСТ Р 7.0.5–2008. Библиографическая ссылка. Общие требования и правила составления. Дата введения – 1 января 2009 года.

Режим доступа: http://www.bookchamber.ru/GOST_P_7.0.5.-2008

Настоящий стандарт устанавливает общие требования и правила составления библиографической ссылки: основные виды, структуру, состав, расположение в документах.

Стандарт распространяется на библиографические ссылки, используемые в опубликованных и неопубликованных документах на любых носителях.

Библиографическая ссылка – совокупность библиографических сведений о цитируемом в тексте документе, необходимых для общей характеристики, идентификации и поиска документа.

По месту расположения в документе различают библиографические ссылки:

– внутритекстовые, помещенные в тексте документа;

– подстрочные, вынесенные из текста вниз полосы документа (в сноску);

– затекстовые, вынесенные за текст документа или его части (в выноску).

При повторе ссылок на один и тот же объект различают библиографические ссылки:

– первичные, в которых библиографические сведения приводятся впервые в данном документе;

– повторные, в которых ранее указанные библиографические сведения повторяют в сокращенной форме.

Если текст цитируется не по первоисточнику, а по другому документу, то в начале ссылки приводят слова: «Цит. по: » (цитируется по), «Приводится по: », с указанием источника заимствования:

Цит. по: Флоренский П. А. У водоразделов мысли. М., 1990. Т. 2. С. 27.

Примеры внутритекстовых библиографических ссылок

(Аренс В. Ж. Азбука исследователя. М. : Интермет Инжиниринг, 2006).

(Потемкин В. К., Казаков Д. Н. Социальное партнерство: формирование, оценка, регулирование. СПб., 2002. 202 с.).

Примеры подстрочных библиографических ссылок

1 Тарасова В. И. Политическая история Латинской Америки. М., 2006. С. 305.

17 История Российской книжной палаты, 1917–1935. М., 2006.

Или более подробно:

1 Тарасова В. И. Политическая история Латинской Америки : учеб. для вузов. – 2-е изд. – М. : Проспект, 2006. – С. 305–412.

17 История Российской книжной палаты, 1917–1935 / Р. А. Айгистов [и др.]. – М. : Рос. кн. палата, 2006. – 447 с. – ISBN 5-901202-22-8.

Для аналитических записей допускается, при наличии в тексте библиографических сведений о составной части, в подстрочной ссылке указывать только сведения об идентифицирующем документе:

2 Адорно Т. В. К логике социальных наук // Вопр. философии. – 1992. – № 10. – С. 76–86.

или, если о данной статье говорится в тексте документа:

2 Вопр. философии. 1992. № 10. С. 76–86.

Для записей на электронные ресурсы допускается при наличии в тексте библиографических сведений, идентифицирующих электронный ресурс удаленного доступа, в подстрочной ссылке указывать только его электронный адрес:

2 Официальные периодические издания : электрон. путеводитель / Рос. нац. б-ка, Центр правовой информации. [СПб.], 2005–2007. URL: http://www.nlr.ru/lawcenter/izd/index.html (дата обращения: 18.01.2007).

или, если о данной публикации говорится в тексте документа:

2 URL: http://www.nlr.ru/lawcenter/izd/index.html

При нумерации подстрочных библиографических ссылок применяют единообразный порядок для всего данного документа: сквозную нумерацию по всему тексту, в пределах каждой главы, раздела, части и т. п., или – для данной страницы документа.

Примеры затекстовых библиографических ссылок

Совокупность затекстовых библиографических ссылок оформляется как перечень библиографических записей, помещенный после текста документа или его составной части.

34. Никонов В. И., Яковлева В. Я. Алгоритмы успешного маркетинга. М., 2007. С. 256–300.

Для связи с текстом документа порядковый номер библиографической записи в затекстовой ссылке указывают в знаке выноски, который набирают на верхнюю линию шрифта, или в отсылке, которую приводят в квадратных скобках в строку с текстом документа.

В тексте:

Общий список справочников по терминологии, охватывающий время не позднее середины XX века, дает работа библиографа И. М. Кауфмана [59].

В затекстовой ссылке:

59 Кауфман И. М. Терминологические словари : библиография. М., 1961.

ИЛИ

В тексте:

Общий список справочников по терминологии, охватывающий время не позднее середины XX века, дает работа библиографа И. М. Кауфмана [59].

В затекстовой ссылке:

59. Кауфман И. М. Терминологические словари : библиография. М., 1961.

Примеры затекстовых ссылок на документ, созданный одним-тремя авторами и четырьмя и более авторами:

В тексте:

[Пахомов, Петрова]

В затекстовой ссылке:

Пахомов В. И., Петрова Г. П. Логистика. М. : Проспект, 2006. 232 с.

В тексте:

[Нестационарная аэродинамика баллистического полета]

В затекстовой ссылке:

Нестационарная аэродинамика баллистического полета / Ю. М. Липницкий [и др.]. М., 2003. 176 с.

В отсылке допускается сокращать длинные заглавия, обозначая опускаемые слова многоточием с пробелом до и после этого предписанного знака:

В тексте:

[Философия культуры …, с. 176]

В затекстовой ссылке:

Философия культуры и философия науки: проблемы и гипотезы : межвуз. сб. науч. тр. / Сарат. гос. ун-т ; [под ред. С. Ф. Мартыновича]. Саратов : Изд-во Сарат. ун-та, 1999. 199 с.

Если ссылку приводят на многочастный (многотомный) документ, в отсылке указывают также обозначение и номер тома (выпуска, части и т. п.):

В тексте:

[Целищев, ч. 1, с. 17]

В затекстовой ссылке:

Целищев В. В. Философия математики. Новосибирск : Изд-во НГУ, 2002. Ч. 1–2.

Повторная библиографическая ссылка

Повторную ссылку на один и тот же документ или его часть приводят в сокращенной форме при условии, что все необходимые для идентификации и поиска этого документа библиографические сведения указаны в первичной ссылке на него. Выбранный прием сокращения библиографических сведений используется единообразно для данного документа.

Предписанный знак точку и тире, разделяющий области библиографического описания, в повторной библиографической ссылке заменяют точкой.

Допускается сокращать длинные заглавия, обозначая опускаемые слова многоточием с пробелом до и после этого предписанного знака:

Внутритекстовые ссылки:

Первичная	(Васильев С. В. Инновационный маркетинг. М., 2005)
Повторная	(Васильев С. В. Инновационный маркетинг. С. 62)

Первичная	(Герасимов Б. Н., Морозов В. В., Яковлева Н. Г. Системы управления: понятия, структура, исследование. Самара, 2002)
Повторная	(Герасимов Б. Н., Морозов В. В., Яковлева Н. Г. Системы управления … С. 53–54)

Подстрочные ссылки:

Первичная	1 Гаврилов В. П., Ивановский С. И. Общество и природная среда. М. : Наука, 2006. 210 с.
Повторная	15 Гаврилов В. П., Ивановский С. И. Общество и природная среда. С. 81.

Первичная	2 Геоинформационное моделирование территориальных рынков банковских услуг / А. Г. Дружинин [и др.]. Шахты : Изд-во ЮРГУЭС, 2006.
Повторная	6 Геоинформационное моделирование … С. 28.

Затекстовые ссылки:

Первичная	57. Шапкин А. С. Экономические и финансовые риски : оценка, управление, портфель инвестиций. Изд. 3-е. М., 2004. 536 с.
Повторная	62. Шапкин А. С. Экономические и финансовые риски. С. 302.

Первичная	97 Анастасевич В. Г. О необходимости в содействии русскому книговедению // Благонамеренный. 1820. Т. 10, № 7. С. 32–42.
Повторная	112 Анастасевич В. Г. О необходимости в содействии … С. 38.

Повторные ссылки многочастного документа

Внутритекстовые ссылки:

Первичная	(Пивинский Ю. Е. Неформованные огнеупоры. М., 2003. Т. 1, кн. 1 : Общие вопросы технологии. 447 с.)
Повторная	(Пивинский Ю. Е. Неформованные огнеупоры. Т. 1, кн. 2. С. 25)

Подстрочные ссылки:

Первичная	1 Фотометрия и радиометрия оптического излучения. М. : Наука, 2002. Кн. 5 : Измерения оптических свойств веществ и материалов, ч. 2 : Колориметрия. Рефрактометрия. Поляриметрия. Оптическая спектрометрия в аналитике / В. С. Иванов [и др.]. 305 с.
Повторная	2 Фотометрия и радиометрия оптического излучения. Кн. 5, ч. 2. С. 158–159.

Затекстовые ссылки:

Первичная	86. Труды Института геологии / Рос. акад. наук, Урал. отд-ние, Коми науч. центр, Ин-т геологии. Вып. 113 : Петрология и минералогия Севера Урала и Тиммана. 2003. 194 с.
Повторная	105. Труды Института геологии. Вып. 113. С. 97.

При последовательном расположении первичной и повторной ссылок текст повторной ссылки заменяют словами «Там же» или «Ibid.» (ibidem) для документов на языках, применяющих латинскую графику. В повторной ссылке на другую страницу к словам «Там же» добавляют номер страницы, в повторной ссылке на другой том (часть, выпуск и т. п.) документа к словам «Там же» добавляют номер тома:

Внутритекстовые ссылки:

Первичная	(Коваленко Б. В., Пирогов А. И., Рыжов О. А. Политическая конфликтология. М., 2002. С. 169–178)
Повторная	(Там же)

Первичная	(Kriesberg L. Constructiv conflicts: from escalaition to resolution. Lanham, 1998)
Повторная	(Ibid.)

Подстрочные ссылки:

Первичная	18 Фенухин В. И. Этнополитические конфликты в современной России: на примере Северо-Кавказского региона : дис. … канд. полит. наук. М., 2002. С. 54–55.
Повторная	19 Там же. С. 68.

Первичная	37 Служебный каталог чешуекрылых. Владимир : Нац. парк «Мещера», 2006. С. 132–136.
Повторная	38 Там же. С. 157. 39 Там же. С. 164.

Затекстовые ссылки:

Первичная	52. Россия и мир : гуманитар. проблемы : межвуз. сб. науч. тр. / С.-Петерб. гос. ун-т вод. коммуникаций. 2004. Вып. 8. С. 145.
Повторная	53. Там же. Вып. 9. С. 112.

В повторных ссылках, содержащих запись на один и тот же документ, созданный одним, двумя или тремя авторами, не следующих за первичной ссылкой, приводят заголовок, а основное заглавие и следующие за ним повторяющиеся элементы заменяют словами «Указ. соч.» (указанное сочинение), «Цит. соч.» (цитируемое сочинение), «Op. cit.» (opus citato – цитированный труд) – для документов на языках, применяющих латинскую графику. В повторной ссылке на другую страницу к словам «Указ. соч.» (и т. п.) добавляют номер страницы, в повторной ссылке на другой том (часть, выпуск и т. п.) документа к словам «Указ. соч.» добавляют номер тома.

Подстрочные ссылки:

Первичная	8 Дживилегов А. К. Армия Великой Французской революции и ее вожди : ист. очерк / Гос. публ. ист. б-ка. М., 2006. С. 151–172.
Повторная	11 Дживилегов А. К. Указ. соч. С. 163.

Затекстовые ссылки:

Первичная	29. Гришаева Л. И., Цурикова Л. В. Введение в теорию межкультурной коммуникации : учеб. пособие для вузов. 3-е изд. М. : Academia, 2006. 123 с. (Высшее профессиональное образование. Языкознание)
Повторная	33. Гришаева Л. И., Цурикова Л. В. Указ. соч. С. 98.

Особенности составления библиографических ссылок на архивные документы

Все элементы поисковых данных документа разделяют точками:

ОР РНБ. Ф. 316. Д. 161. Л. 1.

РО ИРЛИ. Ф. 568. Оп. 1. № 196. Л. 18–19 об.

Название архивохранилища приводят в виде аббревиатуры:

РГИА

РГАДА

ОАД РНБ или Отд. арх. документов РНБ

Арх. РГБ

Арх. кинофонофотодокументов

НА РТ (Нац. арх. Республики Татарстан)

Обозначение и номер фонда приводят после названия архивохранилища. В качестве обозначения архивного фонда, описи, порядкового номера дела (единицы хранения), зафиксированного в архивной описи, используют сокращения: «ф.» (фонд), «оп.» (опись), «д.» (дело), «ед. хр.» (единица хранения), «№» (номер), «об.» (оборот) – в зависимости от того, какие обозначения приняты в данном архивохранилище:

ОР РНБ. Ф. 416. Оп.1. Д. 26. Л. 1.

ОАД РНБ. Ф. 1. Оп. 1. 1856 г. Ед. хр. 21.

После номера фонда в круглых скобках может быть указано его название:

ОР РГБ. Ф. 573 (Б. С. Боднарский).

Внутритекстовые ссылки на архивные документы, как правило, содержат поисковые данные документа и лишь в случае необходимости описание документа полностью:

(ЦГАИПД. Ф. 1728. Ед. хр. 537079).

(Личное дело П. И. Болдина // ЦГАИПД. Ф. 1728. Ед. хр. 537079).

В подстрочных и затекстовых ссылках могут быть приведены сведения о документе – заголовок и основное заглавие документа или только основное заглавие, зафиксированные в описи фонда, в названии дела или в конкретном документе, хранящемся в деле. В этом случае сведения о самом документе отделяют от поисковых сведений о документе знаком две косые черты с пробелами до и после него.

7 Розанов И. Н. Как создавалась библиотека Исторического музея : докл. на заседании Ученого совета Гос. публ. ист. б-ки РСФСР 30 июня 1939 г. // ГАРФ. Ф. А-513. Оп. 1. Д. 12. Л.14.

Таким образом, в отличие от оформления библиографических записей в списке литературы, в оформлении библиографических ссылок следует учитывать следующие особенности:

– в заголовке может быть указано любое количество авторов;

– заголовок, содержащий наименование организации, применяется в исключительных случаях, официальные документы чаще представлены библиографическим описанием, начинающимся с заглавия;

– тире между областями библиографического описания может быть опущено;

– в библиографическом описании могут применяться дополнительные сокращения (при условии, что прилагается перечень сокращений);

– составитель может произвольно расширить или сузить состав библиографического описания за счет факультативных элементов.

4 ЗАЩИТА РЕЗУЛЬТАТОВ ИССЛЕДОВАНИЯ

4.1 Доклад

После окончания исследования, оформления полученных результатов, прочтения и одобрения их научным руководителем наступает последний этап – защита. К сожалению, важность этого этапа иногда недооценивается, и тогда даже качественно проведенное исследование выглядит при публичном представлении неубедительно. Автор или «заваливает» аудиторию и жюри объемом информации, или на ходу пытается выстроить логику своего изложения. В итоге – «смазанное» представление у слушателей и чувство неудовлетворенности у выступающего. И напротив, искусно подготовленный доклад по защите написанной работы может «затенить» некоторые его недостатки и таким образом повысить шансы выступающего на хорошую оценку.

Следует помнить, что типично, на все выступление отводится не более 5–10 мин. По регламенту можно рассчитывать дополнительно на 1–2 мин, но не более. Ни о теме (ее уже объявили), ни о том, что было прочитано (список литературы), говорить не следует. Защита ни в коем случае не должна сводиться к пересказу всего содержания работы. Если вы не сумели заинтересовать аудиторию за отведенное по регламенту время, его продолжение только усилит непонимание и раздражение слушателей.

Подготовку доклада лучше всего начать с продумывания его структуры. Четкое и ясное представление о работе у самого докладчика – залог понимания его аудиторией. Доклад можно разделить на 3 части, состоящие из отдельных, но связанных между собой блоков.

Первая часть по сути кратко повторяет введение исследовательской работы. Здесь обосновывается актуальность выбранной темы, описывается научная проблема, формулируются задачи исследования и указываются основные методы. Для того чтобы ваш доклад вызвал интерес аудитории,

очень важно правильно настроить слушателей с самого начала вашего выступления. Существует несколько способов привлечения внимания аудитории, вот некоторые из них: вы можете начать выступление с приведения примера, интересной цитаты, образного сравнения предмета выступления с конкретным явлением, с истории, случая, задания проблемы или оригинального вопроса.

Во второй части, самой большой по объему, вам нужно представить содержание глав. Особое внимание комиссия обращает на итоги проведенного исследования, на личный вклад в него автора. Поэтому не забудьте после краткого изложения содержания глав реферата отдельно подчеркнуть, в чем состоит новизна предлагаемой вами работы, это могут быть использованные впервые по отношению к данному материалу методики, достигнутые вами результаты исследования.

При изложении основных результатов можно использовать заранее подготовленные схемы, чертежи, графики, таблицы, видеоролики, слайды, видеофильмы. Демонстрируемые материалы должны оформляться так, чтобы они не перегружали выступление и были видны всем присутствующим в аудитории.

В третьей части целесообразно кратко изложить основные выводы по результатам исследования, не повторяя тех выводов, которые уже сделаны в ходе изложения содержания по главам. Постарайтесь в заключении создать кульминацию выступления, предложите слушателям поразмышлять над проблемой, покажите возможные варианты дальнейших исследований, используйте цитату по теме реферата известного ученого.

Особое внимание обращается на речь докладчика. Она должна быть ясной, грамматически точной, уверенной, выразительной. Если докладчик старается говорить быстро, проглатывая окончания слов, тихо, невнятно, то качество его выступления снижается. Спокойное, последовательное и хорошо аргументированное изложение материала импонирует слушателям.

Но использование научного стиля отнюдь не означает пренебрежение к использованию образных сравнений, контрастов, необычных фактов, позволяющих удерживать внимание аудитории.

После того как докладчик закончил свое выступление, члены комиссии задают вопросы. Вопросы может задать и любой присутствующий на вашем выступлении. Вопросов не нужно бояться: это еще одна возможность продемонстрировать обстоятельность и глубину изучения темы. Существует представление, что задавание вопросов продиктовано исключительно желанием «потопить» докладчика. Это ошибочное мнение. Скорее всего, если вам задают вопросы, это значит, что тема заинтересовала, привлекла внимание слушателей.

При ответах на вопросы не стоит забывать о нескольких простых правилах.

Если заданный вопрос выходит за рамки исследования, не стоит на ходу придумывать ответ, не подкрепленный результатом исследования. Вполне допустимо сказать, что это не было предметом вашего исследования или что это планируется исследовать на следующем этапе. Таким образом, вы только поддержите образ вдумчивого исследователя.

Очень важным условием ответа на вопрос является правильное понимание того, что именно спрашивает оппонент. Поэтому будет целесообразно уточнить вопрос и, согласовав понимание вопроса, отвечать на него. В противном случае, есть опасность, что вы отвечаете не на вопрос, который вам задали, а на свою версию этого вопроса: не следует впадать и в другую крайность – начинать уточнять очевидные и понятные вещи. Везде хороша мера.

И еще. По сложившейся этике проведения научной дискуссии перед тем, как отвечать по существу на заданный вопрос, принято благодарить его автора. Ведь спрашивающий проявил интерес к вашей работе. Кроме того,

вопросы часто позволяют увидеть новые направления для дальнейшего исследования.

Поскольку устное выступление является своего рода сценическим искусством, т. е. включает в себя владение голосом, правильно выбранные жесты и позу, не лишней будет предварительная тренировка чтения доклада в знакомых вам условиях.

Перед тем как выступать на научных конференциях городского и более высокого уровня, желательно пройти различные уровни подразделения, отдела, предприятия или на всероссийском или международном мероприятии. Помните, что чем больше вы выступаете, тем больший опыт вы приобретаете. Вы можете попросить прослушать ваш доклад друзей или родителей. Можно читать его и самостоятельно – для себя, но обязательно вслух. При этом контролируйте время своего «выступления»: это поможет вам ориентироваться в условиях настоящего выступления (т. е. правильно выбрать темп и интонацию речи). Можно потренировать свою речь, умение выступать перед аудиторией, записывая ее на аудио- или видеокассету. После чего постараться оценить ее (как выступление другого) по следующим характеристикам:

- логичность;
- точность;
- ясность;
- доступность;
- убедительность;
- интересность;
- выразительность;
- уверенность;
- контакт со слушателями;
- уместность жестов;

– выражение лица и т. д.

Чтобы доклад был интересным и убедительным, следует снабжать теоретические положения и выводы примерами текстов, стараться использовать простые предложения, как можно более точные формулировки. Меняя темп и интонацию речи в соответствии со смыслом читаемого (произносимого), можно избежать монотонности выступления.

В заключение хотелось бы отметить, что на сегодняшний день многие вопросы методологии организации научно-исследовательской деятельности остаются недостаточно разработанными и ясными, другими словами, представляют собой широкий фронт для творческого поиска. Мы попытались указать те моменты (правила, рекомендации), которые признаны оптимальными для проведения исследовательской работы виднейшими специалистами, занимавшимися интересующей нас проблемой. Однако стадия становления, в которой находится ее изучение, позволяет нам не только рекомендовать уже испытанные традиционные формы и приемы работы, но и пригласить начинающих исследователей к самостоятельному поиску.

4.2 Подготовка мультимедийных презентаций

Любой доклад невозможно представить без презентации. По способу организации презентации можно классифицировать следующим образом:

- Интерактивные презентации.
- Непрерывно выполняющиеся презентации.

Интерактивные презентации – это презентации, ход которых управляется пользователем. Для них тоже должен существовать сценарий, но он допускает многовариантность развития действия в зависимости от вопросов пользователей.

Непрерывно выполняющиеся презентации – это нечто вроде демонстрации фильма, пользователь никак не влияет на проведение презентации.

Классификация презентации по направлению деятельности:

– Информационные – представляют основные сведения о направлениях деятельности, товаре, обзоре, достижениях и др.

– Торговые презентации. Представляют всю необходимую информацию о товарах или услугах при заключении сделок. Особенно эффективны при продаже громоздких и сложных товаров (самолеты).

– Обучающие презентации:

Обучающие семинары – презентации: знакомство с новой техникой, обзор рынка, обучение студентов и повышение квалификации и др.

Презентации для самообразования – интерактивные мультимедийные обучающие курсы.

– Корпоративные презентации:

Финансовые презентации – нацеленные на получение средств на фондовом рынке или кредитов. Освещают информацию о состоянии финансов корпорации, планах, традициях, успехах.

Презентации для служб управления персоналом: структура компании, традиции, правила, перспективы.

Другие презентации для целей корпорации.

– И множество других типов, которые характеризуют представляемый материал.

Технология создания презентаций.

Для создания эффективных компьютерных презентаций необходимо выполнить три этапа (три П):

1. Планирование:

– Составление списка вопросов и проблем, решению которых должна помочь презентация.

– Составление схемы проведения презентации.

– Создание сценария презентации.

2. Производство:

– Сбор данных.

– Создание программного продукта с помощью профессиональных инструментальных программных средств.

3. Проведение презентации:

– Установка и проверка технических средств.

– Проведение.

Так как продолжительность доклада не более 5–10 мин, то и презентация не должна содержать более *10–20 слайдов*. Плюс время на ответы слушателям и раздачу материалов, и заключительное общение.

Схема презентации

Прежде чем приступать к созданию презентации с помощью инструментальных средств, следует составить как можно более подробную схему презентации.

Схема должна состоять из трех основных блоков:

НАЧАЛО:

- Знакомство с предметом и актуальность исследования.
- Состояние предмета исследования, описание предмета исследования.
- Постановка цели и задач исследования.

ОСНОВНАЯ ЧАСТЬ:

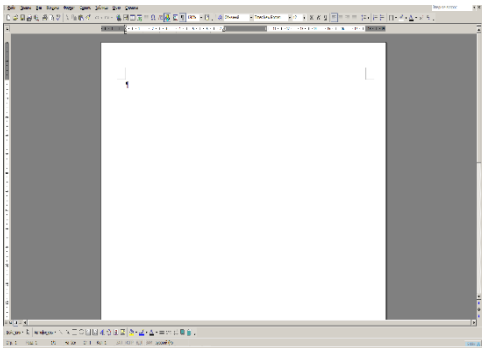
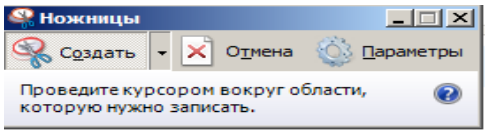
- Описание разработки предмета исследования.
- Описание научных, информационных или технических результатов, полученных в ходе исследования.
- Сравнение с аналогичными результатами.

ЗАКЛЮЧЕНИЕ:

Основные выводы по результатам работы.

Презентация и текст доклада неразрывно связаны между собой. Предварительно составляется план-проспект, который связывает эти два компонента между собой названиями слайдов и описанием. Например, это можно сделать так, как указано в таблице 4.1.

Таблица 4.1 – Пример план-проспекта описания презентации

Номер слайда	Экранная копия слайда презентации	Текст представления информации, изображенной на слайде
1		Представлена экранная копия офисной программы Word 2003. Основными элементами начальной страницы программы является строка функций, расположенная в верхней части экрана и поле ввода текста
2		Данный слайд отражает функцию получения экранных форм под названием «ножницы» операционной системы Windows-7
…	…	…

Для студентов, обучающихся с применением дистанционных образовательных технологий, лишенных возможности участия в конференции, публичного выступления перед аудиторией, защита учебно-исследовательской работы заключается в отправке преподавателю (руководителю УИР) текста доклада и мультимедийной презентации.

5 ПРОГРАММНЫЙ ПРОЕКТ

5.1 Виды программного обеспечения и их характеристики

Программное обеспечение – это совокупность программ, выполненных вычислительной системой. К программному обеспечению (ПО) относится также вся область деятельности по проектированию и разработке (ПО):

– технология проектирования программ (нисходящее проектирование; структурное программирование и др.);

– методы тестирования программ;

– методы доказательства правильности программ;

– анализ качества работы программ и др.

Программное обеспечение – неотъемлемая часть ЭВМ. Оно является логическим продолжением технических средств ЭВМ, расширяющим их возможности и сферу использования.

Существует три категории программного обеспечения (рис. 5.1).

Классификация программного обеспечения



Рис. 5.1 – Структура программного обеспечения

Прикладные программы, непосредственно обеспечивающие выполнение необходимых пользователям работ.

Системные программы:

– управление ресурсами ЭВМ;

– создание копий используемой информации;

– проверка работоспособности устройств компьютера;

– выдача справочной информации о компьютере и др.

Инструментальные программные системы, облегчающие процесс создания новых программ для компьютера. Более или менее определенно сложились следующие группы программного обеспечения:

– операционные системы;

– системы программирования;

– инструментальные системы;

– интегрированные пакеты;

– динамические электронные таблицы;

– системы машинной графики;

– системы управления базами данных (СУБД);

– прикладное программное обеспечение.

В основу работы компьютеров положен программный принцип управления, состоящий в том, что компьютер выполняет действия по заранее заданной программе. Этот принцип обеспечивает универсальность использования компьютера: в определенный момент решается задача соответственно выбранной программе. После ее завершения в память загружается другая программа и т. д.

Программа – это запись алгоритма решения задачи в виде последовательности команд или операторов языком, который понимает компьютер. Конечной целью любой компьютерной программы является управление аппаратными средствами.

Для нормального решения задач на компьютере нужно, чтобы программа была отлажена, не требовала доработок и имела соответствующую документацию. Поэтому относительно работы на компьютере часто используют термин «программное обеспечение» (software), под которым понимают

совокупность программ, процедур и правил, а также документации, касающихся функционирования системы обработки данных.

Программное и аппаратное обеспечения в компьютере работают в неразрывной связи и взаимодействии. Состав программного обеспечения вычислительной системы называется программной конфигурацией. Между программами существует взаимосвязь, т. е. работа множества программ базируется на программах низшего уровня.

Междупрограммный интерфейс – это распределение программного обеспечения на несколько связанных между собою уровней. Уровни программного обеспечения представляют собой пирамиду, где каждый высший уровень базируется на программном обеспечении предшествующих уровней.

Схематично структуру программного обеспечения можно разделить на следующие уровни: прикладной, служебный, системный и базовый.

Базовый уровень является низшим уровнем программного обеспечения. Он отвечает за взаимодействие с базовыми аппаратными средствами. Базовое программное обеспечение содержится в составе базового аппаратного обеспечения и сохраняется в специальных микросхемах постоянного запоминающего устройства (ПЗУ), образуя базовую систему ввода-вывода BIOS. Программы и данные записываются в ПЗУ на этапе производства и не могут быть изменены во время эксплуатации.

Системный уровень – переходный. Программы этого уровня обеспечивают взаимодействие других программ компьютера с программами базового уровня и непосредственно с аппаратным обеспечением. От программ этого уровня зависят эксплуатационные показатели всей вычислительной системы. При подсоединении к компьютеру нового оборудования на системном уровне должна быть установлена программа, обеспечивающая для остальных программ взаимосвязь с устройством. Конкретные программы, предназначенные для взаимодействия с конкретными устройствами, называют драйверами.

Другой класс программ системного уровня отвечает за взаимодействие с пользователем. Благодаря ему можно вводить данные в вычислительную систему, руководить ее работой и получать результат в удобной форме. Это средства обеспечения пользовательского интерфейса, от них зависит удобство и производительность работы с компьютером.

Совокупность программного обеспечения системного уровня образует ядро операционной системы компьютера. Наличие ядра операционной системы – это первое условие для возможности практической работы пользователя с вычислительной системой. Ядро операционной системы выполняет такие функции: управление памятью, процессами ввода-вывода, файловой системой, организация взаимодействия и диспетчеризация процессов, учет использования ресурсов, обработка команд и т. д.

Программы служебного уровня взаимодействуют как с программами базового уровня, так и с программами системного уровня. Назначение служебных программ (утилит) состоит в автоматизации работ по проверке и настройки компьютерной системы, а также для улучшения функций системных программ. Некоторые служебные программы (программы обслуживания) сразу входят в состав операционной системы, дополняя ее ядро, но большинство являются внешними программами и расширяют функции операционной системы. То есть в разработке служебных программ отслеживаются два направления: интеграция с операционной системой и автономное функционирование.

Можно выделить типы служебных программных средств:

1. Диспетчеры файлов (файловые менеджеры). С их помощью выполняется большинство операций по обслуживанию файловой структуры: копирование, перемещение, переименование файлов, создание каталогов (папок), уничтожение объектов, поиск файлов и навигация в файловой структуре. Базовые программные средства содержатся в составе программ системного уровня и устанавливаются вместе с операционной системой.

2. Средства сжатия данных (архиваторы). Предназначены для создания архивов. Архивные файлы имеют повышенную плотность записи информации и, соответственно, эффективнее используют носители информации.

3. Средства диагностики. Предназначены для автоматизации процессов диагностики программного и аппаратного обеспечения. Их используют для исправления ошибок и для оптимизации работы компьютерной системы.

4. Программы инсталляции (установки). Предназначены для контроля за добавлением в текущую программную конфигурацию нового программного обеспечения. Они следят за состоянием и изменением окружающей программной среды, отслеживают и протоколируют образование новых связей, утерянных во время уничтожения определенных программ. Простые средства управления установлением и уничтожением программ содержатся в составе операционной системы, но могут использоваться и дополнительные служебные программы.

5. Средства коммуникации. Разрешают устанавливать соединение с удаленными компьютерами, передают сообщения электронной почты, пересылают факсимильные сообщения и т. п.

6. Средства просмотра и воспроизведения. Преимущественно для работы с файлами их необходимо загрузить в «родную» прикладную программу и внести необходимые исправления. Но если редактирование не нужно, существуют универсальные средства для просмотра (в случае текста) или воспроизведения (в случае звука или видео) данных.

7. Средства компьютерной безопасности. К ним относятся средства пассивной и активной защиты данных от повреждения, несанкционированного доступа, просмотра и изменения данных. Средства пассивной защиты – это служебные программы, предназначенные для резервного копирования. Средства активной защиты применяют антивирусное программное обеспечение. Для защиты данных от несанкционированного доступа, их просмотра

и изменения используют специальные системы, базирующиеся на криптографии.

Программное обеспечение прикладного уровня представляет собой комплекс прикладных программ, с помощью которых выполняются конкретные задачи (производственные, творческие, развлекательные и учебные). Между прикладным и системным программным обеспечением существует тесная связь. Универсальность вычислительной системы, доступность прикладных программ и широта функциональных возможностей компьютера непосредственно зависят от типа имеющейся операционной системы, системных средств, помещенных в ее ядро, и от взаимодействия комплекса человек-программа-оборудование.

Классификация прикладного программного обеспечения

1. Текстовые редакторы. Основные функции – это ввод и редактирование текстовых данных. Для операций ввода, вывода и хранения данных текстовые редакторы используют системное программное обеспечение. С этого класса прикладных программ начинают знакомство с программным обеспечением и на нем приобретают первые навыки работы с компьютером.

2. Текстовые процессоры. Разрешают форматировать, т. е. оформлять текст. Основными средствами текстовых процессоров являются средства обеспечения взаимодействия текста, графики, таблиц и других объектов, составляющих готовый документ, а также средства автоматизации процессов редактирования и форматирования. Современный стиль работы с документами имеет два подхода: работа с бумажными документами и работа с электронными документами. Приемы и методы форматирования таких документов различаются, но текстовые процессоры способны эффективно обрабатывать оба вида документов.

3. Графические редакторы. Широкий класс программ, предназначенных для создания и обработки графических изображений. Различают три категории:

– растровые редакторы;

– векторные редакторы;

– 3D-редакторы (трехмерная графика).

В растровых редакторах графический объект представлен в виде комбинации точек (растров), которые имеют свою яркость и цвет. Такой подход эффективный, когда графическое изображение имеет много цветов и информация о цвете элементов намного важнее, чем информация об их форме. Это характерно для фотографических и полиграфических изображений. Применяют для обработки изображений, создания фотоэффектов и художественных композиций.

Векторные редакторы отличаются способом представления данных изображения. Объектом является не точка, а линия. Каждая линия рассматривается, как математическая кривая третьего порядка, и представлена формулой. Такое представление компактнее, чем растровое, данные занимают меньше места, но построение объекта сопровождается пересчетом параметров кривой в координаты экранного изображения и соответственно требует более мощных вычислительных систем. Широко применяются в рекламе, оформлении обложек полиграфических изданий.

Редакторы трехмерной графики используют для создания объемных композиций. Имеют две особенности: разрешают руководить свойствами поверхности в зависимости от свойств освещения, а также разрешают создавать объемную анимацию.

4. Системы управления базами данных (СУБД). Базой данных называют большие массивы данных, организованные в табличные структуры. Основные функции СУБД:

– создание пустой структуры базы данных;

– наличие средств ее заполнения или импорта данных из таблиц другой базы;

– возможность доступа к данным, средства поиска и фильтрации.

В связи с распространением сетевых технологий от современных СУБД требуется возможность работы с отдаленными и распределенными ресурсами, которые находятся на серверах Интернета.

5. Электронные таблицы. Предоставляют комплексные средства для хранения разных типов данных и их обработки. Основной акцент смещен на преобразование данных, предоставлен широкий спектр методов для работы с числовыми данными. Основная особенность электронных таблиц состоит в автоматическом изменении содержимого всех ячеек при изменении отношений, заданных математическими или логическими формулами.

Широкое применение находят в бухгалтерском учете, анализе финансовых и торговых рынков, средствах обработки результатов экспериментов, т. е. в автоматизации регулярно повторяемых вычислений больших объемов числовых данных.

6. Системы автоматизированного проектирования (CAD-системы). Предназначены для автоматизации проектно-конструкторских работ. Применяются в машиностроении, приборостроении, архитектуре. Кроме графических работ, разрешают проводить простые расчеты и выбор готовых конструктивных элементов из существующей базы данных.

Особенность CAD-систем состоит в автоматическом обеспечении на всех этапах проектирования технических условий, норм и правил. САПР являются необходимым компонентом для гибких производственных систем (ГВС) и автоматизированных систем управления технологическими процессами (АСУ ТП).

7. Настольные издательские системы. Автоматизируют процесс верстки полиграфических изданий. Издательские системы отличаются расширенными средствами управления взаимодействия текста с параметрами

страницы и графическими объектами, но имеют более слабые возможности по автоматизации ввода и редактирования текста. Их целесообразно применять к документам, которые предварительно обработаны в текстовых процессорах и графических редакторах.

8. Редакторы HTML (Веб-редакторы). Особый класс редакторов, объединяющих в себе возможности текстовых и графических редакторов. Предназначены для создания и редактирования веб-страниц Интернета. Программы этого класса можно использовать при подготовке электронных документов и мультимедийных изданий.

9. Браузеры (средства просмотра веб-документов). Программные средства предназначены для просмотра электронных документов, созданных в формате HTML. Воспроизводят, кроме текста и графики, музыку, человеческий язык, радиопередачи, видеоконференции и разрешают работать с электронной почтой.

10. Системы автоматизированного перевода. Различают электронные словари и программы перевода языка. Электронные словари – это средства для перевода отдельных слов в документе. Используются профессиональными переводчиками, которые самостоятельно переводят текст. Программы автоматического перевода используют текст на одном языке и выдают текст на другом, то есть автоматизируют перевод. При автоматизированном переводе невозможно получить качественный исходный текст, поскольку все сводится к переводу отдельных лексических единиц. Но, для технического текста, этот барьер снижен.

Программы автоматического перевода целесообразно использовать:

– при абсолютном незнании иностранного языка;

– при необходимости быстрого ознакомления с документом;

– для перевода на иностранный язык;

– для создания черновика, который потом будет подправлен полноценным переводом.

11. Интегрированные системы делопроизводства. Средства для автоматизации рабочего места руководителя. В частности, это функции создания, редактирования и форматирования документов, централизация функций электронной почты, факсимильной и телефонной связи, диспетчеризация и мониторинг документооборота предприятия, координация работы подразделов, оптимизация административно-хозяйственной деятельности и поставка оперативной и справочной информации.

12. Бухгалтерские системы. Имеют функции текстовых, табличных редакторов и СУБД. Предназначены для автоматизации подготовки начальных бухгалтерских документов предприятия и их учета, регулярных отчетов по итогам производственной, хозяйственной и финансовой деятельности в форме, приемлемой для налоговых органов, внебюджетных фондов и органов статистического учета.

13. Финансовые аналитические системы. Используют в банковских и биржевых структурах. Разрешают контролировать и прогнозировать ситуацию на финансовых, торговых рынках и рынках сырья, выполнять анализ текущих событий, готовить отчеты.

14. Экспертные системы. Предназначены для анализа данных, содержащихся в базах знаний, и выдачи результатов при запросе пользователя. Такие системы используются, когда для принятия решения нужны широкие специальные знания. Используются в медицине, фармакологии, химии, юриспруденции. С использованием экспертных систем связана область науки, которая носит название инженерии знаний.

15. Инженеры знаний – это специалисты, являющиеся промежуточным звеном между разработчиками экспертных систем (программистами) и ведущими специалистами в конкретных областях науки и техники (экспертами).

16. Геоинформационные системы (ГИС). Предназначены для автоматизации картографических и геодезических работ на основе информации, полученной топографическими или аэрографическими методами.

17. Системы видеомонтажа. Предназначены для цифровой обработки видеоматериалов, монтажа, создания видеоэффектов, исправления дефектов, добавления звука, титров и субтитров. Отдельные категории представляют учебные, справочные и развлекательные системы и программы. Характерной особенностью являются повышенные требования к мультимедийной составляющей.

18. Инструментальные языки и системы программирования. Эти средства служат для разработки новых программ. Компьютер «понимает» и может выполнять программы в машинном коде. Каждая команда при этом имеет вид последовательности нулей и единиц. Писать программы на машинном языке крайне неудобно. Поэтому программы разрабатываются на языке, понятном человеку (инструментальный язык или алгоритмический язык программирования), после чего специальной программой, которая называется транслятором, текст программы переводится (транслируется) на машинный код.

Трансляторы бывают двух типов: интерпретаторы и компиляторы. Интерпретатор читает один оператор программы, анализирует его и сразу выполняет, после чего переходит к обработке следующего оператора. Компилятор сначала читает, анализирует и переводит на машинный код всю программу, и только после завершения всей трансляции эта программа выполняется.

Инструментальные языки делятся на языки низкого уровня (близкие к машинному языку) и языки высокого уровня (близкие к человеческим языкам). К языкам низкого уровня принадлежат ассемблеры, а высокого – Pascal, Basic, C/C++, языки баз данных и т. д. В систему программирования, кроме транслятора, входит текстовый редактор, компоновщик, библиотека

стандартных программ, отладчик, визуальные средства автоматизации программирования. Примерами таких систем являются Delphi, Visual Basic, Visual C++, Visual FoxPro и др.

19. Системы автоматизированного проектирования. Деятельность научно-исследовательских институтов, конструкторских бюро, проектных организаций связана с разработкой новых технологий, устройств, приборов, конструкций. Проектирование сопровождается оформлением большого объема технической документации: чертежей, схем, планов.

Современные программные и технические средства вычислительной техники позволяют перейти от традиционных ручных (рутинных) методов конструирования к новым информационным технологиям проектирования с использованием ЭВМ. Для облегчения труда конструкторов, проектировщиков, изобретателей и рационализаторов разработаны системы автоматизированного проектирования (САПР). Этому термину соответствует английская аббревиатура CAD (Computer-Aided Design). Эти три буквы входят в названия многих иностранных программ, предназначенных для конструирования, черчения, трехмерного моделирования объемных объектов и оформления инженерной документации (например, P-CAD, OrCAD, AutoCAD, CADdy, ArchiCAD, T-FLEX CAD).

В отличие от автоматических систем проектирования САПР способны решать задачи, не поддающиеся полной формализации. Проектирование в таких системах является автоматизированным и осуществляется под непосредственным контролем оператора (пользователя), чаще всего в форме человеко-машинного диалога. Режим работы, в котором часть решений принимает оператор, называется интерактивным режимом.

Наиболее широко системы автоматизированного проектирования используются в электронике, электротехнике, архитектуре, строительстве, машиностроении, автомобилестроении, нефтехимической промышленности, аэрокосмической технике.

САПР образно сравнивают с «электронным кульманом». Это одно из средств для перехода к безбумажной технологии делопроизводства.

В ряде САПР из области электроники заложен принцип сквозного проектирования. При этом с помощью САПР выполняют полный цикл проектирования и производства: составление технического задания, разработку объекта, моделирование его работы, автоматизированное изготовление объекта, оформление документации. По этой причине все чаще говорят о системе CAD/CAM (Computer-Aided Design/Computer-Aided Manufacturing) – системе автоматизированного проектирования и производства, которая охватывает широкий спектр задач от начального конструирования до подготовки данных, необходимых для реального производства изделия.

Системы автоматизированного проектирования не только облегчают процесс создания и описания новых объектов, но и являются удобными справочниками, которые позволяют пользователям накапливать и хранить информацию (данные о компонентах, размерах корпусов микросхем, условные графические обозначения, сведения из стандартов и т. д.). Например, в базах данных машиностроительных САПР содержатся подробные сведения о болтах, винтах, гайках, шайбах, шпильках, штифтах и т. п. Существует большое число средств проектирования электронных устройств. Они позволяют автоматизировать разработку важного элемента радиоэлектронных устройств – печатных плат.

5.2 Жизненный цикл программного средства

Разработка программных средств имеет ряд специфических особенностей. Прежде всего, следует отметить некоторое противостояние: неформальный характер требований к ПС (постановки задачи) и понятия ошибки в нем, но формализованный основной объект разработки – программы ПС. Тем самым разработка ПС содержит определенные этапы формализации, а переход от неформального к формальному существенно неформален.

Разработка ПС носит творческий характер (на каждом шаге приходится делать какой-либо выбор, принимать какое-либо решение), а не сводится к выполнению какой-либо последовательности регламентированных действий. Тем самым эта разработка ближе к процессу проектирования каких-либо сложных устройств, но никак не к их массовому производству. Этот творческий характер разработки ПС сохраняется до самого ее конца.

Следует отметить также особенность продукта разработки. Он представляет собой некоторую совокупность текстов (т. е. статических объектов), смысл же (семантика) этих текстов выражается процессами обработки данных и действиями пользователей, запускающих эти процессы (т. е. является динамическим). Это предопределяет выбор разработчиком ряда специфичных приемов, методов и средств.

Продукт разработки имеет и другую специфическую особенность: ПС при своем использовании (эксплуатации) не расходуется и не расходует используемых ресурсов.

Под жизненным циклом ПС (см. рис. 5.2) понимают весь период его разработки и эксплуатации (использования), начиная от момента возникновения замысла ПС и кончая прекращением всех видов его использования. Жизненный цикл охватывает довольно сложный процесс создания и использования ПС. Этот процесс может быть организован по-разному для разных классов ПС и в зависимости от особенностей коллектива разработчиков.

В настоящее время можно выделить 5 основных подходов к организации процесса создания и использования ПС.

Водопадный подход. При таком подходе разработка ПС состоит из цепочки этапов. На каждом этапе создаются документы, используемые на последующем этапе. В исходном документе фиксируются требования к ПС. В конце этой цепочки создаются программы, включаемые в ПС.

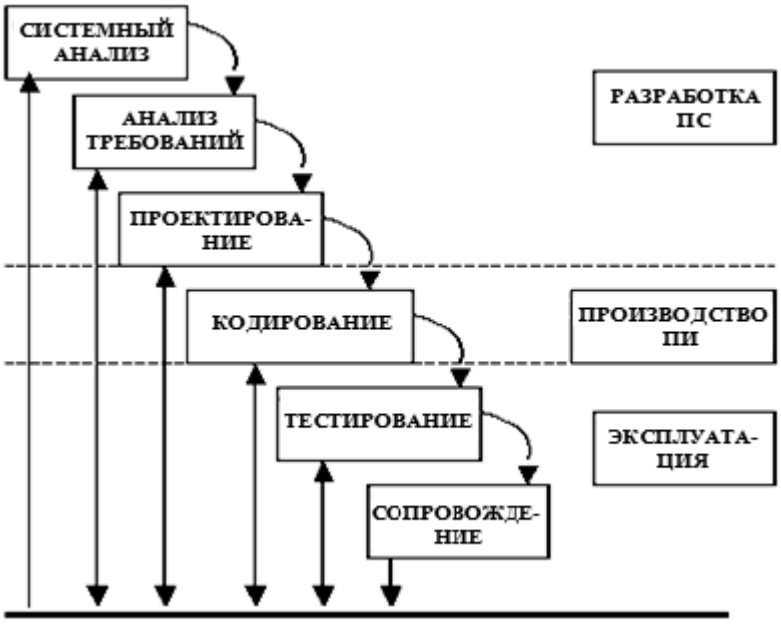


Рис. 5.2 – Стадии и фазы жизненного цикла ПС

Исследовательское программирование. Этот подход предполагает быструю (насколько это возможно) реализацию рабочих версий программ ПС, выполняющих лишь в первом приближении требуемые функции. После экспериментального применения реализованных программ производится их модификация с целью сделать их более полезными для пользователей. Этот процесс повторяется до тех пор, пока ПС не будет достаточно приемлемо для пользователей. Такой подход применялся на ранних этапах развития программирования, когда технологии программирования не придавали большого значения (использовалась интуитивная технология). В настоящее время этот подход применяется для разработки таких ПС, для которых пользователи не могут точно сформулировать требования (например, для разработки систем искусственного интеллекта).

Прототипирование. Этот подход моделирует начальную фазу исследовательского программирования вплоть до создания рабочих версий программ, предназначенных для проведения экспериментов с целью установить требования к ПС. В дальнейшем должна последовать разработка ПС

по установленным требованиям в рамках какого-либо другого подхода (например, водопадного).

Формальные преобразования. Этот подход включает разработку формальных спецификаций ПС и превращение их в программы путем корректных преобразований. На этом подходе базируется компьютерная технология (CASE-технология) разработки ПС.

Сборочное программирование. Этот подход предполагает, что ПС конструируется, главным образом, из компонент, которые уже существуют. Должно быть некоторое хранилище (библиотека) таких компонент, каждая из которых может многократно использоваться в разных ПС. Такие компоненты называются повторно используемыми. Процесс разработки ПС при данном подходе состоит скорее из сборки программ из компонент, чем из их программирования.

Обычно, на практике, рассматривается водопадный подход с некоторыми модификациями. Во-первых, потому, что в этом подходе приходится иметь дело с большинством процессов программной инженерии, а, во-вторых, потому, что в рамках этого подхода создается большинство больших программных систем. Именно этот подход рассматривается в качестве индустриального подхода разработки программного обеспечения.

Исследовательское программирование исходит из взгляда на программирование как на искусство. Оно применяется тогда, когда водопадный подход не применим из-за того, что не удается точно сформулировать требования к ПС. Прототипирование рассматривается как вспомогательный подход, используемый в рамках других подходов, в основном, для прояснения требований к ПС. Компьютерной технологии (включая обсуждение жизненного цикла ПС, созданного по этой технологии) будет посвящена отдельная лекция. Сборочное программирование мы в нашем курсе рассматривать не будем, хотя о повторно используемых программных модулях мы говорить будем, обсуждая свойства программных модулей.

В рамках водопадного подхода различают следующие стадии жизненного цикла ПС (см. рис. 5.2): разработку ПС, производство программных изделий (ПИ) и эксплуатацию ПС.

Стадия разработки ПС состоит из этапа его внешнего описания, этапа конструирования ПС, этапа кодирования (программирование в узком смысле) ПС и этапа аттестации ПС. Всем этим этапам сопутствуют процессы документирования и управления ПС. Этапы конструирования и кодирования часто перекрываются, иногда довольно сильно. Это означает, что кодирование некоторых частей программного средства может быть начато до завершения этапа конструирования.

Этап внешнего описания ПС включает процессы, приводящие к созданию некоторого документа, который мы будем называть внешним описанием ПС. Этот документ является описанием поведения ПС с точки зрения внешнего по отношению к нему наблюдателя с фиксацией требований относительно его качества. Внешнее описание ПС начинается с анализа и определения требований к ПС со стороны пользователей (заказчика), а также включает процессы спецификации этих требований. Конструирование ПС охватывает процессы: разработку архитектуры ПС, разработку структур программ ПС и их детальную спецификацию.

Кодирование ПС включает процессы создания текстов программ на языках программирования, их отладку с частичным тестированием ПС.

На этапе аттестации ПС производится оценка качества ПС. Если эта оценка оказывается приемлемой для практического использования ПС, то разработка ПС считается законченной. Это обычно оформляется в виде некоторого документа, фиксирующего решение комиссии, проводящей аттестацию ПС.

Программное изделие (ПИ) – экземпляр или копия разработанного ПС. Изготовление ПИ – это процесс генерации и/или воспроизведения (снятия копии) программ и программных документов ПС с целью их поставки пользователю для применения по назначению.

Производство ПИ – это совокупность работ по обеспечению изготовления требуемого количества ПИ в установленные сроки. Стадия производства ПИ в жизненном цикле ПС является, по существу, вырожденной (не существенной), так как представляет рутинную работу, которая может быть выполнена автоматически и без ошибок. Этим она принципиально отличается от стадии производства различной техники. В связи с этим в литературе эту стадию, как правило, не включают в жизненный цикл ПС.

Стадия эксплуатации ПС охватывает процессы хранения, внедрения и сопровождения ПС, а также транспортировки и применения ПИ по своему назначению. Она состоит из двух параллельно проходящих фаз: фазы применения ПС и фазы сопровождения ПС.

Применение ПС – это использование ПС для решения практических задач на компьютере путем выполнения его программ.

Сопровождение ПС – это процесс сбора информации о качестве ПС в эксплуатации, устранения обнаруженных в нем ошибок, его доработки и модификации, а также извещения пользователей о внесенных в него изменениях.

5.3 Техническая документация

В состав технической документации входят:

– техническая документация на автоматизированные системы;

– техническая документация на изделия, реализующие автоматизированные системы;

– техническая документация на программные изделия – программная документация.

Следует отметить, что разработка технической документации на автоматизированные системы в современных условиях неумолимо влечет за собой разработку технической документации на изделия и программные изделия, за исключением готовых продуктов сторонних фирм.

С другой стороны, все, что производится в современных условиях, проявляет все больше и больше признаков автоматизированных систем. Любая программа на стадии разработки является программной системой, например тот же Word. После установки, программа Word начинает автоматически выполнять некоторые функции: проверку орфографии, распознавания текста и др. функции, т. е. становится полноценной автоматизированной системой.

Техническая документация опирается на знание всевозможных стандартов. Опираясь на стандарты, можно отметить, что жизненный цикл технической документации включает в себя как минимум:

– процесс разработки технической документации;

– процесс публикации технической документации как на бумажных носителях, так и в электронном виде;

– процессы учета и хранения технической документации;

– процессы модификации, отслеживания изменений технической документации – сопровождения;

– процесс обмена технической документацией между подразделениями компании;

– процесс передачи технической документации Заказчику (конечному пользователю).

В настоящее время основополагающим документом при создании автоматизированных систем (АС), изделий и программных систем (ПС) является Техническое задание (на создание АС – ГОСТ 34.602–89). Техническое задание останется основополагающим документом и в обозримом будущем. Неважно, как будет называться этот документ – все равно это будет Техническое задание.

В ряде подразделов Технического задания приводится перечень нормативно-технической документации (НТД), на основании которой создается система.

Любая НТД, приведенная в указанном разделе согласованного и утвержденного Технического задания, обретает силу Закона как для Заказчика, так и для Исполнителя.

В целом для создания любого вида АС перечень НТД должен включать в себя:

– ГОСТ 34.601–90, регламентирующий стадии (и этапы) создания АС – описание процессов в их хронологическом порядке;

– ГОСТ 34.201–89, регламентирующий виды, комплектность и обозначения (наименования) документов, разрабатываемых на стадиях и этапах проведения работ по созданию АС;

– ГОСТ 34.689–90, регламентирующий требования к содержанию документов на АС;

– ряд ГОСТ–2 (ЕСКД), регламентирующих требования к содержанию и оформлению документов на изделия, входящие в состав АС;

– ГОСТ 2.601–95, регламентирующий требования к эксплуатационной документации на изделия, входящие в состав АС;

– ряд ГОСТ–19 (ЕСПД), регламентирующих требования к содержанию и оформлению документов на программные средства, входящие в состав АС;

– ГОСТы по качеству технической документации;

– ГОСТы (технические регламенты) предметной области.

5.3.1 Подготовка документации на программные средства в соответствии с ГОСТом

При подготовке документации на программные средства (ПС) необходимо использовать ГОСТы. При написании документации сначала надо ответить на основные вопросы при разработке программных средств:

- Что должно быть разработано кроме собственно программы?

- Что и как должно быть оформлено в виде документации?
- Что передавать пользователям?
- Что должно входить в само задание на программирование?
- Что должно входить в техническое задание на разработку ПС?

Кроме упомянутых выше вопросов, есть и масса других.

Основу отечественной нормативной базы в области документирования ПС составляет комплекс стандартов Единой системы программной документации (ЕСПД). Этот комплекс представляет собой систему межгосударственных стандартов стран СНГ (ГОСТ), действующих на территории Российской Федерации на основе межгосударственного соглашения по стандартизации. Стандарты ЕСПД в основном охватывают ту часть документации, которая создается в процессе разработки ПС, и связаны с документированием функциональных характеристик ПС. Следует отметить, что стандарты ЕСПД носят рекомендательный характер и становятся обязательными лишь при заключении контрактов (то есть при ссылке на них в договоре на разработку ПС).

Краткое представление стандартов ЕСПД. Стандарты ЕСПД могут с пользой применяться в практике документирования ПС по следующим моментам:

– стандарты ЕСПД вносят элемент упорядочения в процесс документирования ПС;

– предусмотренный стандартами ЕСПД состав программных документов не строго фиксированный и позволяет вносить в комплект документации на ПС новые виды документов;

– стандарты ЕСПД позволяют изменять структуры и содержание установленных видов программной документации (ПД) исходя из требований заказчика и пользователя.

Стандарты ЕСПД (как и ГОСТы) подразделяются на группы, приведенные в таблице 5.1.

Таблица 5.1 – Группы стандартов ЕСПД

Код группы	Наименование группы
1	Общие положения
2	Основополагающие стандарты
3	Правила выполнения документации разработки
4	Правила выполнения документации изготовления
5	Правила выполнения документации сопровождения
6	Правила выполнения эксплуатационной документации
7	Правила обращения программной документации
8	Резервные группы

Перечень документов ЕСПД

ГОСТ 19.001–77 ЕСПД. Общие положения.

ГОСТ 19.101–77 ЕСПД. Виды программ и программных документов.

ГОСТ 19.102–77 ЕСПД. Стадии разработки.

ГОСТ 19.103–77 ЕСПД. Обозначение программ и программных документов.

ГОСТ 19.104–78 ЕСПД. Основные надписи.

ГОСТ 19.105–78 ЕСПД. Общие требования к программным документам.

ГОСТ 19.106–78 ЕСПД. Требования к программным документам, выполненным печатным способом.

ГОСТ 19.201–78 ЕСПД. Техническое задание. Требования к содержанию и оформлению.

ГОСТ 19.202–78 ЕСПД. Спецификация. Требования к содержанию и оформлению.

ГОСТ 19.301–79 ЕСПД. Порядок и методика испытаний.

ГОСТ 19.401–78 ЕСПД. Текст программы. Требования к содержанию и оформлению.

ГОСТ 19.402–78 ЕСПД. Описание программы.

ГОСТ 19.404–79 ЕСПД. Пояснительная записка. Требования к содержанию и оформлению.

ГОСТ 19.501–78 ЕСПД. Формуляр. Требования к содержанию и оформлению.

ГОСТ 19.502–78 ЕСПД. Описание применения. Требования к содержанию и оформлению.

ГОСТ 19.503–79 ЕСПД. Руководство системного программиста. Требования к содержанию и оформлению.

ГОСТ 19.504–79 ЕСПД. Руководство программиста.

ГОСТ 19.505–79 ЕСПД. Руководство оператора.

ГОСТ 19.506–79 ЕСПД. Описание языка.

ГОСТ 19.508–79 ЕСПД. Руководство по техническому обслуживанию. Требования к содержанию и оформлению.

ГОСТ 19.604–78 ЕСПД. Правила внесения изменений в программные документы, выполняемые печатным способом.

ГОСТ 19.701–90 ЕСПД. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения.

ГОСТ 19.781–90. Обеспечение систем обработки информации программное.

Из всех стандартов ЕСПД остановимся только на тех, которые могут чаще использоваться на практике. Первым укажем стандарт, который можно использовать при формировании заданий на программирование. ГОСТ 19.201–78 (1626–79). ЕСПД. «Техническое задание. Требование к содержанию и оформлению». Техническое задание (ТЗ) содержит совокупность требований к ПС и может использоваться как критерий проверки и приемки разработанной программы. Поэтому достаточно полно составленное (с учетом возможности внесения дополнительных разделов) и принятое заказчиком и разработчиком, ТЗ является одним из основополагающих документов проекта ПС.

Техническое задание, согласно ГОСТ, 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;
- порядок контроля и приемки;
- в техническое задание допускается включать приложения.

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

Следующий стандарт ГОСТ (СТ СЭВ) 19.101–77 (1626–79). ЕСПД. «Виды программ и программных документов» устанавливает виды программ и программных документов для вычислительных машин, комплексов и систем независимо от их назначения и области применения.

Выделяют виды программ (табл. 5.2), виды программных документов (табл. 5.3), виды эксплуатационных документов (табл. 5.4).

Таблица 5.2 – Виды программ

Вид программы	Определение
Компонент	Программа, рассматриваемая как единое целое, выполняющая законченную функцию и применяемая самостоятельно или в составе комплекса
Комплекс	Программа, состоящая из двух или более компонентов и (или) комплексов, выполняющих взаимосвязанные функции, и применяемая самостоятельно или в составе другого комплекса

Таблица 5.3 – Виды программных документов

Вид программного документа	Содержание программного документа
Спецификация	Состав программы и документации на нее

Окончание табл. 5.3

Вид программного документа	Содержание программного документа
Ведомость держателей подлинников	Перечень предприятий, на которых хранят подлинники программных документов
Текст программы	Запись программы с необходимыми комментариями
Описание программы	Сведения о логической структуре и функционировании программы
Программа и методика испытаний	Требования, подлежащие проверке при испытании программы, а также порядок и методы их контроля
Техническое задание	Назначение и область применения программы, технические, технико-экономические и специальные требования, предъявляемые к программе, необходимые стадии и сроки разработки, виды испытаний
Пояснительная записка	Схема алгоритма, общее описание алгоритма и (или) функционирования программы, а также обоснование принятых технических и технико-экономических решений
Эксплуатационные документы	Сведения для обеспечения функционирования и эксплуатации программы

Таблица 5.4 – Виды эксплуатационных документов

Вид эксплуатационного документа	Содержание эксплуатационного документа
Ведомость эксплуатационных документов	Перечень эксплуатационных документов на программу
Формуляр	Основные характеристики программы, комплектность и сведения об эксплуатации программы
Описание применения	Сведения о назначении программы, области применения, применяемых методах, классе решаемых задач, ограничениях для применения, минимальной конфигурации технических средств
Руководство системного программиста	Сведения для проверки, обеспечения функционирования и настройки программы на условия конкретного применения
Руководство программиста	Сведения для эксплуатации программы
Руководство оператора	Сведения для обеспечения процедуры общения оператора с вычислительной системой в процессе выполнения программы

Окончание табл. 5.4

Вид эксплуатационного документа	Содержание эксплуатационного документа
Описание языка	Описание синтаксиса и семантики языка
Руководство по техническому обслуживанию	Сведения для применения тестовых и диагностических программ при обслуживании технических средств

В зависимости от способа выполнения и характера применения программные документы подразделяются на подлинник, дубликат и копию (ГОСТ 2.102–68), предназначенные для разработки, сопровождения и эксплуатации программы (табл. 5.5).

Таблица 5.5 – Виды программных документов, разрабатываемых на разных стадиях, и их коды

Код вида документа	Вид документа	Стадии разработки			
		Эскизный проект	Технический проект	Рабочий проект	
				компонент	комплекс
–	Спецификация	–	–	!	+
05	Ведомость держателей подлинников	–	–	–	?
12	Текст программы	–	–	+	?
13	Описание программы	–	–	?	?
20	Ведомость эксплуатационных документов	–	–	?	?
30	Формуляр	–	–	?	?
31	Описание применения	–	–	?	?
32	Руководство системного программиста	–	–	?	?
33	Руководство программиста	–	–	?	?

Окончание табл. 5.5

Код вида документа	Вид документа	Стадии разработки			
		Эскизный проект	Технический проект	Рабочий проект	
				компонент	комплекс
34	Руководство оператора	–	–	?	?
35	Описание языка	–	–	?	?
46	Руководство по техническому обслуживанию	–	–	?	?
51	Программа и методика испытаний	–	–	?	?
81	Пояснительная записка	?	?	–	–
90–99	Прочие документы	?	?	?	?

Условные обозначения:

+ – документ обязательный;

! – документ обязательный для компонентов, имеющих самостоятельное применение;

? – необходимость составления документа определяется на этапе разработки и утверждения технического задания;

– – документ не составляют.

Допускается объединять отдельные виды эксплуатационных документов (за исключением ведомости эксплуатационных документов и формуляра). Необходимость объединения этих документов указывается в техническом задании. Объединенному документу присваивают наименование и обозначение одного из объединяемых документов. В объединенных документах должны быть приведены сведения, которые необходимо включать в каждый объединяемый документ.

Рассмотрим ГОСТ 19.102–77. ЕСПД. «Стадии разработки». Этот документ устанавливает стадии разработки программ и программной документации для вычислительных машин, комплексов и систем независимо от их назначения и области применения (табл. 5.6).

Таблица 5.6 – Стадии разработки, этапы и содержание работ

Стадии разработки	Этапы работ	Содержание работ
Техническое задание	Обоснование необходимости разработки программы	Постановка задачи. Сбор исходных материалов. Выбор и обоснование критериев эффективности и качества разрабатываемой программы. Обоснование необходимости проведения научно-исследовательских работ
	Научно-исследовательские работы	Определение структуры входных и выходных данных. Предварительный выбор методов решения задач. Обоснование целесообразности применения ранее разработанных программ. Определение требований к техническим средствам. Обоснование принципиальной возможности решения поставленной задачи
	Разработка и утверждение технического задания	Определение требований к программе. Разработка технико-экономического обоснования разработки программы. Определение стадий, этапов и сроков разработки программы и документации на нее. Выбор языков программирования. Определение необходимости проведения научно-исследовательских работ на последующих стадиях. Согласование и утверждение технического задания
Эскизный проект	Разработка эскизного проекта	Предварительная разработка структуры входных и выходных данных. Уточнение методов решения задачи. Разработка общего описания алгоритма решения задачи. Разработка технико-экономического обоснования

Окончание табл. 5.6

Стадии разработки	Этапы работ	Содержание работ
	Утверждение эскизного проекта	Разработка пояснительной записки. Согласование и утверждение эскизного проекта
Технический проект	Разработка технического проекта	Уточнение структуры входных и выходных данных. Разработка алгоритма решения задачи. Определение формы представления входных и выходных данных. Определение семантики и синтаксиса языка. Разработка структуры программы. Окончательное определение конфигурации технических средств
	Утверждение технического проекта	Разработка плана мероприятий по разработке и внедрению программ. Разработка пояснительной записки. Согласование и утверждение технического проекта
Рабочий проект	Разработка программы	Программирование и отладка программы
	Разработка программной документации	Разработка программных документов в соответствии с требованиями ГОСТ 19.101–77
	Испытания программы	Разработка, согласование и утверждение программы и методики испытаний. Проведение предварительных государственных, межведомственных, приемо-сдаточных и других видов испытаний. Корректировка программы и программной документации по результатам испытаний
Внедрение	Подготовка и передача программы	Подготовка и передача программы и программной документации для сопровождения и (или) изготовления. Оформление и утверждение акта о передаче программы на сопровождение и (или) изготовление. Передача программы в фонд алгоритмов и программ

ГОСТ 19.103–77 ЕСПД. «Обозначение программ и программных документов» устанавливает для программ структуру обозначения, которая включает: Код страны-разработчика и код организации-разработчика присваивают в установленном порядке. Регистрационный номер в порядке возрастания, начиная с 00001 до 99999, для каждой организации-разработчика. Номер издания программы или номер редакции. Номер документа данного вида, номер части документа присваиваются в порядке возрастания с 01 до 99. (Если документ состоит из одной части, то дефис и порядковый номер части не указывают.) Номер редакции спецификации и ведомости эксплуатационных документов на программу должны совпадать с номером издания этой же программы.

ГОСТ 19.105–78 ЕСПД. «Общие требования к программным документам» устанавливает общие требования к оформлению программных документов для вычислительных машин, комплексов и систем, независимо от их назначения и области применения и предусмотренных стандартами ЕСПД для любого способа выполнения документов на различных носителях данных.

Программный документ может быть представлен на различных типах носителей данных и состоит из следующих условных частей:

– титульной;

– информационной;

– основной.

Правила оформления документа и его частей на каждом носителе данных устанавливаются стандартами ЕСПД на правила оформления документов на соответствующих носителях данных. Согласно ГОСТ 19.106–78 ЕСПД. «Требования к программным документам, выполненным печатным способом» программные документы оформляют:

– на листах формата А4 (ГОСТ 2.301–68) при изготовлении документа машинописным или рукописным способом;

– допускается оформление на листах формата А3;

– при машинном способе выполнения документа допускаются отклонения размеров листов, соответствующих форматам А4 и А3, определяемые возможностями применяемых технических средств; на листах форматов А4 и А3, предусматриваемых выходными характеристиками устройств вывода данных, при изготовлении документа машинным способом;

– на листах типографических форматов при изготовлении документа типографским способом.

Расположение материалов программного документа осуществляется в следующей последовательности:

– титульная часть: лист утверждения (не входит в общее количество листов документа) и титульный лист (первый лист документа);

– информационная часть: аннотация и лист содержания;

– основная часть: текст документа (с рисунками, таблицами и т. п.), перечень терминов и их определений, перечень сокращений, приложения, предметный указатель и перечень ссылочных документов;

– часть регистрации изменений: лист регистрации изменений.

Перечень терминов и их определений, перечень сокращений, приложения, предметных указатель, перечень ссылочных документов выполняются при необходимости.

Стандарт ГОСТ 19.402–78 ЕСПД. «Описание программы» ориентирован на документирование результирующего продукта разработки. Состав документа в своей содержательной части может дополняться разделами и пунктами, взятыми из стандартов для других описательных документов и руководств: ГОСТ 19.404–79 ЕСПД. «Пояснительная записка», ГОСТ 19.502–78 ЕСПД. «Описание применения», ГОСТ 19.503–79 ЕСПД. «Руководство системного программиста», ГОСТ 19.504–79 ЕСПД. «Руководство программиста», ГОСТ 19.505–79 ЕСПД. «Руководство оператора».

Среди всех документов можно выделить документы: ГОСТ 19.301–79 ЕСПД. «Программа и методика испытаний», который может использоваться

для разработки документов планирования и проведения испытательных работ по оценке готовности и качества ПС. Стандарт ГОСТ 19.701–90 ЕСПД. «Схемы алгоритмов, программ, данных и систем. Обозначения условные графические и правила выполнения» устанавливает правила выполнения схем, используемых для отображения различных видов задач обработки данных и средств их решения. ГОСТ 19781–90 «Обеспечение систем обработки информации программное. Термины и определения» устанавливает термины и определения понятий в области программного обеспечения (ПО) систем обработки данных (СОД), применяемые во всех видах документации и литературы, входящих в сферу работ по стандартизации или использующих результаты этих работ. Документ ГОСТ 28388–89 «Системы обработки информации. Документы на магнитных носителях данных. Порядок выполнения и обращения» применим не только к программным, но и конструкторским, технологическим и другим проектным документам, выполняемые на магнитных носителях.

5.3.2 Стандарты комплекса ГОСТ 34

ГОСТ 34 рассчитан на взаимодействие заказчика и разработчика и в основном уделяет внимание содержанию проектных документов, распределению действий между сторонами (табл. 5.7).

Таблица 5.7 – Стадии и этапы разработки автоматизированных систем (АС) согласно ГОСТ 34

1. ФТ – Формирование требований к АС	1.1. Обследование объекта и обоснование необходимости создания АС; 1.2. Формирование требований пользователя к АС; 1.3. Оформление отчета о выполненной работе и заявки на разработку АС (тактико-технического задания)

Окончание табл. 5.7

2. РК – Разработка концепции АС	2.1. Изучение объекта; 2.2. Проведение необходимых научно-исследовательских работ; 2.3. Разработка вариантов концепции АС, удовлетворяющей требованиям пользователя 2.4. Оформление отчета о выполненной работе
3. ТЗ – Техническое создание АС	3.1. Разработка и утверждение технического задания на задание
4. ЭП – Эскизный проект	4.1. Разработка предварительных проектных решений по системе и ее частям; 4.2. Разработка документации на АС и ее части
5. ТП – Технический проект	5.1. Разработка проектных решений по системе и ее частям; 5.2. Разработка документации на АС и ее части; 5.3. Разработка и оформление документации на поставку изделий для комплектования АС и/или технических требований (технических заданий) на их разработку; 5.4. Разработка заданий на проектирование в смежных частях проекта объекта автоматизации
6. РД – Рабочая документация	6.1. Разработка рабочей документации на систему и ее части; 6.2. Разработка или адаптация программ
7. ВД – Ввод в действие	7.1. Подготовка объекта автоматизации к вводу АС в действие; 7.2. Подготовка персонала; 7.3. Комплектация АС поставляемыми изделиями (программными и техническими средствами, программно-техническими комплексами, информационными изделиями); 7.4. Строительно-монтажные работы; 7.5. Пуско-наладочные работы; 7.6. Проведение предварительных испытаний; 7.7. Проведение опытной эксплуатации; 7.8. Проведение приемочных испытаний
8. Сп – Сопровождение АС	8.1. Выполнение работ в соответствии с гарантийными обязательствами; 8.2. Послегарантийное обслуживание

3. Международный стандарт ISO/IEC 12207: 1995-08-01

ISO12207 – базовый стандарт процессов жизненного цикла программного обеспечения (ЖЦ ПО), ориентирован на различные виды ПО и типы проектов АС, куда ПО входит как часть. Стандарт определяет стратегию и общий порядок в создании и эксплуатации ПО, он охватывает ЖЦ ПО от концептуализации идей до завершения ЖЦ.

Определения стандарта:

1. Система – это объединение одного или более процессов, аппаратных средств, программного обеспечения, оборудования и людей для обеспечения возможности удовлетворения определенных потребностей или целей.

2. Модель жизненного цикла – структура, содержащая процессы, действия и задачи, которые осуществляются в ходе разработки, функционирования и сопровождения программного продукта в течение всей жизни системы, от определения требований до завершения ее использования. Множество процессов и задач сконструировано так, что возможна их адаптация в соответствии с проектами ПО. Процесс адаптации является процессом исключения процессов, видов деятельности и задач, не применимых в конкретном проекте. Степень адаптивности: максимальная.

3. Требование квалификации – набор критериев или условий (квалификационные требования), которые должны быть удовлетворены для того, чтобы квалифицировать программный продукт как подчиняющийся (удовлетворяющий условиям) его спецификациям и готовый для использования в целевой окружающей среде.

Стандарт не предписывает конкретную модель ЖЦ или метод разработки ПО, но определяет, что стороны-участники использования стандарта ответственны за выбор модели ЖЦ для проекта ПО, за адаптацию процессов и задач стандарта к этой модели, за выбор и применение методов разработки ПО, за выполнение действий и задач, подходящих для проекта ПО.

Каждый процесс ЖЦ разделен на набор действий, каждое действие – на набор задач. Очень важное отличие ISO: каждый процесс, действие или

задача инициируется и выполняется другим процессом по мере необходимости, причем нет заранее определенных последовательностей (естественно, при сохранении логики связей, по исходным сведениям, задач и т. п.).

В стандарте ISO12207 описаны:

1. Пять основных процессов ЖЦ ПО:

Процесс приобретения. Определяет действия предприятия-покупателя, которое приобретает АС, программный продукт или сервис ПО.

Процесс поставки. Определяет действия предприятия-поставщика, которое снабжает покупателя системой, программным продуктом или сервисом ПО.

Процесс разработки. Определяет действия предприятия-разработчика, которое разрабатывает принцип построения программного изделия и программный продукт.

Процесс функционирования. Определяет действия предприятия-оператора, которое обеспечивает обслуживание системы (а не только ПО) в процессе ее функционирования в интересах пользователей. В отличие от действий, которые определяются разработчиком в инструкциях по эксплуатации (эта деятельность разработчика предусмотрена во всех трех рассматриваемых стандартах), определяются действия оператора по консультированию пользователей, получению обратной связи и др., которые он планирует сам и берет на себя соответствующе обязанности.

Процесс сопровождения. Определяет действия персонала сопровождения, который обеспечивает сопровождение программного продукта, что представляет собой управление модификациями программного продукта, поддержку его текущего состояния и функциональной пригодности, включает в себя инсталляцию и удаление программного изделия на вычислительной системе.

2. Восемь вспомогательных процессов, которые поддерживают реализацию другого процесса, будучи неотъемлемой частью всего ЖЦ программного изделия, и обеспечивают должное качество проекта ПО:

- Решения проблем.

- Документирования.
- Управления конфигурацией.
- Гарантирования качества, который использует результаты остальных процессов группы обеспечения качества, в которую входят:

– процесс верификации;

– процесс аттестации;

– процесс совместной оценки;

– процесс аудита.

3. Четыре организационных процесса:

– процесс управления;

– процесс создания инфраструктуры;

– процесс усовершенствования;

– процесс обучения.

К ним примыкает особый Процесс адаптации, который определяет основные действия, необходимые для адаптации стандарта к условиям конкретного проекта. Под процессом усовершенствования здесь понимается не усовершенствование АС или ПО, а улучшение самих процессов приобретения, разработки, гарантирования качества и т. п., реально осуществляемых в организации.

Такой характер позволяет реализовывать любую модель ЖЦ. При выполнении анализа требований к ПО предусмотрено 11 классов характеристик качества, которые используются позже при гарантировании качества. При этом разработчик должен установить и документировать как требования к программному обеспечению:

– функциональные и возможные спецификации, включая исполнение, физические характеристики и условия среды эксплуатации, при которых единица программного обеспечения должна быть выполнена;

– внешние связи (интерфейсы) с единицей программного обеспечения;

– требования квалификации;

– спецификации надежности, включая спецификации, связанные с методами функционирования и сопровождения, воздействия окружающей среды и вероятностью травмы персонала;

– спецификации защищенности;

– человеческие факторы спецификаций по инженерной психологии (эргономике), включая связанные с ручным управлением, взаимодействием человека и оборудования, ограничениями на персонал и областями, нуждающимися в концентрированном человеческом внимании, которые являются чувствительными к ошибкам человека и обучению;

– определение данных и требований базы данных;

– установочные и приемочные требования поставляемого программного продукта в местах функционирования и сопровождения (эксплуатации);

– документация пользователя;

– работа пользователя и требования выполнения;

– требования сервиса пользователя.

6 УНИВЕРСАЛЬНЫЙ ЯЗЫК МОДЕЛИРОВАНИЯ UML

6.1 Общие сведения об UML

Унифицированный язык моделирования UML (Unified Modeling Language) – это язык для определения, представления, проектирования и документирования программных систем, организационно-экономических систем, технических систем и других систем различной природы. UML содержит стандартный набор диаграмм и нотаций самых разнообразных видов.

UML является наследником методов объектно-ориентированного анализа и проектирования, появившихся в конце 1980-х и начале 1990-х гг. Создание UML началось в конце 1994 г., с объединения методов Booch и OMT (Object Modeling Technique) под эгидой компании Rational Software. К концу 1995 г. Гради Буч и Джеймс Рамбо создали первую спецификацию Unified Method. Тогда же в 1995 г. к ним присоединился создатель метода OOSE (Object-Oriented Software Engineering) Ивар Якобсон. UML является унификацией методов Буча, Рамбо и Якобсона. Разработка UML преследовала следующие цели:

– предоставить разработчикам единый язык визуального моделирования;

– предусмотреть механизмы расширения и специализации языка;

– обеспечить независимость языка от языков программирования и процессов разработки;

– интегрировать накопленный практический опыт.

UML широко используется в индустрии ПО. Практически все мировые производители CASE-средств поддерживают UML в своих продуктах. В 2004 г. Object Management Group приняла UML версии 2.0. Ранее в 1997 г. OMG приняла стандарт UML 1.1.

Состав диаграмм UML:

- *структурные:*

– диаграммы классов, моделирующие статическую структуру классов системы и связи между классами;

– диаграммы компонентов, моделирующие иерархии компонентов ПО;

– диаграммы размещения, моделирующие физическую архитектуру системы;

- *поведенческие:*

– диаграммы вариантов использования, моделирующие бизнес-процессы и требования к ПО;

– диаграммы взаимодействия (диаграммы последовательности и коммуникационные диаграммы), моделирующие обмен сообщениями между объектами;

– диаграммы состояний, моделирующие поведение объектов;

– диаграммы деятельности, моделирующие поведение системы в целом и потоки управления.

В UML 2.0 введены новые типы диаграмм, которых ранее не было: диаграммы обзора взаимодействия, временные диаграммы и диаграммы составных структур.

Вариант использования – это ответные действия ПО, являющиеся реакцией на событие, инициируемое извне. Вариант использования описывает типичное взаимодействие между пользователем и ПО. Он отражает представление о поведении системы с точки зрения пользователя. На диаграммах варианты использования представляются в виде овалов.

Действующее лицо – это роль, которую пользователь играет по отношению к системе. На диаграммах вариантов использования они изображаются в виде стилизованных человеческих фигурок. Действующим лицом может быть пользователь-человек, внешняя программная система или время, если запуск каких-либо событий в системе зависит от времени.

Диаграммы вариантов использования показывают, какие действующие лица инициируют варианты использования (от них идет стрелка к варианту использования). Из диаграмм понятно, какие действующие лица получают данные в ходе выполнения варианта использования (к ним идет стрелка от варианта использования).

Правила построения диаграмм вариантов использования:

1. Не следует моделировать связи между действующими лицами, поскольку это не относится к системе.

2. Не следует соединять стрелкой два варианта использования.

3. Каждый вариант использования должен быть инициирован действующим лицом.

Диаграмма вариантов использования является самым общим представлением функциональных требований к системе. Детально функциональные требования описываются в документе, называемом «сценарий варианта использования» или «поток событий». Он подробно документирует процесс взаимодействия действующего лица с системой, реализуемый в рамках варианта использования.

Описание потока событий включает следующие разделы:

1) краткое описание;
2) предусловия;
3) основной поток событий;
4) альтернативные потоки событий;
5) постусловия;
6) расширения и подчиненные потоки.

Хорошо написанный поток событий должен легко читаться и состоять из предложений, написанных в единой грамматической форме. Правила составления описания потока событий:

– следует использовать простые предложения;

– нужно явно указывать в каждом пункте, кто выполняет действие – действующее лицо или система;

– не следует включать в поток событий слишком незначительные действия;

– в описании основного потока не следует рассматривать ошибочные ситуации;

– некорректные действия пользователя и внутренние ошибки следует описывать в альтернативных потоках.

В диаграммах вариантов использования может присутствовать несколько типов связей:

– связи коммуникации (линия со стрелкой, обозначающая связь между вариантом использования и действующим лицом);

– связи включения (пунктирная линия со стрелкой, обозначающая включение многократно используемой функциональности, представленной в виде абстрактного варианта использования);

– связи расширения (пунктирная линия со стрелкой, указывающая на особый случай, описанный в абстрактном варианте использования);

– связи обобщения (линия с треугольным концом, показывающая, что у нескольких действующих лиц имеются общие черты и различия).

Диаграммы взаимодействия описывают поведение взаимодействующих групп объектов в рамках потока событий. На диаграмме отображается ряд объектов и сообщения, которыми они обмениваются между собой. Сообщение – это средство, с помощью которого объект-отправитель запрашивает у объекта-получателя выполнение одной из его операций. Существует два вида диаграмм взаимодействия: диаграммы последовательности и коммуникационные диаграммы (ранее называемые кооперативными).

Диаграммы последовательности отражают временную последовательность событий, происходящих в рамках варианта использования. Все

действующие лица и объекты системы изображаются в верхней части диаграммы. От каждого из них вниз проведена вертикальная черта – «линия жизни». Стрелки, соответствующие сообщениям, которые передаются между действующим лицом и объектом или между объектами, соединяют линии жизни отправителя и получателя сообщения. Порядок отправки сообщений соответствует их размещению на диаграмме сверху вниз.

Вторым видом диаграмм взаимодействия являются коммуникационные диаграммы. Как и диаграммы последовательности, они отображают поток событий варианта использования. На коммуникационных диаграммах внимание сконцентрировано на связях между объектами. Из них легче понять связи между объектами, однако труднее уяснить последовательность событий. Объекты и/или действующие лица, обменивающиеся сообщениями, соединяются линиями, над которым в виде стрелок обозначаются сообщения. Нумерация сообщений указывает их последовательность во времени.

Диаграмма классов определяет классы системы и различного рода статические связи, которые существуют между ними. На диаграммах классов изображаются также атрибуты классов, операции классов и ограничения, которые накладываются на связи между классами. Классы изображаются в виде прямоугольников, ассоциации – в виде линий со стрелками, агрегации и композиции – в виде линий с ромбом на конце, связь обобщения – в виде линии с треугольником на конце, зависимость – в виде пунктирной линии со стрелкой. Ответственность классов изображается на диаграммах с помощью стереотипов. Класс может быть помечен как граничный (boundary), если он отвечает за взаимодействие с пользователем или внешней системой. Класс-контроллер реализует бизнес-логику приложения. Класс-сущность отвечает за представление данных.

Для группировки классов, обладающих некоторой общностью, применяются пакеты. Пакет – общий механизм для организации элементов модели

в группы. Каждый пакет – это группа элементов модели, иногда сопровождаемая диаграммами, поясняющими структуру группы. Каждый элемент модели может входить только в один пакет. Диаграммы пакетов отображают зависимости между пакетами, возникающие, если элемент одного пакета зависит от элемента другого.

Пакеты также используются для представления подсистем. Подсистема – это комбинация пакета (поскольку она включает некоторое множество классов) и класса (поскольку она обладает поведением, т. е. реализует набор операций, которые определены в ее интерфейсах). Связь между подсистемой и интерфейсом называется связью реализации.

Диаграммы состояний определяют все возможные состояния, в которых может находиться конкретный объект, а также процесс смены состояний объекта в результате наступления некоторых событий. На каждой диаграмме состояний имеется одно начальное состояние и одно (или более одного) финальное. Начальное состояние выделено черной точкой, оно соответствует состоянию объекта, когда он только что был создан. Финальное состояние обозначается черной точкой в белом кружке, оно соответствует состоянию объекта непосредственно перед его уничтожением. Когда объект находится в каком-то конкретном состоянии, могут выполняться различные процессы. Они называются деятельностями и указываются на диаграмме. Деятельность – это прерываемое поведение. Оно может выполняться до своего завершения, пока объект находится в данном состоянии, или может быть прервано переходом объекта в другое состояние. Деятельность изображают внутри самого состояния, ей должно предшествовать слово do и двоеточие. Для состояния могут быть указаны входное и выходное действия. Входное действие выполняется, когда объект переходит в данное состояние как часть этого перехода. В отличие от деятельности входное действие рассматривается как непрерываемое. Входное действие также показывают внутри состояния,

ему предшествует слово entry и двоеточие. Выходное действие осуществляется как составная часть выхода из данного состояния. Оно также является непрерываемым. Его изображают внутри состояния, ему предшествует слово exit и двоеточие.

Переходом (transition) называется перемещение объекта из одного состояния в другое. На диаграмме все переходы изображают в виде линий со стрелками. Объект может перейти в то же состояние, в котором он в настоящий момент находится. С переходом можно связать событие, ограждающее условие и действие. Событие вызывает переход из одного состояния в другое. Событие размещают на диаграмме вдоль линии перехода. Ограничивающие условия определяют, когда переход может произойти, а когда нет. Их изображают на диаграмме вдоль линии перехода после имени события, заключая в квадратные скобки. Действие, являющееся частью перехода, изображают вдоль линии перехода после имени события, ему предшествует косая черта. Для некоторых состояний указывают вложенные подсостояния. В них может быть указано так называемое историческое состояние, переход в которое означает возврат к предыдущему активному подсостоянию.

Диаграммы состояний создают лишь для классов, экземпляры которых имеют сложное поведение.

Диаграммы деятельности полезны в описании поведения, включающего большое количество параллельных процессов. Также их можно применять для представления потоков событий вариантов использования в наглядной графической форме.

Основным элементом диаграмм деятельности является деятельность. Интерпретация этого термина зависит от той точки зрения, с которой строится диаграмма (это может быть некоторая задача, которую необходимо выполнить вручную или автоматизированным способом, или операция класса). Деятельность изображается в виде закругленного прямоугольника

с текстовым описанием. Любая диаграмма деятельности должна иметь начальную точку, определяющую начало потока событий. Конечная точка необязательна. На диаграмме может быть несколько конечных точек, но только одна начальная. На диаграмме могут присутствовать объекты и потоки объектов, в тех случаях, если объект используется или изменяется в одной из деятельностей. Поток объектов отмечается пунктирной стрелкой от деятельности к изменяемому объекту или от объекта к деятельности, использующей объект. Переходы (стрелки) показывают, как поток управления переходит от одной деятельности к другой. Если для перехода определено событие, то переход выполняется только после наступления такого события. Ограничивающие условия определяют, когда переход может произойти, а когда нет. Ветвление, а также слияние потоков управления изображается ромбом. Если необходимо показать, что две или более ветвей потока выполняются параллельно, используются линейки синхронизации (жирные линии).

Диаграммы деятельности следует использовать в следующих ситуациях:

1) при анализе потоков событий в конкретном варианте использования;

2) при анализе потоков событий во взаимодействующих вариантах использования (в этом случае диаграмма с помощью вертикальных пунктирных линий – «плавательных дорожек» разделяется на зоны, где показываются потоки событий одного из вариантов использования, а связь между разными потоками изображается в виде переходов или потоков объектов).

Диаграммы компонентов моделируют физический уровень системы. На них изображаются компоненты ПО и связи между ними. На такой диаграмме обычно выделяют два типа компонентов: исполняемые компоненты и библиотеки кода. Каждый класс модели (или подсистема) преобразуется в компонент исходного кода. Между отдельными компонентами изображают зависимости, соответствующие зависимостям на этапе компиляции

или выполнения программы. В модели системы может быть несколько диаграмм компонентов, в зависимости от числа подсистем или исполняемых файлов. Каждая подсистема является пакетом компонентов. Диаграммы компонентов применяются теми участниками проекта, кто отвечает за компиляцию и сборку системы.

Диаграмма размещения отражает физические взаимосвязи между программными и аппаратными компонентами системы. Она является хорошим средством для того, чтобы показать размещение объектов и компонентов в распределенной системе. Ее основные элементы:

– узел (node) – вычислительный ресурс – процессор или другое устройство (дисковая память, контроллеры различных устройств и т. д.);

– соединение (connection) – канал взаимодействия узлов.

Для узла можно задать выполняющиеся на нем процессы.

Диаграмма размещения используется менеджером проекта, пользователями, архитектором системы и эксплуатационным персоналом, чтобы понять физическое размещение системы и расположение ее отдельных подсистем.

Механизмы расширения UML предназначены для того, чтобы разработчики могли адаптировать язык моделирования к своим конкретным нуждам, не меняя при этом его метамодель. Наличие механизмов расширения принципиально отличает UML от других средств моделирования. К механизмам расширения UML относятся:

– стереотипы;

– тегированные (именованные) значения;

– примечания;

– ограничения.

Стереотип – это новый тип элемента модели, который определяется на основе уже существующего элемента. Стереотипы расширяют нотацию модели, могут применяться к любым элементам модели и представляются

в виде текстовой метки или пиктограммы. Стереотипы классов – это механизм, позволяющий разделять классы на категории. Например, основными стереотипами, используемыми в процессе анализа системы, являются: Boundary (граничный класс), Entity (класс-сущность) и Control (управляющий класс).

Помимо упомянутых стереотипов, разработчики ПО могут создавать свои собственные наборы стереотипов, формируя тем самым специализированные подмножества UML (например, для описания бизнес-процессов, веб-приложений, баз данных и т. д.). Такие подмножества (наборы стереотипов) в стандарте языка UML носят название профилей языка.

Именованное значение – это пара строк «тег = значение», или «имя = содержимое», в которых хранится дополнительная информация о каком-либо элементе системы, например, время создания, статус разработки или тестирования, время окончания работы над ним и т. п. Примечание – элемент диаграммы для комментария или другой текстовой информации. Примечание может содержать дополнительные сведения об элементах модели (с ними его соединяет пунктирная линия).

Ограничение – это семантическое ограничение, имеющее вид текстового выражения на естественном или формальном языке (OCL – Object Constraint Language), которое невозможно выразить с помощью нотации UML. Средства OCL не предназначены для описания процессов вычисления выражений, а только лишь фиксируют необходимость выполнения тех или иных условий применительно к отдельным компонентам моделей. Он может быть использован для решения следующих задач:

- описание инвариантов классов и типов в модели классов;
- описание пред- и постусловий в операциях и методах;
- описание ограничивающих условий элементов модели;
- навигация по структуре модели;
- спецификация ограничений на операции.

Рассмотрим пример: бронирование билета.

На рынок вышла новая авиакомпания «Avia». Менеджеры компании решили заказать у вашей фирмы разработку системы бронирования билетов. При заказе фирма поставила ряд условий, которые обязательно должны быть выполнены. В первой версии системы они хотят видеть две части. Работа первой части системы связана с занесением информации. Вторая часть системы предназначена для общения с клиентами.

При формулировании требований менеджеры упомянули, что рейсы спланированы так, что до пункта назначения можно долететь с пересадками. Одно из требований заключалось в том, чтобы система помогала покупать билеты в зависимости от пожеланий пользователя. Задача является математической. Система должна уметь решать однокритериальную задачу поиска кратчайших путей на графах. Критерий – цена. Система распределенная: так как в каждом аэропорте своя база направлений полетов самолетов, то знают о рейсе только аэропорты-соседи по рейсам. Объекты системы: распределенное хранилище рейсов, покупатель билетов, менеджер рейсов. Распределенное хранилище рейсов: название рейсов, номера и стоимость билетов. Покупатель: ФИО, сумма. Покупатель задает параметры, связанные с суммой, которую он хочет потратить. Система должна подобрать оптимальный маршрут.

При отсутствии прямых маршрутов система должна попробовать найти маршруты с пересадками. Если таковых не находится, система должна сказать, что с такими ограничениями нельзя добраться до места назначения. Среди причин: Отсутствие рейсов в желаемом направлении даже с учетом пересадок. Нехватка денег. В ответ, пользователь должен иметь возможность поменять параметры с учетом предыстории. Менеджер рейсов: должен иметь следующие возможности: создания и удаления аэропортов в системе. создания и удаления рейсов в аэропортах.

6.2 Краткое описание UML

Язык UML имеет сложную иерархическую структуру, показанную на рисунке 6.1, где первый иерархический уровень языка UML составляют сущности, отношения между сущностями и наглядные диаграммы. Далее на следующем уровне показано, что понятие «структурные сущности» является именем семи видов пиктограмм (выразительных рисунков). Поведенческие сущности делятся на два вида диаграмм. Группирующие и аннотационные сущности имеют один вид пиктограмм, называемых примечаниями.

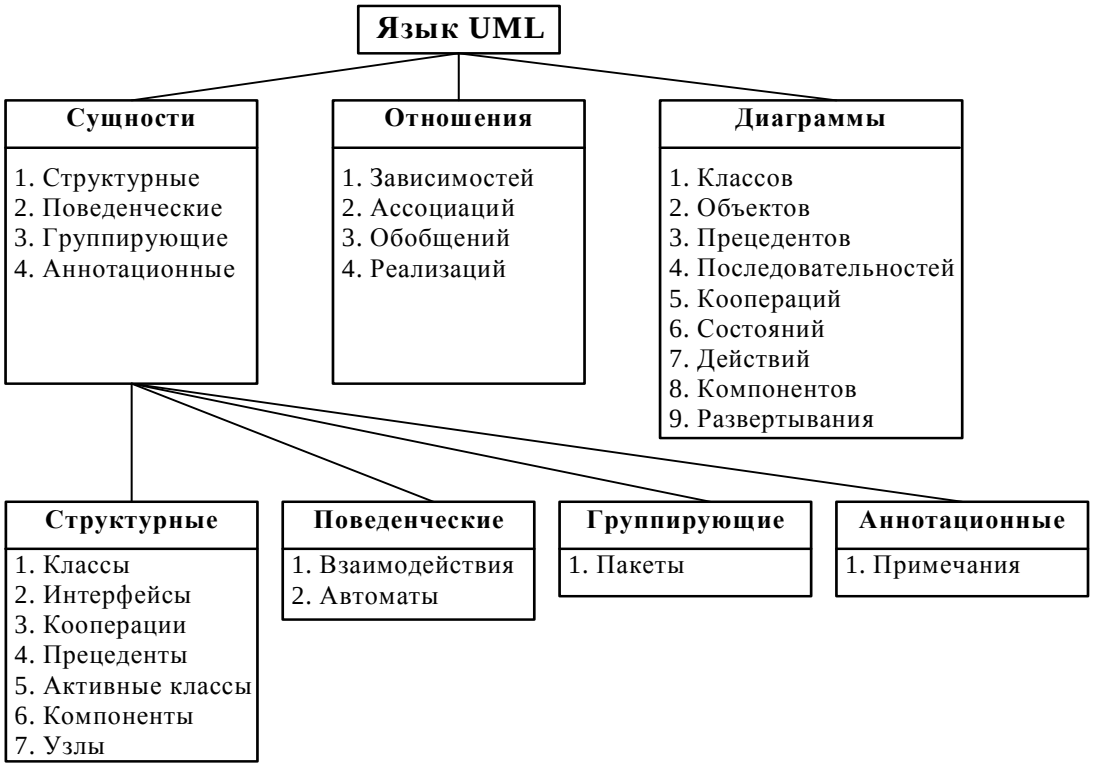


Рис. 6.1 – Структура языка UML

На рисунке 6.2 в качестве примеров показаны пять видов пиктограмм: класс, актер, прецедент, пакет и примечание.

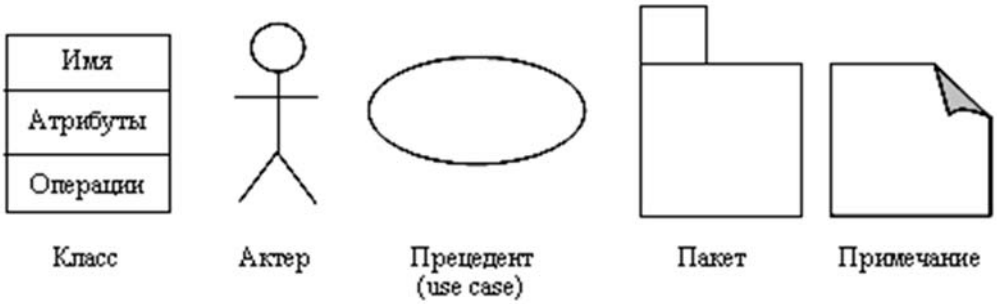


Рис. 6.2 – Пиктограммы UML – класс, актер,
прецедент, пакет и примечание

Класс изображается прямоугольником, разделенным на три поля. В первом поле помещается имя класса, однозначно определяющее данный класс среди множества других классов. Во втором поле помещаются атрибуты (общие свойства) класса. В третьем поле располагаются типовые операции, выполняемые объектами, принадлежащими данному классу. На рисунке 6.3 показаны пиктограммы, изображающие класс «окно» и класс «экранный кадр».

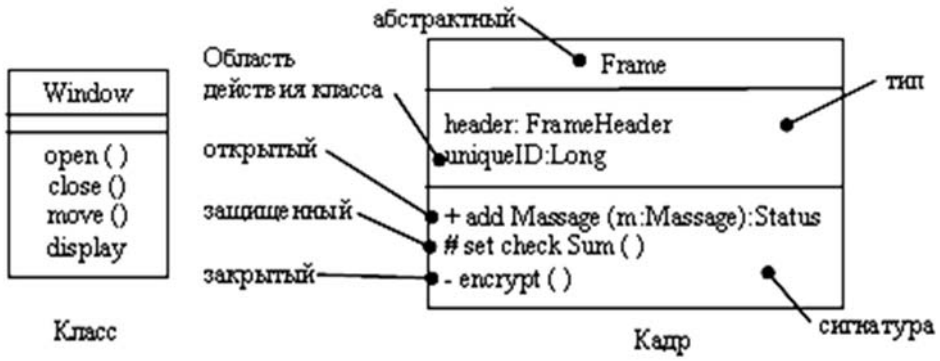


Рис. 6.3 – Пиктограммы UML, изображающие класс «окно»
и класс «экранный кадр»

Левая пиктограмма на рисунке 6.3 показывает класс с именем «окно» и основные свойства (характеристики), присущие объектам этого класса. Правая пиктограмма показывает класс «кадр (frame)» с его основными характеристиками и подробными характеристиками, детализирующими основные

характеристики. Детальная нотация класса дает возможность программистам и аналитикам визуализировать, специфицировать, конструировать и документировать класс на любом желаемом уровне детализации свойств класса, достаточном для поддержки прямого и обратного проектирования моделей и кода.

Актер – это внешний по отношению к компьютерной системе объект, взаимодействующий с ней. Графически актер изображается в виде пиктограммы, представляющей человека, поскольку актер – это человек или группа людей, использующих данные, предоставляемые компьютерной системой. Например, в системе регистрации учебных курсов, актером являются студенты, записывающиеся на курсы преподавателей через автоматизированную систему регистрации курсов.

Прецедент – это описание множества последовательных событий, выполняемых компьютерной системой, которые приводят к наблюдаемому актером результату. Графически прецедент изображается в виде ограниченного непрерывной линией эллипса, обычно содержащего только имя прецедента. На UML-диаграммах пиктограммы прецедента и актера обычно располагаются рядом.

Пакет – это единственная в языке UML первичная группирующая сущность. В пакет можно поместить структурные и поведенческие сущности и даже другие пакеты. Изображается пакет в виде папки с закладкой.

Примечание изображается в виде прямоугольника с загнутым краем. На UML-диаграмме примечание присоединяется к одному или нескольким элементам диаграммы. Внутри прямоугольника-примечания помещаются комментарии или ограничения, относящиеся к элементу (или нескольким элементам) диаграммы. Комментарий может быть текстовым или графическим.

Рассмотрим теперь пиктограммы отношений, используемых в UML-диаграммах. Однонаправленные отношения представляются на UML-диаграммах стрелками различных видов, а двунаправленное отношение представляется линией (рис. 6.4).

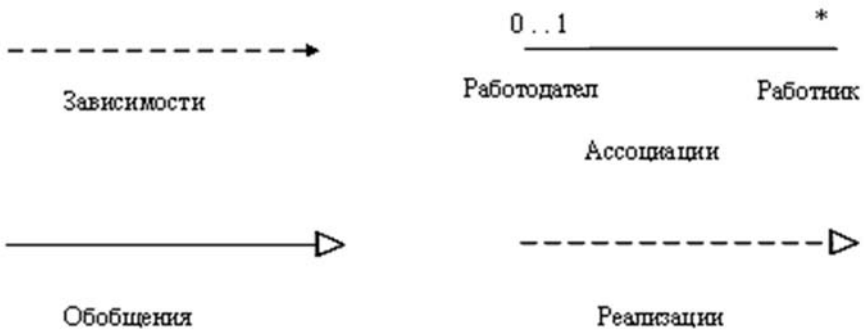


Рис. 6.4 – Виды отношений в UML

Показанное на рисунке однонаправленное отношение зависимость – это семантическое отношение между двумя сущностями, такое, при котором изменение одной (первичной) сущности вызывает изменение семантики другой, зависимой сущности.

Ассоциация – это структурное двунаправленное отношение, описывающее совокупность взаимоотношений между объектами. Пометка единица (1) на левом конце линии ассоциации означает, что в двунаправленном отношении, наряду с многими работниками, участвует один работодатель. Единица и звездочка на правом конце линии означает «единица или больше» (1..*). Если один конец линии ассоциации помечен единицей (1), то пометка на другом конце линий называется кратностью ассоциации. Кратность правого конца ассоциации равна единице или больше. На линии ассоциации можно также задать кратность, равную единице (1), можно указать диапазон кратности: ноль или единица (0..1), много (0..*). Разрешается также указывать кратность определенным числом (например, 5). С помощью списков можно задавать и более сложные кратности. Например, список 0..1, 3..4, 6..* означает «любое число объектов, кроме 2 и 5».Частным случаем ассоциации является отношение типа «часть/целое». Отношение такого типа называется агрегированием. В языке UML оно причислено к отношениям вида «имеет». Агрегирование изображается в виде ассоциации с не закрашенным ромбом со стороны целого, как показано на рисунке 6.5.

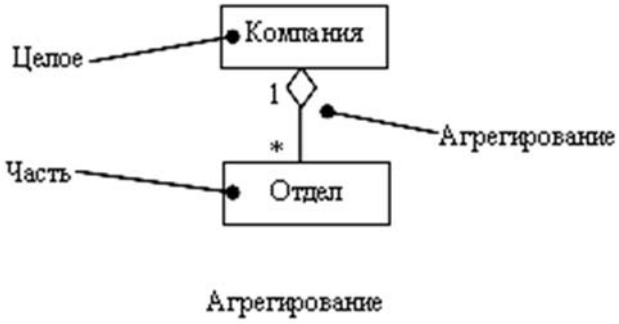


Рис. 6.5 – Агрегирование

Обобщение – это однонаправленное отношение, называемое «потомок/предок», в котором объект «потомок» может быть подставлен вместо объекта предка. Потомок наследует структуру и поведение своего предка. Стрелка всегда указывает на предка.

Реализация – это семантическое однонаправленное отношение, которое может устанавливаться, во-первых, между интерфейсами и реализующими их классами или компонентами, во-вторых, между прецедентами и реализующими их кооперациями. Интерфейсы, компоненты и кооперации – это еще три вида пиктограмм, относящихся к структурным сущностям. Пиктограммы интерфейса, кооперации и компонента показаны на рисунке 6.6.



Рис. 6.6 – Пиктограммы интерфейса, кооперации и компонента

Интерфейс – это совокупность операций, предоставляемых классом или компонентом. Следовательно, интерфейс описывает поведение класса

или компонента, видимое извне. Интерфейс определяет только описание (спецификации) операций класса или компонента, но он никогда не определяет физические реализации операций. Графически интерфейс изображается небольшим кружочком, под которым пишется его имя, как показано на рисунке 6.6.

Кооперация определяет взаимодействие, например, классов. Участвуя в кооперации классы совместно производят некоторый кооперативный результат. Один и тот же класс может принимать участие в нескольких кооперациях. Графически кооперация изображается в виде эллипса, ограниченного пунктирной линией.

Компонент – это физическая часть компьютерной или иной системы. Компонент соответствует некоторому набору интерфейсов и обеспечивает физическую реализацию этого набора. Компоненты могут быть разных видов. Например, одним из видов компонентов, используемых для моделирования программного обеспечения компьютерной системы, могут быть файлы исходного кода. Компонент, как правило, представляет собой физическую упаковку логических элементов, таких как классы, интерфейсы и кооперации. Графически компонент изображается в виде прямоугольника с вкладками. Внутри прямоугольника обычно пишется только имя компонента.

6.3 UML-диаграммы

С помощью комбинации пиктограмм строятся UML-диаграммы. Девять основных видов диаграмм перечислены в третьей ветви структурного дерева языка UML. Рассмотрим три из них: диаграммы прецедентов, диаграммы классов и диаграммы действий.

Диаграмма прецедентов – это графическое представление всех или части актеров, прецедентов и взаимодействий между ними. В каждой системе обычно есть главная диаграмма прецедентов, которая описывает внешнюю

границу системы и основные внешние функции (внешнее поведение) системы. В качестве примера диаграммы прецедентов рассмотрим диаграмму, изображающую все прецеденты для одного актера, которым является регистратор учебных курсов (см. рис. 6.7).

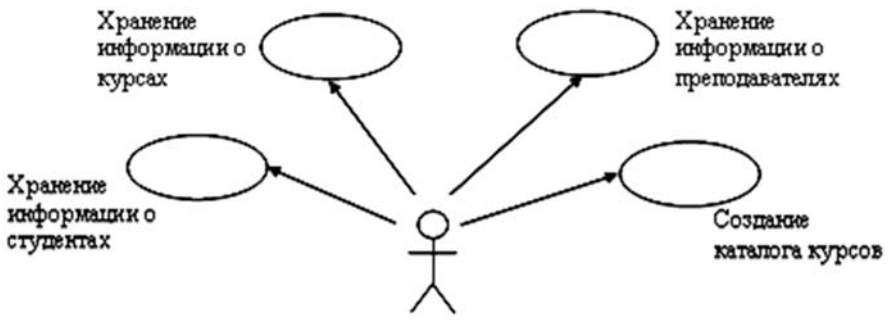


Рис. 6.7 – Диаграмма прецедентов

Диаграммы классов применяются для моделирования объектно-ориентированных систем. На простых диаграммах показываются классы и отношения между классами. На сложных диаграммах показываются классы, интерфейсы, кооперации и отношения между ними. Диаграммы классов дают статический вид системы. Можно также сказать, что диаграммы классов представляют собой взгляды разработчиков на статические состояния проектируемых систем. С помощью диаграмм классов составляется словарь системы. Диаграммы классов являются основой для создания диаграмм компонентов и развертывания. Следует особо подчеркнуть, что диаграммы классов важны не только для визуализации, специфицирования и документирования структурных моделей, но также для прямого и обратного проектирования исполняемых кодов систем.

UML-диаграммы классов включают в себя как частный случай диаграммы «сущность-связь», которые используются для логического проектирования реляционных, объектно-ориентированных и гибридных объектно-реляционных баз данных.

6.3.1 Основные понятия диаграмм классов UML

Диаграммой классов в терминологии UML называется диаграмма, на которой показан набор классов (и некоторых других сущностей, не имеющих явного отношения к проектированию БД), а также связей между этими классами (иногда термин «relationship» переводится на русский язык не как «связь», а как «отношение»). Кроме того, диаграмма классов может включать комментарии и ограничения. Ограничения могут неформально задаваться на естественном языке или же могут формулироваться на языке OCL (Object Constraints Language), который является частью общей спецификации UML, но, в отличие от других частей языка, имеет не графическую, а линейную нотацию. Мы затронем эту тему ниже более подробно.

Классом называется именованное описание совокупности объектов с общими атрибутами, операциями, связями и семантикой.

Атрибутом класса называется именованное свойство класса, описывающее множество значений, которые могут принимать экземпляры этого свойства. Класс может иметь любое число атрибутов. Свойство, выражаемое атрибутом, является свойством моделируемой сущности, общим для всех объектов данного класса. Так что атрибут является абстракцией состояния объекта. Любой атрибут любого объекта класса должен иметь некоторое значение.

Операцией класса называется именованная услуга, которую можно запросить у любого объекта этого класса. Операция – это абстракция того, что можно делать с объектом. Класс может содержать любое число операций. Набор операций класса является общим для всех объектов данного класса. Операции класса определяются в разделе, расположенном ниже раздела с атрибутами. При этом можно ограничиться только указанием имен операций, оставив более детальную спецификацию выполнения операций на поздние этапы моделирования. Описание операции может также содержать ее сигнатуру, т. е. имена и типы всех параметров, а если операция является

функцией, то и тип ее значения. Класс Человек с определенными операциями показан на рисунке 6.8.

В диаграмме классов могут участвовать связи трех разных категорий: зависимость, обобщение и ассоциирование. Для проектирования реляционных БД наиболее важны вторая и третья категории связей. Поэтому по поводу связей-зависимостей мы будет очень кратки.

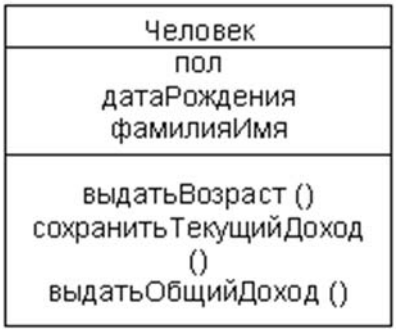


Рис. 6.8 – Отображение класса в UML

Связь-зависимость – связь по использованию, когда изменение в спецификации одного класса может повлиять на поведение другого, использующего первый класса. Чаще всего зависимости применяются в диаграммах классов, чтобы отразить в сигнатуре операции одного класса тот факт, что параметром этой операции могут быть объекты другого класса. Понятно, что если интерфейс этого второго класса изменяется, то это влияет на поведение объектов первого класса. Простой пример диаграммы классов со связью-зависимостью показан на рисунке 6.9.

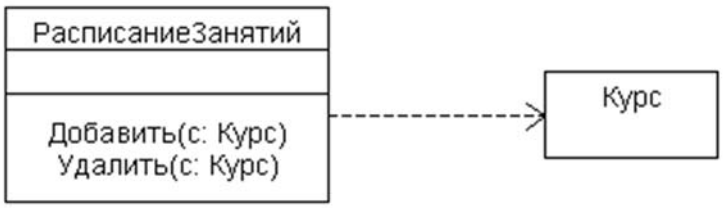


Рис. 6.9 – Диаграммы классов со связью-зависимостью

Легко видеть, что связи-зависимости существенны для объектно-ориентированных систем.

Связь-обобщение – связь между общей сущностью, называемой суперклассом, или родителем, и более специализированной разновидностью этой сущности, называемой подклассом, или потомком. Класс-потомок наследует все атрибуты и операции класса-предка, но в нем могут быть определены дополнительные атрибуты и операции.

Объекты класса-потомка могут использоваться везде, где могут использоваться объекты класса-предка. Это свойство называют полиморфизмом по включению, имея в виду, что объекты потомка можно считать включаемыми в класс-предок. Графически обобщения изображаются в виде сплошной линии с большой не закрашенной стрелкой, направленной к суперклассу. На рисунке 6.10 показан пример иерархии одиночного наследования: у каждого подкласса имеется только один суперкласс.

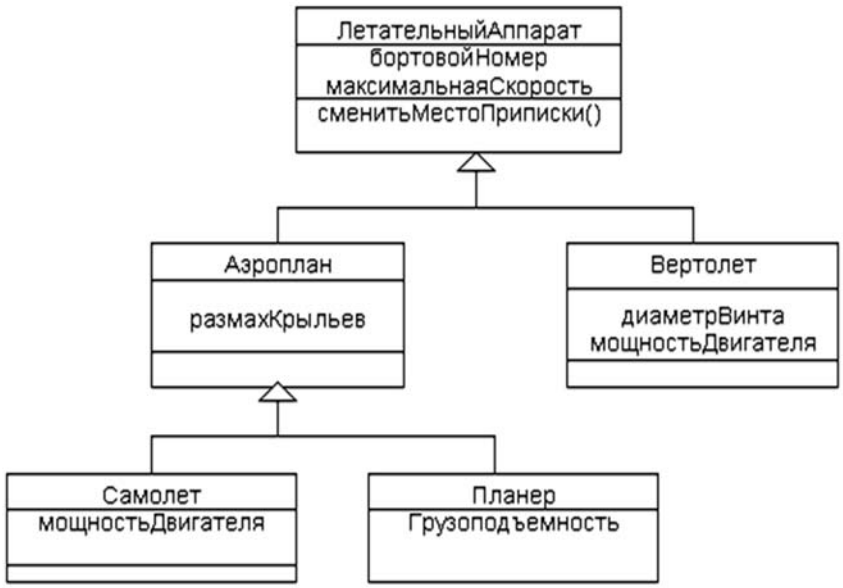


Рис. 6.10 – Иерархия одиночного наследования классов

Одиночное наследование является достаточным в большинстве практических случаев применения связи-обобщения. Однако в UML допускается и множественное наследование, когда один подкласс определяется

на основе нескольких суперклассов. В качестве одного из разумных примеров рассмотрим следующую диаграмму классов (для упрощения диаграммы не будем указывать имена атрибутов и операций) (см. рис. 6.11).

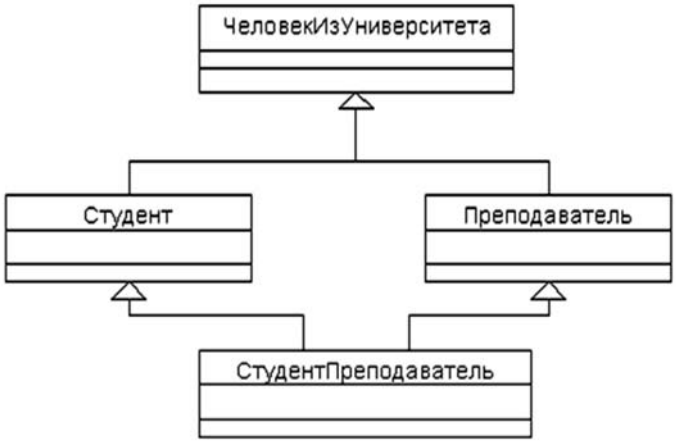


Рис. 6.11 – Пример множественного наследования классов

На этой диаграмме классы *Студент* и *Преподаватель* порождены из одного суперкласса *ЧеловекИзУниверситета*. Но как это часто случается, многие студенты уже в студенческие годы начинают преподавать, так что могут существовать такие два объекта классов *Студент* и *Преподаватель*, которым соответствует один объект класса *ЧеловекИзУниверситета*. Итак, среди объектов класса *Студент* могут быть преподаватели, а некоторые преподаватели могут быть студентами. Мы можем определить класс *СтудентПреподаватель* путем множественного наследования от суперклассов *Студент* и *Преподаватель*. Объект класса *СтудентПреподаватель* обладает всеми свойствами и операциями классов *Студент* и *Преподаватель* и может быть использован везде, где могут использоваться объекты этих классов. Так что полиморфизм по включению продолжает работать.

Но множественное наследование, помимо того, что не слишком часто требуется на практике, порождает ряд проблем, из которых одной из наиболее

известных является проблема именования атрибутов и операций в подклассе, полученном путем множественного наследования. Например, предположим, что при образовании подклассов *Студент* и П*реподаватель* в них обоих был создан атрибут с именем *номерКомнаты*. Очень вероятно, что для объектов класса *Студент* значениями этого атрибута будут номера комнат в студенческом общежитии, а для объектов класса *Преподаватель* – номера служебных кабинетов. Как быть с объектами класса *СтудентПреподаватель*, для которых существенны оба одноименных атрибута? На практике применяется одно из следующих решений:

1) запретить образование подкласса *СтудентПреподаватель*, пока в одном из суперклассов не будет произведено переименование атрибута *номерКомнаты*;

2) наследовать это свойство только от одного из суперклассов, так что, например, значением атрибута *номерКомнаты* у объектов класса *Студент-Преподаватель* всегда будут номера служебных кабинетов;

3) унаследовать в подклассе оба свойства, но автоматически переименовать оба атрибута, чтобы прояснить их смысл; назвать их, например, *номерКомнатыСтудента* и *номерКомнатыПреподавателя*.

Ни одно из решений не является полностью удовлетворительным. Первое решение требует возврата к ранее определенному классу, имена атрибутов и операций которого, возможно, уже используются в приложениях. Второе решение нарушает логику наследования, не давая возможности на уровне подкласса использовать все свойства суперклассов. Наконец, третье решение заставляет использовать длинные имена атрибутов и операций, которые могут стать недопустимо длинными, если процесс множественного наследования будет продолжаться от полученного подкласса.

Но, конечно, сложность проблемы именования атрибутов и операций не идет ни в какое сравнение со сложностью реализации множественного

наследования в реляционных БД. Поэтому при использовании UML для проектирования реляционных БД нужно очень осторожно использовать наследование классов и стараться вообще избегать множественного наследования.

6.3.2 Связи-ассоциации

Ассоциацией называется структурная связь, показывающая, что объекты одного класса некоторым образом связаны с объектами другого или того же самого класса. Допускается, чтобы оба конца ассоциации относились к одному классу. В ассоциации могут связываться два класса, и тогда она называется бинарной. Допускается создание ассоциаций, связывающих сразу n классов (они называются *n*-арными ассоциациями). Графически ассоциация изображается в виде линии, соединяющей класс сам с собой или с другими классами.

С понятием ассоциации связаны четыре важных дополнительных понятия: имя, роль, кратность и агрегация. Во-первых, ассоциации может быть присвоено имя, характеризующее природу связи. Смысл имени уточняется черным треугольником, располагаемым над линией связи справа или слева от имени ассоциации. Этот треугольник указывает направление, в котором должно читаться имя связи. Пример именованной ассоциации показан на рисунке 6.12 – треугольник означает, что именованная ассоциация должна читаться как «Студент учится в Университете».



Рис. 6.12 – Пример именованной ассоциации

Другим способом именования ассоциации является указание роли каждого класса, участвующего в этой ассоциации. Роль класса задается именем,

помещаемым под линией ассоциации ближе к данному классу. На рисунке 6.13 показаны две ассоциации между классами Человек и Университет, в которых эти классы играют разные роли. Как мы видим, объекты класса Человек могут выступать в роли РАБОТНИКОВ при участии в ассоциации, в которой объекты класса Университет играют роль НАНИМАТЕЛЯ. В другой ассоциации объекты класса Человек играют роль СТУДЕНТА, а объекты класса УНИВЕРСИТЕТ – в роли ОБУЧАЮЩЕГО.



Рис. 6.13 – Две ассоциации с разными ролями классов

В общем случае, для ассоциации может задаваться и ее собственное имя, и имена ролей классов. Это связано с тем, что класс может играть одну и ту же роль в разных ассоциациях, так что в общем случае пара имен ролей классов не идентифицирует ассоциацию. С другой стороны, в простых случаях, когда между двумя классами определяется только одна ассоциация, можно вообще не связывать с ней дополнительные имена.

Кратностью (multiplicity) роли ассоциации называется характеристика, указывающая, сколько объектов класса с данной ролью может или должно участвовать в каждом экземпляре ассоциации (в UML экземпляр ассоциации называется соединением – link). Наиболее распространенным способом задания кратности роли ассоциации является указание конкретного числа или диапазона. Например, указание «1» говорит о том, что все объекты класса с данной ролью должны участвовать в некотором экземпляре

данной ассоциации, причем в каждом экземпляре ассоциации может участвовать ровно один объект класса с данной ролью. Указание диапазона «0..1» говорит о том, что не все объекты класса с данной ролью обязаны участвовать в каком-либо экземпляре данной ассоциации, но в каждом экземпляре ассоциации может участвовать только один объект. Аналогично, указание диапазона «1..*» говорит, что все объекты класса с данной ролью должны участвовать в некотором экземпляре данной ассоциации и в каждом экземпляре ассоциации должен участвовать хотя бы один объект (верхняя граница не задана). Толкование диапазона «0..*» является очевидным расширением случая «0..1».

В более сложных (но крайне редко встречающихся на практике) случаях определения кратности можно использовать списки диапазонов. Например, список «2, 4..6, 8..*» говорит, что все объекты класса с указанной ролью должны участвовать в некотором экземпляре данной ассоциации и в каждом экземпляре ассоциации должны участвовать два, от четырех до шести или более восьми объектов класса с данной ролью.

На диаграмме классов на рисунке 6.14 показано, что произвольное (может быть, нулевое) число людей являются работниками произвольного числа университетов. Каждый университет обучает произвольное (может быть, нулевое) число студентов, но каждый студент может быть студентом только одного университета.



Рис. 6.14 – Ассоциации с указанными кратностями ролей

Обычная ассоциация между двумя классами характеризует связь между равноправными сущностями: оба класса находятся на одном концептуальном уровне. Иногда в диаграмме классов требуется отразить тот факт, что ассоциация между двумя классами имеет специальный вид «часть-целое». В этом случае класс «целое» имеет более высокий концептуальный уровень, чем класс «часть». Ассоциация такого рода называется агрегатной. Графически агрегатные ассоциации изображаются в виде простой ассоциации с не закрашенным ромбом на стороне класса-«целого». Простой пример агрегатной ассоциации показан на рисунке 6.15.



Рис. 6.15 – Пример агрегатной ассоциации

Объектами класса *Аудитория* являются студенческие аудитории, в которых проходят занятия. В каждой аудитории должны быть установлены парты. Поэтому в некотором смысле класс Парта является «частью» класса *Аудитория*. Мы умышленно сделали роль класса Парта необязательной, поскольку могут существовать аудитории без парт (например, класс для занятий танцами) и некоторые парты могут находиться на складе. Обратите внимание, что, хотя аудитории, не оснащенные партами, как правило, непригодны для занятий, объекты классов *Аудитория* и *Парта* существуют независимо. Если некоторая аудитория ликвидируется, то находящиеся в ней парты не уничтожаются, а переносятся на склад.

Бывают случаи, когда связь «части» и «целого» настолько сильна, что уничтожение «целого» приводит к уничтожению всех его «частей». Агрегатные ассоциации, обладающие таким свойством, называются композитными, или просто композициями. При наличии композиции объект-часть

может быть частью только одного объекта-целого (композита). При обычной агрегатной ассоциации «часть» может одновременно принадлежать нескольким «целым». Графически композиция изображается в виде простой ассоциации, дополненной закрашенным ромбом со стороны «целого». Пример композитной агрегатной ассоциации показан на рисунке 6.16.



Рис. 6.16 – Пример композитной агрегатной ассоциации

Любой факультет является частью одного университета, и ликвидация университета приводит к ликвидации всех существующих в нем факультетов (хотя во время существования университета отдельные факультеты могут ликвидироваться и создаваться).

В контексте проектирования реляционных БД агрегатные и в особенности композитные ассоциации влияют только на способ поддержки ссылочной целостности. В частности, композитная связь является явным указанием того, что ссылочная целостность между «целым» и «частями» должна поддерживаться путем каскадного удаления частей при удалении целого. При наличии простой ассоциации между двумя классами предполагается возможность навигации между объектами, входящими в один экземпляр ассоциации.

Если известен конкретный объект-студент, то должна обеспечиваться возможность узнать соответствующий объект-университет. Если известен конкретный объект-университет, то должна обеспечиваться возможность узнать все соответствующие объекты-студенты. Другими словами, если не оговорено противное, то навигация по ассоциации может проводиться

в обоих направлениях. Однако бывают случаи, в которых желательно ограничить направление навигации для некоторых ассоциаций. В этом случае на линии ассоциации ставится стрелка, указывающая направление навигации. Возможный пример показан на рисунке 6.17.



Рис. 6.17 – Ассоциация с указанным направлением навигации

В библиотеке должно содержаться некоторое число книг, но каждая книга должна принадлежать некоторой библиотеке. С точки зрения библиотечного хозяйства разумно иметь возможность найти книгу в библиотеке, т. е. произвести навигацию от объекта-библиотеке к связанным с ним объектам-книгам. Однако вряд ли потребуется по данному экземпляру книги узнать, в какой библиотеке она находится.

6.3.3 Сущности

Основными объектно-ориентированными блоками являются сущности. В языке UML имеется четыре вида сущностей: структурные, поведенческие, группирующие, аннотационные. Структурные сущности – это имена существительные в моделях на языке UML. Они представляют собой статические части модели, соответствующие концептуальным или физическим элементам системы. Существует пять разновидностей концептуальных и логических сущностей.

Класс (Class) – это описание совокупности объектов с общими атрибутами, операциями, отношениями и семантикой (рис. 6.18). Класс реализует один или несколько интерфейсов.

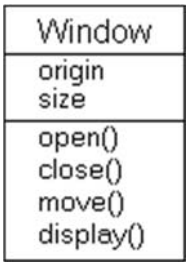


Рис. 6.18 – Классы

Интерфейс (Interface) – это совокупность операций, которые определяют набор услуг, предоставляемый классом или компонентом. Интерфейс описывает видимое извне поведение элемента (рис. 6.19). Интерфейс редко существует сам по себе – обычно он присоединяется к реализующему его классу или компоненту.



Рис. 6.19 – Интерфейс

Кооперация (Collaboration) определяет взаимодействие и представляет совокупность ролей, работающих совместно, производящих некоторый кооперативный эффект. Кооперация имеет как структурный, так и поведенческий аспект – один и тот же класс может принимать участие в нескольких кооперациях (рис. 6.20).



Рис. 6.20 – Кооперация

Прецедент (Use case) – это описание последовательности выполняемых системой действий, которая производит наблюдаемый результат,

значимый для какого-то определенного *актера* (Actor). Прецедент применяется для структурирования поведенческих сущностей модели и реализуются посредством кооперации (рис. 6.21).

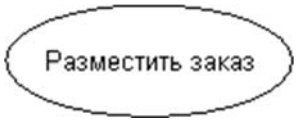


Рис. 6.21 – Прецедент

Активным классом (Active class) называется класс, объекты которого вовлечены в один или несколько процессов (Threads) и могут инициировать управляющее воздействие. Активный класс отличается от обычного класса тем, что деятельность его объектов осуществляется одновременно с деятельностью других элементов. Графически активный класс изображается так же, как простой класс, но ограничивающий прямоугольник рисуется жирной линией и обычно включает имя, атрибуты и операции (рис. 6.22).

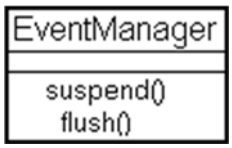


Рис. 6.22 – Активный класс

Компоненты и узлы соответствуют физическим сущностям системы. Компонент (Component) – это физическая заменяемая часть системы, соответствующая некоторому набору интерфейсов и обеспечивающая его реализацию (рис. 6.23). Компонент – это физическая упаковка логических элементов (классов, интерфейсов и кооперации), например, компоненты COM+ или Java Beans, а также файлы исходного кода.

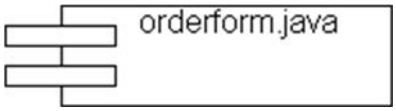


Рис. 6.23 – Компонент

Узел (Node) – это элемент, представляющий вычислительный ресурс, обладающий памятью и способностью обработки. Совокупность компонентов может размещаться в узле, а также мигрировать с одного узла на другой (рис. 6.24).



Рис. 6.24 – Узел

Существуют также разновидности этих семи сущностей: актеры, сигналы, утилиты (виды классов), процессы и нити (виды активных классов), приложения, документы, файлы, библиотеки, страницы и таблицы (виды компонентов).

Поведенческие сущности (Behavioral things) являются динамическими составляющими модели UML. Это глаголы языка: они описывают поведение модели. Существует всего два типа поведенческих сущностей.

Взаимодействие (Interaction) – это поведение, суть которого заключается в обмене сообщениями между объектами для достижения определенной цели. С помощью взаимодействия описывается как отдельная операция, так и поведение совокупности объектов. Взаимодействие предполагает ряд других элементов, таких как сообщение (рис. 6.25), последовательность действий (поведение, инициированное сообщением) и связь (между объектами).

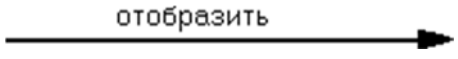


Рис. 6.25 – Сообщение

Автомат (State machine) – это алгоритм поведения, определяющий последовательность состояний, через которые объект или взаимодействие

проходят на протяжении своего жизненного цикла в ответ на различные события, а также реакции на эти события (рис. 6.26). С помощью автомата можно описать поведение отдельного класса или кооперации классов. С автоматом связан ряд других элементов: состояния, переходы (из одного состояния в другое), события (сущности, инициирующие переходы) и виды действий (реакция на переход).

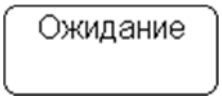


Рис. 6.26 – Состояние

Взаимодействия и автоматы семантически связаны с различными структурными элементами, такими как классы, кооперации и объекты.

6.3.4 Отношения

В языке UML определены четыре типа отношений: зависимость, ассоциация, обобщение, реализация. Эти отношения являются основными связующими строительными блоками в UML.

Зависимость (Dependency) – это семантическое отношение между двумя сущностями, при котором изменение одной из них, независимой, может повлиять на семантику другой, зависимой (рис. 6.27).

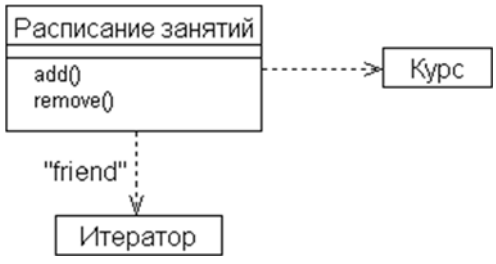


Рис. 6.27 – Отношения зависимости

Ассоциация (Association) – отношение, описывающее совокупность связей между объектами. Разновидностью ассоциации является *агрегирование* (Aggregation) – структурное отношение между целым и его частями

(рис. 6.28). Графическое изображение ассоциации может включать кратность и имена ролей (рис. 6.29).

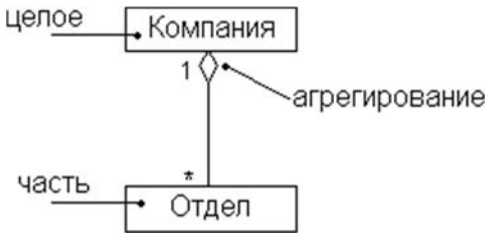


Рис. 6.28 – Агрегирование



Рис. 6.29 – Имена ассоциаций

Обобщение (Generalization) – это отношение «специализация/обобщение» (рис. 6.30), при котором объект специализированного элемента (потомок) может быть подставлен вместо объекта обобщенного элемента (родителя или предка). Таким образом, потомок (Child) наследует структуру и поведение своего родителя (Parent).

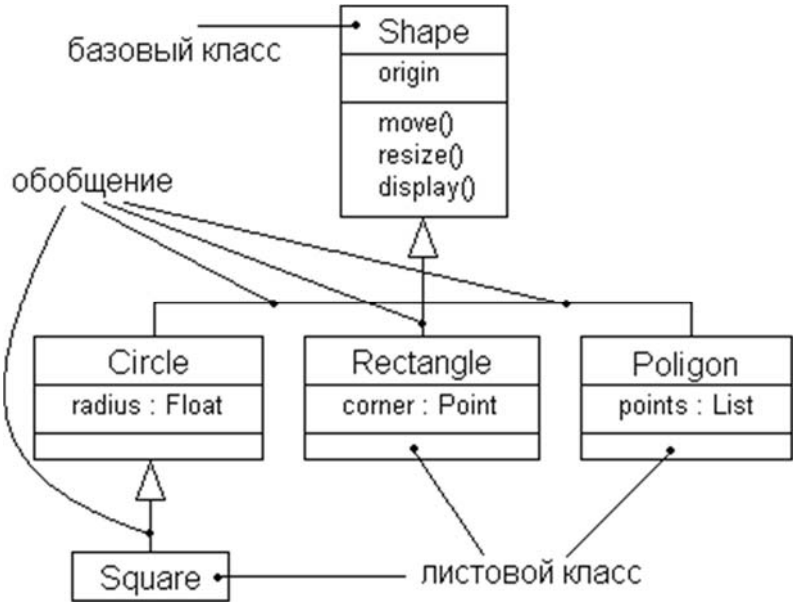


Рис. 6.30 – Обобщение

Реализация (Realization) – это отношение между классификаторами, при котором один классификатор определяет «контракт», а другой гарантирует его выполнение (рис. 6.31).

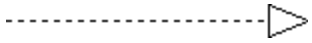


Рис. 6.31 – Реализация

Отношения реализации встречаются в двух случаях: во-первых, между интерфейсами и реализующими их классами или компонентами, а во-вторых, между прецедентами и реализующими их кооперациями.

6.4 Краткое описание работы в Umbrello UML Modeller

Umbrello UML Modeller (http://umbrello.kde.org/) помогает в процессе разработки программного обеспечения, используя промышленный стандарт – язык UML (UML), предоставляя возможность создавать диаграммы для разработки и документирования ваших систем. Все действия в Umbrello UML Modeller доступны через меню и панели инструментов, но Umbrello UML Modeller также интенсивно использует контекстное меню. Вы можете нажать правую кнопку мыши практически на любом элементе рабочей области Umbrello UML Modeller или дерева модели для вызова меню с наиболее полезными функциями, которые могут применяться к данному элементу. Некоторые пользователи найдут это немного неудобным в начале своей работы, потому что они чаще используют меню или панели инструментов, но впоследствии не смогут не согласиться, что данный метод значительно увеличивает скорость работы.

6.4.1 Пользовательский интерфейс

Главное окно Umbrello UML Modeller делится на три области (см. рис. 6.32), которые предоставляют полный обзор системы и быстрый

доступ к различным диаграммам при работе над моделью. Перечислим эти области:

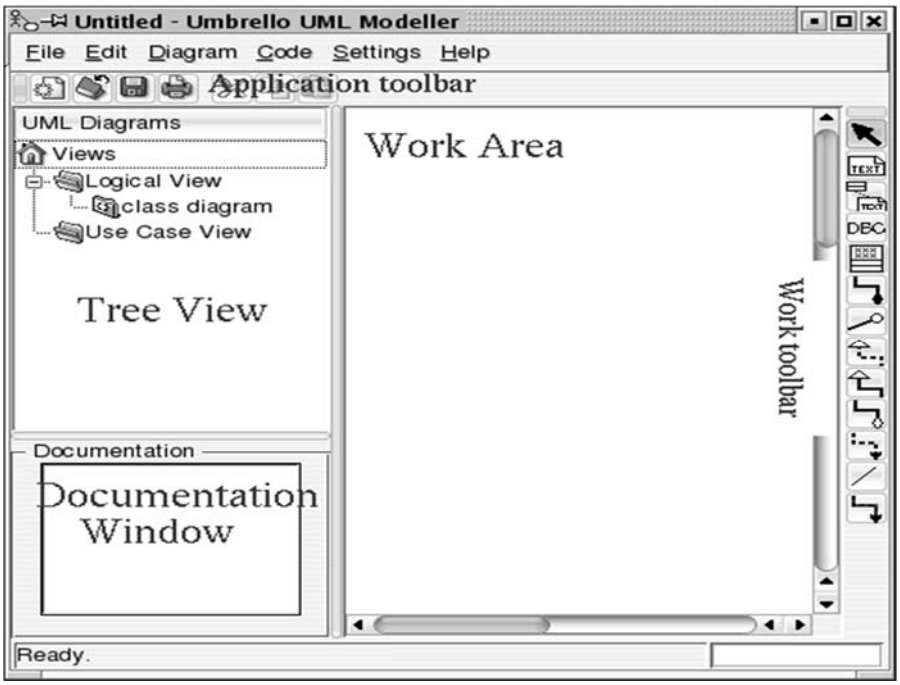


Рис. 6.32 – Интерфейс пользователя Umbrello UML Modeller

- Дерево модели (tree view).
- Рабочая область (work area).
- Окно документации (documentation window).

Дерево модели располагается в левом верхнем углу окна приложения и отображает все диаграммы, классы, действующие лица и варианты действий, которые составляют модель. Дерево модели даёт обзор разрабатываемой модели в целом, одновременно позволяя быстрое переключение между элементами модели и возможность размещения элементов модели на текущей диаграмме.

Если вы работаете над моделью с несколькими классами и диаграммами, дерево модели поможет вам организовать элементы вашей модели с помощью папок. Папки можно создавать с помощью контекстного меню

и затем перемещать в них элементы дерева модели, просто перетаскивая их мышкой.

Окно документации расположено в левом нижнем углу Umbrello UML Modeller и предоставляет краткую информацию о выделенном объекте. Окно документации специально сделано небольшим, чтобы просматривать информацию по объекту, не занимая много места на экране. Если понадобится просмотреть более подробную документацию по элементу, то следует перейти к свойствам элемента с помощью контекстного меню.

Рабочая область – это основное окно приложения, в котором происходит вся работа с диаграммами модели. Рабочая область отображает текущую активную диаграмму. В одно время может отображаться только одна диаграмма.

6.4.2 Создание, загрузка и сохранение моделей

Первое, что необходимо сделать для работы в Umbrello UML Modeller, – это создать новую модель. При запуске Umbrello UML Modeller автоматически загружает последнюю использованную модель или создаёт новую, пустую модель (данное поведение зависит от настроек Umbrello UML Modeller).

Для создания новой модели необходимо выбрать элемент Новый из меню Файл или просто нажать на иконку Новый на панели инструментов. Если в это время вы уже работали с какой-либо моделью и не сохранили изменения, то Umbrello UML Modeller предложит их сохранить.

Можно сохранить модель в любое время с помощью элемента Сохранить из меню Файл или с помощью нажатия на иконку Сохранить на панели инструментов. Если требуется сохранить вашу модель под другим именем, то для этого следует выбрать элемент Сохранить как из меню Файл.

Для удобства Umbrello UML Modeller также предлагает опцию для автоматического сохранения вашей работы через определённый промежуток

времени. При необходимости этот параметр и период времени можно настроить через меню Настройка.

Для загрузки уже существующей модели следует выбрать элемент Открыть из меню Файл или нажать на иконку Открыть на панели инструментов. Модели, которые использовались в последнее время, отображаются в подменю Открыть недавние в меню Файл.

Umbrello UML Modeller может работать только с одной моделью. Таким образом, если вы, работая с несохраненной моделью, попробуете загрузить другую, то приложение запросит вас о сохранении ваших данных.

В Umbrello UML Modeller существует два основных способа редактирования элементов вашей модели:

– редактирование элементов напрямую через дерево модели;

– редактирование элементов модели через диаграмму.

Используя контекстное меню различных элементов дерева модели, вы имеете возможность добавлять, удалять и модифицировать практически все элементы вашей модели. Контекстное меню папок дерева модели позволяет создавать различные типы диаграмм, а также, в зависимости от того, является ли текущая папка вариантом использования или отображением, создавать действующих лиц, варианты использования, классы и так далее.

После добавления нового элемента в модель, его также можно редактировать через диалог свойств, который доступен через элемент Свойства контекстного меню дерева модели. Также можно редактировать созданную модель, добавляя и удаляя элементы на диаграммах, это будет подробнее рассмотрено в последующих разделах.

Созданная UML-модель состоит из набора UML-элементов и связей между ними. Тем не менее автор не может видеть модель напрямую, для этого надо использовать диаграммы.

Для создания новой диаграммы в созданной модели рекомендуется выбрать необходимый тип диаграммы через элемент Новый из меню Диаграмма и указать его имя. Диаграмма будет создана, станет активной и отобразится в дереве модели.

Следует помнить, что Umbrello UML Modeller интенсивно использует контекстные меню: вы можете нажать правую кнопку на папке в дереве модели и выбрать соответствующий тип диаграммы из элемента Новый контекстного меню. Отметим, что можно создать диаграммы вариантов использования только в папках обзора вариантов использования, а другие типы диаграмм могут быть созданы только в папках логического отображения.

Для удаления диаграммы из созданной модели следует выбрать соответствующую диаграмму и выбрать элемент Удалить из меню Диаграмма. Того же можно достичь, выбрав элемент Удалить из контекстного меню диаграммы в дереве модели.

Так как удаление диаграммы является серьёзным действием, которое может привести к потере данных, Umbrello UML Modeller потребует подтверждение операции удаления перед её выполнением. После того, как диаграмма будет удалена и файл проекта будет сохранён, нет никакой возможности восстановить удалённую информацию.

Если требуется изменить имя существующей диаграммы, следует выбрать элемент Переименовать из контекстного меню диаграммы в дереве модели. Другим способом переименования диаграммы является диалог свойств диаграммы, который доступен через Свойства контекстного меню или через двойной щелчок по диаграмме в дереве модели.

При работе с диаграммой Umbrello UML Modeller пытается помочь вам с помощью применения нескольких простых правил, например какие элементы могут размещаться на диаграммах определённого типа или какие взаимоотношения могут быть между ними. Если вы являетесь экспертом

в UML, то можете даже не обратить на это внимание, но это будет хорошей подсказкой для новичков при создании стандартных диаграмм.

После создания диаграмм их следует отредактировать. Следует понимать разницу между редактированием диаграммы и редактированием модели. Как было уже сказано, диаграммы являются отображениями модели. Например, если вы создаёте класс, редактируя диаграмму класса, то вы одновременно редактируете и диаграмму, и модель. Если же вы изменяете цвет или другие настройки отображения класса на вашей диаграмме класса, то вы редактируете только диаграмму, но ничего не изменяете в вашей модели.

Первое, что делается при редактировании новой диаграммы, – вставляются элементы в неё (классы, действующие лица, варианты использования и так далее). Существует два основных метода:

– перемещение существующих элементов в модель из дерева модели;

– создание новых элементов в модели и одновременное добавление их в диаграмму с помощью кнопок панели инструментов.

Для вставки элементов, уже существующих в модели, просто перетащите их из дерева модели на их место на диаграмме. Всегда можно перемещать элементы по диаграмме, выделяя их кнопкой выбора на панели инструментов. Второй метод добавления элементов в диаграмму – использование кнопок панели инструментов (следует отметить, что это приведёт к добавлению элементов в модель).

Панель инструментов по умолчанию находилась в правой стороне окна приложения, в новой версии панель перемещена на верх окна. Можно переместить панель в любую сторону окна или оставить в виде отдельного окна. Набор инструментов, представленных на этой панели в виде кнопок, зависит от типа текущей диаграммы. Кнопка текущего инструмента активирована на Панели. Можно переключиться на инструмент Стрелка с помощью нажатия на клавишу Esc.

При выборе инструмента из панели (например, инструмента вставки классов) указатель мыши меняет свой вид на перекрестье, и вы можете вставлять элементы данного типа в модель одиночными щелчками на диаграмме. Следует отметить, что элементы в UML должны иметь уникальные имена. Таким образом, если на одной диаграмме модели есть класс с именем «ClassA» и вы пытаетесь создать на другой диаграмме новый класс с таким же именем, Umbrello UML Modeller не позволит этого сделать, вам придётся дать классу другое уникальное имя. Если вы пытаетесь добавить тот же элемент на диаграмму с помощью инструмента вставки классов, то вы явно ошиблись с выбором инструмента. Вместо этого вам следует перетащить класс на диаграмму из дерева модели.

Можно удалить любой элемент диаграммы, выбрав элемент удалить из его контекстного меню.

Существует большая разница между удалением объекта из диаграммы и удалением объекта из модели. Если вы удаляете объект из диаграммы, то он удаляется только из определённой диаграммы: элемент продолжает оставаться частью вашей модели и это действие никак не влияет на остальные диаграммы. С другой стороны, если вы удаляете элемент из дерева модели, то происходит удаление элемента из модели. Так как данный элемент больше не присутствует в модели, он автоматически удаляется из всех диаграмм, где он присутствовал.

Можно редактировать большинство элементов UML в вашей модели и в диаграммах через диалог Свойства контекстного меню элементов. Каждый элемент имеет диалог, содержащий несколько страниц с соответствующими параметрами элемента. Для некоторых элементов, например, для якорей, вы можете изменять только имя элемента и текстовое пояснение, в то время как для других элементов, таких как классы, вы можете редактировать их свойства и методы, выбирать, что следует отображать на диаграмме (всю сигнатуру методов или только их имена и так далее).

Для большинства элементов UML можно открыть диалог свойств двойным щелчком на элементе при условии, что выбран инструмент Стрелка. Единственным исключением являются ассоциации, в их случае двойной щелчок создаёт точку якоря. Для Ассоциаций следует использовать контекстное меню для доступа к диалогу редактирования свойств.

Следует отметить, что доступ к свойствам элемента можно получить через контекстное меню элементов в дереве модели. Это позволяет вам глобально менять свойства элемента для всех диаграмм.

Хотя редактирование свойств всех объектов было уже рассмотрено в предыдущих разделах, классы заслуживают отдельного упоминания, потому что они отличаются от остальных элементов своей сложностью и имеют большее количество опций, чем большинство других элементов UML.

В диалоге редактирования свойств класса можно отредактировать всё – от используемого цвета до методов и свойств, которые он имеет. Назначение страницы общих настроек диалога свойств очевидно. На ней можно изменять имя класса, область видимости, документацию и так далее. Эта страница всегда доступна.

На странице настроек атрибутов можно добавлять, редактировать и удалять атрибуты (переменные) класса, перемещать атрибуты вверх и вниз по списку с помощью стрелок. Эта страница всегда доступна.

Подобно странице настроек атрибутов, на странице настроек методов можно добавлять, редактировать и удалять методы класса. При добавлении или редактировании метода вводятся основные данные в диалог Свойства метода. Если необходимо добавить параметры для метода, следует нажать на кнопку Новый параметр, которая вызовет появление диалога Свойства параметра. Эта страница всегда доступна.

Данная страница позволяет добавлять шаблоны классов, которые являются неопределёнными классами или типами данных. В Java 1.5 их будут обозначать термином «Generic».

Страница Ассоциации класса отображает все ассоциации данного класса на текущей диаграмме. Двойной щелчок на ассоциации показывает её свойства, и в зависимости от типа ассоциации, вы можете редактировать некоторые параметры, например, настройки множественности или ролевое имя. Если ассоциация не позволяет изменение этих параметров, то диалог свойств ассоциации позволяет изменять только документацию ассоциации.

Данная страница доступна, только если вы открыли свойства класса из диаграммы. Если вы открыли свойства класса из контекстного меню дерева модели, данная страница будет недоступна.

На странице Настройки отображения вы можете указать, что следует показывать на диаграмме. Класс может быть показан только в виде прямоугольника с именем класса в нём (что особенно полезно при наличии множества классов на диаграмме, или если на текущий момент не важна информация по классу) или с отображением всей информации по классу.

В зависимости от объёма информации, которая должна отображаться, вы можете активировать соответствующие опции на данной странице. Изменения, вносимые здесь, влияют только на отображение диаграммы. Это означает, что настройка «скрытия» методов класса приведёт к тому, что методы данного класса не будут показаны на диаграмме, но будут являться частью модели. Данная настройка доступна только, если вы открыли свойства класса из диаграммы.

На странице Цвет элемента можно выбрать цвета для линий и заливки объекта. Очевидно, что опции на данной странице имеют смысл только для классов, показанных на диаграммах, и недоступны при открытии диалога свойств класса из дерева модели.

Ассоциации связывают два UML-элемента друг с другом. Обычно ассоциации определяются между двумя классами, но некоторые типы ассоциаций могут также существовать между вариантами использования и действующими лицами.

Для создания ассоциации выбирается соответствующий инструмент из панели инструментов (общая ассоциация, обобщение, объединение и так далее), затем нужно щелкнуть сначала по первому, а потом по второму элементу, участвующему в ассоциации. Не следует пытаться перетащить мышкой первый элемент на второй, это ни к чему не приведёт.

Если вы попытаетесь создать ассоциацию, противоречащую спецификации UML, Umbrello UML Modeller отменит ваше действие, и вы получите сообщение об ошибке. Например, это может произойти, если существует обобщение от класса A к классу B и вы пытаетесь создать обобщение от класса B к классу A.

У ассоциации есть контекстное меню. При необходимости удалить ассоциацию просто выберите из него элемент Удалить. Также через контекстное меню можно открыть диалог свойств элемента.

По умолчанию ассоциации рисуются в виде прямой линии, соединяющей два объекта в диаграмме.

Добавление точки прикрепления производится с помощью двойного щелчка на линии ассоциации, после этого появившуюся точку можно перемещать для придания нужной формы линии ассоциации.

Если необходимо убрать точку прикрепления из линии ассоциации, рекомендуется щелкнуть по ней два раза.

Следует отметить, что редактирование свойств ассоциаций можно произвести только через их контекстное меню. В отличие от других UML элементов двойной щелчок на ассоциации приводит к созданию точки прикрепления, а не к появлению диалога редактирования свойств элемента.

Заметки, строки текста и группировка являются элементами, которые могут быть размещены на диаграммах любого типа и не несут смысловой нагрузки для модели, но очень полезны для оформления диаграмм.

Для добавления заметки или строки текста выберите соответствующий инструмент из панели инструментов и один раз щёлкните там, где хотите разместить элемент на диаграмме. Вы можете редактировать текст элемента через контекстное меню или, в случае заметки, двойного щелчком на нём.

Якоря используются для прикрепления заметки к элементу UML. Например, если вы обычно используете текст для описания или комментирования класса или ассоциации, то якорь позволяет явным образом «прикрепить» заметку к определённому элементу.

Для добавления якоря между заметкой и другим UML-элементом используйте соответствующий инструмент из панели инструментов. Щёлкните не по заметке, а по элементу UML, к которому следует прикрепить данную заметку.

6.4.3 Импорт и генерация кода

Umbrello UML Modeller является средством UML-моделирования, и его главной целью является помощь в анализе и проектировании ваших систем. Тем не менее для перехода от вашего проекта к его реализации Umbrello UML Modeller позволяет вам сгенерировать исходный код для различных языков программирования. Если вам понадобится использовать UML для существующего проекта на C++, Umbrello UML Modeller может помочь в создании модели вашей системы на основе исходного кода с помощью его анализа и импорта обнаруженных классов.

Umbrello UML Modeller может генерировать исходный код для различных языков программирования на основе вашей модели UML, помогая вам начать реализацию вашего проекта. Сгенерированный код состоит из объявлений классов с их методами и атрибутами, вам остаётся только «заполнить пустые места» для реализации функциональности вашего класса.

Umbrello UML Modeller поставляется с поддержкой генерации кода для ActionScript, Ada, C++, CORBA IDL, Java™, JavaScript, PHP, Perl, Python, SQL и XMLSchema.

Для генерации кода с помощью Umbrello UML Modeller вам сперва необходимо создать или загрузить модель, содержащую хотя бы один класс. При необходимости работы с исходным кодом выберите элемент Мастер генерации кода меню, Код для запуска мастера, который проведёт вас через процесс генерации кода.

Первым шагом будет указание классов, для которых требуется сгенерировать код. По умолчанию выбираются все классы вашей модели, вы имеете возможность убрать выделение с ненужных классов, перемещая их на левую сторону списка.

7 РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ

7.1 Общие сведения о техническом задании

Техническое задание представляет собой документ, в котором сформулированы основные цели разработки, требования к программному продукту, определены сроки и этапы разработки и регламентирован процесс приемно-сдаточных испытаний. В разработке технического задания участвуют как представители заказчика, так и представители исполнителя. В основе этого документа лежат исходные требования заказчика, анализ передовых достижений техники, результаты выполнения научно-исследовательских работ, предпроектных исследований, научного прогнозирования и т. п.

На рисунке 7.1 схематически показаны основные факторы, определяющие характеристики разрабатываемого программного обеспечения. Такими факторами являются:

– исходные данные и требуемые результаты, которые определяют функции программы или системы;

– среда функционирования (программная и аппаратная) – может быть задана, а может выбираться для обеспечения параметров, указанных в техническом задании;

– возможное взаимодействие с другим программным обеспечением и/или специальными техническими средствами – также может быть определено, а может выбираться исходя из набора выполняемых функций.

Разработка технического задания выполняется в следующей последовательности. Прежде всего устанавливают набор выполняемых функций, а также перечень и характеристики исходных данных. Затем определяют перечень результатов, их характеристики и способы представления. Далее уточняют среду функционирования программного обеспечения: конкретную комплектацию и параметры технических средств, версию используемой

операционной системы и, возможно, версии и параметры другого установленного программного обеспечения, с которым предстоит взаимодействовать будущему программному продукту.

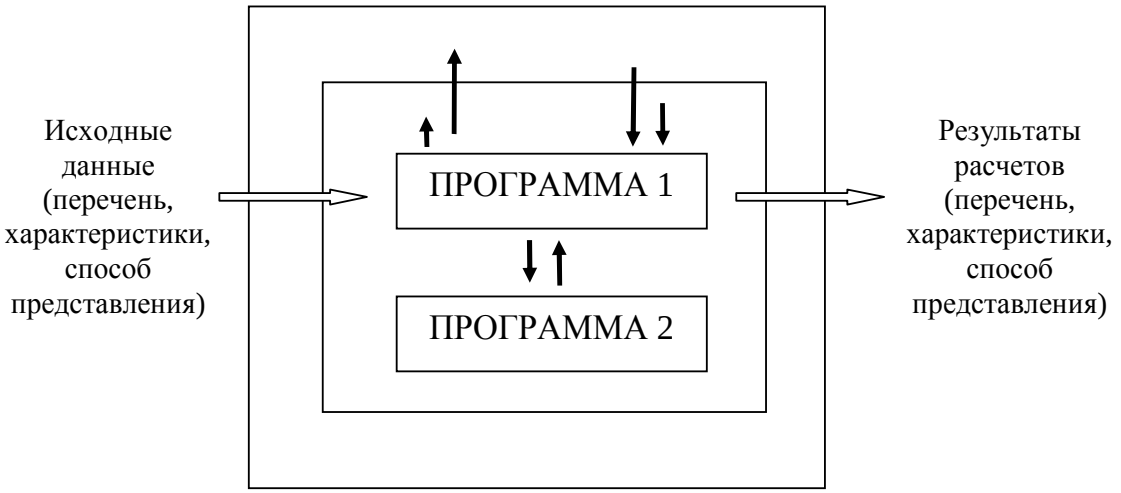


Рис. 7.1 – Основные факторы, определяющие характеристики разрабатываемого программного обеспечения

В случаях, когда разрабатываемое программное обеспечение собирает и хранит некоторую информацию или включается в управление каким-либо техническим процессом, необходимо также четко регламентировать действия программы: в случае сбоев оборудования и энергоснабжения.

На техническое задание существует стандарт ГОСТ 19.201–78 «Техническое задание. Требования к содержанию и оформлению». В соответствии с этим стандартом 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;

– порядок контроля и приемки.

При необходимости допускается в техническое задание включать приложения.

Рассмотрим более подробно содержание каждого раздела.

«Введение» должно включать наименование и краткую характеристику области применения программы или программного продукта, а также объекта (например, системы), в котором предполагается их использовать. Основное назначение введения – продемонстрировать актуальность данной разработки и показать, какое место эта разработка занимает в ряду подобных.

Раздел «Основания для разработки» должен содержать наименование документа, на основании которого ведется разработка, организации, утвердившей данный документ, и наименование или условное обозначение темы разработки. Таким документом может служить план, приказ, договор и т. п.

Раздел «Назначение разработки» должен содержать описание функционального и эксплуатационного назначения программного продукта с указанием категорий пользователей.

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

– требования к функциональным характеристикам;
– требования к надежности;
– условия эксплуатации;
– требования к составу и параметрам технических средств;
– требования к информационной и программной совместимости;
– требования к маркировке и упаковке;
– требования к транспортированию и хранению;
– специальные требования.

Наиболее важным из перечисленных выше является подраздел «Требования к функциональным характеристикам». В этом разделе должны быть

перечислены выполняемые функции и описаны состав, характеристики и формы представления исходных данных и результатов. В этом же разделе при необходимости указывают критерии эффективности: максимально допустимое время ответа системы, максимальный объем используемой оперативной и/или внешней памяти и др. Если разработанное программное обеспечение не будет выполнять указанных в техническом задании функций, то оно считается не соответствующим техническому заданию, т. е. неправильным с точки зрения критериев качества. Универсальность будущего продукта также обычно специально не оговаривается, но подразумевается.

В подразделе «Требования к надежности» указывают уровень надежности, который должен быть обеспечен разрабатываемой системой, и время восстановления системы после сбоя. Для систем с обычными требованиями к надежности в этом разделе иногда регламентируют действия разрабатываемого продукта по увеличению надежности результатов (контроль входной и выходной информации, создание резервных копий промежуточных результатов и т. п.).

В подразделе «Условия эксплуатации» указывают особые требования к условиям эксплуатации: температуре окружающей среды, относительной влажности воздуха и т. п. Как правило, подобные требования формулируют, если разрабатываемая система будет эксплуатироваться в нестандартных условиях или использует специальные внешние устройства, например для хранения информации. Здесь же указывают вид обслуживания, необходимое количество и квалификацию персонала. В противном случае допускается указывать, что требования не предъявляются.

В подразделе «Требования к составу и параметром технических средств» указывают необходимый состав технических средств с указанием их основных технических характеристик: тип микропроцессора, объем памяти, наличие внешних устройств и т. п. При этом часто указывают два варианта конфигурации: минимальный и рекомендуемый.

В подразделе «Требования к информационной и программной совместимости» при необходимости можно задать методы решения, определить язык или среду программирования для разработки, а также используемую операционную систему и другие системные и пользовательские программные средства, с которым должно взаимодействовать разрабатываемое программное обеспечение. В этом же разделе, при необходимости, указывают, какую степень защиты информации необходимо предусмотреть.

В разделе «Требования к программной документации» указывают необходимость наличия руководства программиста, руководства пользователя, руководства системного программиста, пояснительной записки и т. п. На все эти типы документов также существуют ГОСТы.

В разделе «Технико-экономические показатели» рекомендуется указывать ориентировочную экономическую эффективность, предполагаемую годовую потребность и экономические преимущества по сравнению с существующими аналогами.

В разделе «Стадии и этапы разработки» указывают стадии разработки, этапы и содержание работ с указанием сроков разработки и исполнителей.

В разделе «Порядок контроля и приемки» указывают виды испытаний и общие требования к приемке работы.

В приложениях при необходимости приводят: перечень научно-исследовательских работ, обосновывающих разработку; схемы алгоритмов, таблицы, описания, обоснования, расчеты и другие документы, которые следует использовать при разработке.

В зависимости от особенностей разрабатываемого продукта разрешается уточнять содержание разделов, т. е. использовать подразделы, вводить новые разделы или объединять их.

В случаях, если какие-либо требования, предусмотренные техническим заданием, заказчик не предъявляет, следует в соответствующем месте указать «Требования не предъявляются».

Разработка технического задания – процесс трудоемкий, требующий определенных навыков. Наиболее сложным, как правило, является четкое формулирование основных разделов: введения, назначения и требований к программному продукту. В качестве примеров рассмотрим два технических задания на выполнение курсового проектирования: ТЗ, составленное по сокращенной схеме, и сравнительно полное ТЗ на выполнение госбюджетной научно-исследовательской работы.

7.2 Рекомендации по оформлению ТЗ

Рекомендации по оформлению ТЗ изложены в таблице 7.1.

Таблица 7.1 – Порядок построения и изложения ТЗ

№ раздела/ подраздела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
1	Основание для выполнения	Основанием для выполнения работы является	
2	Цель выполнения	2.1 Цель выполнения 2.2 Разрабатываемые образцы продукции (приводят полное наименование разрабатываемых образцов) предназначены для: (приводят описание области применения разрабатываемых образцов)	
3	Головной исполнитель и соисполнители	3.1. Головной исполнитель (приводят полное наименование организации головного исполнителя с указанием места фактического нахождения) 3.2. Соисполнители (перечисляют организации-соисполнители с указанием полного	

Продолжение табл. 7.1

№ раздела/ подраздела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
		наименования и места фактического нахождения) – в части (приводят описание направления работы для каждой организации)	
4	Технические требования		
4.1	Состав опытного образца продукции	Опытный образец продукции должен содержать следующие составные части (перечисляют основные составные части образца или приводят требования к его составу, а также, при необходимости, указывают назначение составных частей. Если разрабатываемый образец должен иметь несколько исполнений, то в ТЗ должна быть определена его базовая конструкция и приведен состав каждого исполнения):	ГОСТ 2.711–82
		– комплекс аппаратно-программных средств (в т. ч. программное обеспечение) (при наличии);	ГОСТ 19.202–78 ГОСТ Р 51904–2002
		– эксплуатационная документация;	ГОСТ 2.601–95
		– запасное имущество и принадлежности (ЗИП) – (при наличии)	
		– упаковка и тара	
		При необходимости уточнения состава изделия на последующих этапах ОКР допускается дополнять подраздел фразой «Окончательный состав (приводится полное наименование разрабатываемых(ого) изделий(я))	

Продолжение табл. 7.1

№ раз-дела/ под-раз-дела	Наименова-ние раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
		уточняется в процессе разра-ботки образца и согласовыва-ется с (приводится наименова-ние организации, с которой тре-буется согласование	
		состава изделия) на стадии (при-водится наименование этапа ОКР или календарного плана, на котором будет согласован окончательный состав изделия»	
4.2	Требования по назначе-нию	4.2.1 Выполняемые функции 4.2.2 Нормы и количественные показатели, определяющие эф-фективность изделия (точность выполнения операций, диапазон, производительность) в сравнении с мировыми аналогами. 4.2.3 Технические характери-стики (параметры) изделия, обеспечивающие выполнение возложенных на него задач (мощность, чувствительность, коэффициент полезного дей-ствия, время готовности к работе, время непрерывной или цикличе-ской работы и т. п.) в сравнении с мировыми аналогами. 4.2.4 Требования к порядку и способам взаимодействия с сопрягаемыми объектами, пара-метрам воздействий, поступаю-щих на сопрягаемые изделия от разрабатываемого изделия или на разрабатываемое изделие	Ключевые тре-бования должны быть приведены в сравнении с лучшими ми-ровыми дости-жениями с ука-занием страны, фирмы и года достижения

Продолжение табл. 7.1

№ раздела/ подраздела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
		от сопрягаемых объектов, способам обмена информацией. 4.2.5 Требования к функциональной, геометрической, биологической, электромагнитной, электрической, прочностной, программной, технологической, метрологической, диагностической, организационной, информационной и другим видам совместимости. 4.2.6 Требования к составу и квалификации обслуживающего персонала 4.2.7 Требования по мобильности изделия (стационарное, возимое, переносное и т. д.)	
4.3	Требования к конструкторской и технологической документации	4.3.1 Конструкторская, технологическая и программная документация должна соответствовать требованиям стандартов ЕСКД, ЕСТД, ЕСПД. 4.3.2 Номенклатура разрабатываемой конструкторской, эксплуатационной, ремонтной, технологической и программной документации приведена в приложениях	ГОСТ 2.102–68 ГОСТ 3.1102–81 ГОСТ 3.1119–83 ГОСТ 3.1121–84 ГОСТ 2.601–95 ГОСТ 2.602–95 ГОСТ 19.101–77
4.4	Требования к электропитанию	Приводят величины напряжения, частоты переменного тока, допустимые колебания напряжения и частоты и др. Указывают максимальную мощность, потребляемую изделием, и этап разработки, на котором уточняется ее значение	ГОСТ 13109–97

Продолжение табл. 7.1

№ раз-дела/ под-раз-дела	Наименова-ние раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
4.5	Требования по надежно-сти		
4.5.1	Требования по безотказ-ности	Указывают требуемые значения, показателей надежности, выби-раемые по ГОСТ 27.003–90, в том числе: – вероятности безотказной ра-боты; – средней наработки на отказ; – средней наработки на неис-правность; – среднее время восстановления; и т. п. Подтверждение заданных ТЗ требований проводится: расчетным методом – на этапе разработки изделия в соответ-ствии с ГОСТ 27.301–95 «Надежность в технике. Расчет надежности. Основные положе-ния»; экспериментальным (рас-чётно-экспериментальным) мето-дом на этапе предварительных и приёмочных испытаний по мето-дике, согласованной с заказчиком	ГОСТ 27.002–89 ГОСТ 27.402–95 ГОСТ 27.301–95 ГОСТ 27.410–87
4.5.2	Требования по долговеч-ности	Указывают средние, назначен-ные или гамма-процентные: – ресурс между средними (капи-тальными) ремонтами; – ресурс до списания. Срок службы между средними (капитальными) ремонтами. Срок службы до списания	ГОСТ 27.002–89 ГОСТ 15467–79

Продолжение табл. 7.1

№ раздела/ подраздела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
4.5.3	Требования по сохраняемости	Указывают: – гамма-процентный срок сохраняемости; – средний срок сохраняемости	ГОСТ 27.002–89 ГОСТ 21493–76
4.5.4	Критерии отказов и предельного состояния изделия	Устанавливают критерии отказов и предельного состояния изделия, применительно к которым устанавливают показатели долговечности и сохраняемости	ГОСТ 27.003–90
4.6	Требования по эргономике и технической эстетике	Приводят эргономические требования к организации и средствам деятельности человека-оператора: – к распределению функций, алгоритмам работы операторов, способам решения поставленных задач, циклограммам деятельности, режиму труда и отдыха, средствам отображения информации, организации рабочего места и т. д. Приводят требования по технической эстетике, определяющие композиционную целостность, информационную выразительность, рациональность формы и культуру производственного выполнения создаваемого изделия	ГОСТ 12.2.049–80 ГОСТ 21752–76 ГОСТ 21753–76 ГОСТ 22613–77 ГОСТ 22614–77 ГОСТ 22615–77 ГОСТ 22902–78 ГОСТ 20.39.108–85 ГОСТ 30.001–83 ГОСТ 21480–76 ГОСТ 21829–76 ГОСТ Р 50949–2001
4.7	Требования по эксплуатации, удобству технического обслуживания, ремонту и хранения		
4.7.1	Указывают: – вид климатического исполнения изделия и, при необходимости, требования к изделию в части видов и номинальных значений климатических факторов; – группу механического исполнения изделия		ГОСТ 15150–69 ГОСТ 50766–95 ГОСТ 17516.1–90

Продолжение табл. 7.1

№ раздела/ подраздела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
4.7.2	Устанавливают требования по условиям и срокам хранения в различных условиях		ГОСТ 15150–69 ГОСТ 23216–78 ГОСТ 51908–2002
4.7.3	Устанавливают требования к составу и квалификации обслуживающего персонала		
4.7.4	Устанавливают требования по эксплуатационным и дежурным режимам		
4.7.5	Устанавливают требования по времени непрерывной или циклической работы		
4.7.6	Устанавливают требования по условиям эксплуатации в аварийных ситуациях		
4.7.7	Устанавливают требования к системе средств эксплуатационного контроля		ГОСТ 26656–85
4.7.8	Требования по ремонтопригодности	Указывают требуемые значения: – средней оперативной продолжительности планового (не планового) текущего ремонта; – средней оперативной трудоемкости технического обслуживания и т. п.	ГОСТ 23660–79 ГОСТ 21623– 76 ГОСТ 27.410–87 ГОСТ 28.001–83
4.7.9	Требования по удобству ремонта	Устанавливают требования по удобству ремонта в условиях эксплуатации и на ремонтном предприятии, сборки и разборки при техническом обслуживании, доступности к отдельным составным частям при выполнении этих операций без демонтажа других составных частей	ГОСТ 2.602–95 ГОСТ 21623–76 ГОСТ 23660 ГОСТ 2.604–2000 ГОСТ 28.001–83
4.7.10	Требования по исключению возможности неправильной сборки	Устанавливают требования по исключению возможности неправильной сборки и неправильного подключения кабелей, шлангов и других ошибок персонала во время эксплуатации, технического обслуживания и ремонта	

Продолжение табл. 7.1

№ раз-дела/ под-раз-дела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
4.7.11	Устанавливают требования к составу инструментов и приспособлений для проведения технического обслуживания, и ремонта		
4.7.12	Устанавливают требования к гарантийным срокам		ГОСТ 22352–77
4.7.13	Устанавливают требования к эксплуатационным документам (ЭД)		ГОСТ 2.601–95
4.7.14	Требования к ЗИП	Устанавливают исходные данные для расчёта ЗИП: состав, стратегия пополнения, доставки ЗИП и т. п. Требования уточняются на этапе технического проекта	
4.7.15	Приводят требования к ремонтным документам (РД) – для ремонтно-пригодных изделий		ГОСТ 2.602–95
4.8	Требования по транспортабельности	Перечисляют виды транспортных средств, на которых возможна перевозка изделия (автомобильный, железнодорожный, морской и авиационный транспорт). Устанавливают требования по условиям транспортирования и допустимым механическим и климатическим воздействиям при транспортировании	ГОСТ 15150–69 ГОСТ 51908–2002
4.9	Требования по безопасности	Приводят требования: – по безопасности при монтаже, эксплуатации, обслуживании и ремонте (от воздействия электрического тока, теплового воздействия, высокочастотных полей, ядовитых и взрывчатых паров, пыли и газов, акустических шумов и т. п.).	ГОСТ ССБТ (серия 12), соответствующие тематике проекта. ГОСТ ИСО/ТО 12100-1–2001 ГОСТ ИСО/ТО 12100-2–2002

Продолжение табл. 7.1

№ раздела/ подраздела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
		– по обеспечению охраны окружающей среды при производстве, эксплуатации, транспортировании, хранении, утилизации продукции	
4.10	Требования по технологичности	Устанавливают базовые значения показателей технологичности в соответствии с ГОСТ 14.201–83 «Обеспечение технологичности конструкции изделий. Общие требования»	ГОСТ 14.201–83 ГОСТ 24444–87
4.11	Конструктивные требования	Устанавливают конструктивные требования к создаваемому изделию и составным частям: – габаритные, – установочные, – присоединительные размеры, – масса, – способы крепления, – регулировка органов управления, – соответствие образцам-эталонам, – виды покрытий и т. п. При необходимости уточнения конструктивных требований на последующих этапах ОКР допускается дополнять подраздел фразой «Значения ___ (перечень конструктивных требований, требующих уточнения) уточняются в процессе разработки изделия на этапе (указывается этап ОКР) и согласовываются с ____»	

Продолжение табл. 7.1

№ раздела/ подраздела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
		организации, с которой требуется согласование)» Приводят требования по обеспечению жизнедеятельности обслуживающего персонала	
4.12	Прогноз развития требований	4.12.1 На предполагаемый пятилетний период выпуска разрабатываемых образцов продукции прогнозируется повышение с ___ до _____. 4.12.2 Рекомендуемый срок модернизации продукции с учетом повышения эффективности ____ – 20___ год	
5	Технико-экономические требования	5.1. В процессе выполнения проекта должны быть достигнуты следующие значения индикаторов Количество публикаций (указать класс изданий), содержащих информацию о результатах интеллектуальной деятельности, полученных в рамках выполнения проектов научно-исследовательских работ: ___ – 2012 г. ___ – 2013 г.; Количество разработанных технологий, соответствующих или превосходящих мировой уровень: ___ – 2012 г. ___ – 2013 г.; Количество патентов (в том числе международных) на результаты интеллектуальной деятельности, полученные в рамках выполнения проектов научно-	

Продолжение табл. 7.1

№ раздела/ подраздела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
		исследовательских и опытно-конструкторских работ:___ – 2012 г.___ – 2013 г.; Количество диссертаций на соискание ученых степеней, защищенных в рамках выполнения проектов научно-исследовательских и опытно-конструкторских работ по приоритетным направлениям научно-технологического развития: ___ – 2011 г. ___ – 2013 г. Объем внебюджетных средств, привлеченных в рамках реализации важнейшего предметно-ориентированного инновационного проекта: (млн руб.): ___ – 2011 г.___ – 2012 г. ___ – 2013 г.; Объем новой и усовершенствованной высокотехнологичной продукции, дополнительно произведенной в результате реализации важнейшего предметно-ориентированного инновационного проекта (млн руб.): ___ – 2011 г. ___ – 2012 г. ___ – 2013 г.	
		5.2. Приводят требования, обеспечивающие разработку изделия в обоснованных пределах полной стоимости его жизненного цикла и с оптимальной трудоемкостью серийного производства и технического обслуживания в процессе	

Продолжение табл. 7.1

№ раздела/ подраздела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
		эксплуатации. В разделе указывают ориентировочные предельные значения: – стоимости разработки изделия; – стоимости строительства или переоборудования объектов, необходимых для проведения испытаний опытных образцов	
6	Требования по видам обеспечения		
6.1	Требования по метрологическому обеспечению	Приводят требования: – к системе и методам измерений и контроля параметров образцов продукции; – к контролю пригодности; – к средствам измерения и контроля для комплектации образца; – к разработке специальных средств измерений для технического и регламентного обслуживания образца при эксплуатации; – к метрологической экспертизе технической документации и средств измерений	ГОСТ-серия 8, в том числе: ГОСТ 8.009–84. ПР 50.2.009–94 ГОСТ 26656–85
6.2	Требования по программному обеспечению	Приводят требования по программному обеспечению (ПО) встроенных систем: – к схемам алгоритмов и программ; – к языку программирования и т. д.; – к составу и структуре общего и специального обеспечения; – к операционным системам; – к алгоритмам, точности и времени решения задач;	ГОСТ Р 51904–2002 ГОСТ 19.001–77 ГОСТ 19.101–77 ГОСТ 19.106–77 ГОСТ 19.201–78 ГОСТ 19.301–79 ГОСТ 19.402–78 ГОСТ 19.701–90 ГОСТ 28388–89

Продолжение табл. 7.1

№ раздела/ подраздела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
		– к ресурсу памяти вычислительных средств; – к составу программной и эксплуатационной документации	
6.3	Требования по другим видам обеспечения	Подразделы вводятся по усмотрению заказчика	
7	Требования к консервации, упаковке и маркировке		
7.1	Требования к консервации	Приводят требования к консервации с учетом сроков и условий хранения изделия (в том числе необходимость консервации перед упаковкой) на открытых площадках, под навесами, в хранилищах и т. п.	ГОСТ 9.014–78
7.2	Требования к упаковке	Приводят требования к упаковке, способу и возможным вариантам упаковки в зависимости от сроков и условий хранения и транспортирования	ГОСТ 23170–78 ГОСТ 2.114–95 ГОСТ 23216–78 ГОСТ 24686–81
7.3	Требования к маркировке	Приводят требования к маркировке, наносимой на изделия и тару (место нанесения, способ нанесения, требования к качеству маркировки, содержанию предупредительных и указательных подписей)	ГОСТ 26828–86 ГОСТ 14192–96 ГОСТ Р 51474–99
8	Требования к патентной чистоте и патентоспособности	8.1 На этапах ____ (указываются этапы) календарного плана должны быть проведены патентные исследования в соответствии с ГОСТ Р 15.011–96. 8.2 Должна быть определена рыночная стоимость интеллектуальной собственности в соответствии с Методическими	ГОСТ Р 15.011–96 ГОСТ 15.012–84

Продолжение табл. 7.1

№ раз-дела/ под-раз-дела	Наименова-ние раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
		рекомендациями, утвержден-ными Министерством имуще-ственных отношений 26 ноября 2002 г. № СК-4/21297. 8.4 Патентная чистота на методы изготовления и конструктивные решения должна быть обеспечена в отношении Российской Федера-ции и следующих стран, куда возможна поставка изделий, а также передача технической, информационной и другой доку-ментации: _____	
9	Специальные требования	Приводят требования: – к виду и составу специального оборудования и оснастки, необ-ходимые для обеспечения экс-плуатации и технического об-служивания образца продукции. – к специальному ремонтно-тех-нологическому оборудованию, предназначенному для ремонта и поддержания образца в рабо-тоспособном состоянии в про-цессе эксплуатации. – по разработке средств для обеспечения испытаний и моде-лирования образца. – к методам испытаний образца при разработке и серийном про-изводстве, необходимость раз-работки его математической мо-дели, программных средств	

Окончание табл. 7.1

№ раздела/ подраздела	Наименование раздела/ подраздела	Содержание, рекомендуемый текст	ГОСТы, рекомендуемые для становления показателей
10	Перечень, сроки выполнения и стоимость этапов	«Наименование этапов, содержание выполняемых работ, перечень документов, разрабатываемых на этапах выполнения ОКР, сроки исполнения и контрактная цена приведены в календарном плане»	ГОСТ 2.103–68 ГОСТ 2.119–73 ГОСТ 2.120–73 ГОСТ Р 15.201–2000 ГОСТ Р 15.005–86
11	Порядок выполнения и приемки этапов	11.1 Работа должна выполняться в соответствии с требованиями ГОСТ Р 15.201–2000 (ГОСТ Р 15.005–86). 11.2 Количество опытных образцов для предварительных испытаний – ___ (количество образцов уточняется на этапе ___). 11.3 Количество опытных образцов для приемочных испытаний – __ (количество образцов уточняется на этапе __). 11.4 Место проведения предварительных и приемочных испытаний – ____. 11.5 Приемка этапов работ производится в соответствии с требованиями ГОСТ Р 15.201–2000 (ГОСТ Р 15.005–86) и Положением о порядке приемки работ	
12	Освоение серийного производства	Приводится перечень работ, выполняемых за счет внебюджетных средств	

Ниже приведен пример написания календарного плана выполнения работ.

КАЛЕНДАРНЫЙ ПЛАН

_____

_____от _____ № _____

По согласованию с заказчиком допускается объединять, исключать этапы работ и (или) их содержание, а также вводить другие этапы работ. Согласно п. 2. ГОСТ 19.102–77 «Единая система программной документации. Стадии разработки» процесс выполнения ОКР **в общем случае** состоит из следующих этапов:

№ п/п	**Наименование этапов**	**Содержание выполняемых работ**	**Перечень документов, разрабатываемых на этапах выполнения ОКР**	**Срок исполнения (начало – окончание)**	**Контрактная цена, (тыс. руб.)**
1	Эскизный проект	Предварительная разработка структуры входных и выходных данных. Уточнение методов решения задачи. Разработка общего описания алгоритма решения задачи Разработка технико-экономического обоснования. Разработка пояснительной записки. Проведение патентных исследований по ГОСТ 15.011–96.	**Пояснительная записка к эскизному проекту.** Перечень результатов интеллектуальной деятельности, полученных в ходе выполнения этапа. Решение НТС предприятия по защите ЭП. Технический акт. Акт сдачи-приемки		

№ п/п	Наименование этапов	Содержание выполняемых работ	Перечень документов, разрабатываемых на этапах выполнения ОКР	Срок исполнения (начало – окончание)	Контрактная цена, (тыс. руб.)
		Согласование и утверждение эскизного проекта			
2	Технический проект	Уточнение структуры входных и выходных данных. Разработка алгоритма решения задачи. Определение формы представления входных и выходных данных. Определение семантики и синтаксиса языка. Разработка структуры программы. Окончательное определение конфигурации технических средств. Разработка плана мероприятий по разработке и внедрению программ. Разработка пояснительной записки. Проведение дополнительных	**Пояснительная записка к техническому проекту.** Перечень результатов интеллектуальной деятельности, полученных в ходе выполнения этапа. Решение НТС предприятия по защите ТП. Технический акт. Акт сдачи-приемки		

№ п/п	Наименование этапов	Содержание выполняемых работ	Перечень документов, разрабатываемых на этапах выполнения ОКР	Срок исполнения (начало – окончание)	Контрактная цена, (тыс. руб.)
		патентных исследований. Согласование и утверждение технического проекта			
3	Рабочий проект	Программирование и отладка программы. Изготовление программы-оригинала. Разработка программных документов в соответствии с требованиями ГОСТ 19.101–77. Разработка, согласование и утверждение программы и методики (ПМ) предварительных испытаний (ПИ). Проведение дополнительных патентных исследований	Комплект программных документов. ПМ ПИ. Утвержденный Акт готовности программной документации к ПИ. Перечень результатов интеллектуальной деятельности, полученных в ходе выполнения этапа. Решение НТС предприятия по защите ТП. Технический акт. Акт сдачи-приемки		
4	Предварительные испытания	Проведение предварительных испытаний. Корректировка программы и программной документации	Протоколы испытаний по пунктам программы. Перечень результатов интеллектуальной дея-		

№ п/п	Наименование этапов	Содержание выполняемых работ	Перечень документов, разрабатываемых на этапах выполнения ОКР	Срок исполнения (начало – окончание)	Контрактная цена, (тыс. руб.)
		по результатам испытаний. Разработка ПМ приемочных испытаний. Проведение дополнительных патентных исследований	тельности, полученных в ходе выполнения этапа. ПМ приемочных испытаний. Технический акт. Акт сдачи-приемки		
5	Приемочные испытания	Проведение приемочных испытаний. Корректировка программы и программной документации по результатам испытаний. Передача программы в фонд алгоритмов и программ. Проведение дополнительных патентных исследований. Подготовка и согласование предложения по коммерческой реализации программы	Протоколы испытаний по пунктам программы. Перечень результатов интеллектуальной деятельности, полученных в ходе выполнения этапа. ПМ приемочных испытаний. Документ, подтверждающий передачу программы в фонд алгоритмов и программ. Согласованное предложение по коммерческой реализации программы. Технический акт. Акт сдачи-приемки		

От Исполнителя От Заказчика

7.3 Примеры технических заданий

7.3.1 Техническое задание на разработку программного продукта построения графика функций одного аргумента

Необходимо разработать техническое задание на программный продукт, предназначенный для наглядной демонстрации графиков функций одного аргумента $y = f(x)$. Разрабатываемая программа должна рассчитывать таблицу значений и строить график функций на заданном отрезке по заданной формуле и менять шаг аргумента и границы отрезка. Кроме этого, программа должна запоминать введенные формулы.

1. ВВЕДЕНИЕ

Настоящее техническое задание распространяется на разработку программы построения графиков и таблиц значений функций одной переменной, предназначенной для использования в учебных и демонстрационных целях.

В курсе элементарной алгебры тема анализа функций является одной из самых сложных. При изучении данной темы учащиеся должны научиться исследовать и строить графики функций одной переменной, используя все известные характеристические точки функции, включая корни, точки разрыва первого и второго рода и т. д.

Существующее программное обеспечение, которое может решать подобные задачи, является универсальным, например MathCad. Разрабатываемая программа позволит учащимся с минимальными знаниями проверить свои знания при изучении указанной темы.

2. ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ

Программа разрабатывается на основе учебного плана.

3. НАЗНАЧЕНИЕ

Основным назначением программы является помощь учащимся при изучении раздела «Исследование функций одного аргумента» курса элементарной алгебры.

4. ТРЕБОВАНИЯ К ПРОГРАММЕ ИЛИ ПРОГРАММНОМУ ИЗДЕЛИЮ

4.1. Требования к функциональным характеристикам.

4.1.1. Программа должна обеспечивать возможность выполнения следующих функций:

- ввод аналитического представления функции одной переменной и длительное хранение его в системе;
- ввод и изменение интервала определения функции;
- ввод и корректировку шага аргумента;
- построение таблицы значений функции на заданном интервале или изображение графика функции на заданном интервале при условии, что на указанном интервале она не имеет точек разрыва.

4.1.2. Исходные данные:

- аналитическое задание функции;
- интервал определения функции;
- шаг изменения аргумента, определяющий количество точек на интервале.

4.2. Требования к надежности.

4.2.1. Предусмотреть контроль вводимой информации.

4.2.2. Предусмотреть блокировку некорректных действий пользователя при работе с системой.

4.3. Требования к составу и параметрам технических средств.

4.3.1. Система должна работать на IBM-совместимых персональных компьютерах.

4.3.2. Минимальная конфигурация:

- тип процессора – Pentium и выше;
- объем оперативного запоминающего устройства – 32 Мб и более.

4.4. Требования к информационной и программной совместимости: Система должна работать под управлением семейства операционных систем Win 32 (Windows 95, Windows 98, Windows 2000, Windows NT и т. п.).

5. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

5.1. Разрабатываемые программные модули должны быть документированы, т. е. тексты программ должны содержать все необходимые комментарии.

5.2. Разрабатываемая программа должна включать справочную информацию об основных терминах соответствующего раздела математики и подсказки учащимся.

5.3. В состав сопровождающей документации должны входить:

5.3.1. Пояснительная записка на 25–30 листах, содержащая описание разработки.

5.3.2. Руководство пользователя.

7.3.2 Техническое задание на разработку программного продукта учета успеваемости студентов

Разработать техническое задание на создание системы «Учет успеваемости студентов». Система предназначена для оперативного учета успеваемости студентов в сессию деканом, заместителями декана по курсам и сотрудниками деканата. Сведения об успеваемости студентов должны храниться в течение всего срока их обучения и использоваться при составлении справок о прослушанных курсах и приложений к диплому.

1. ВВЕДЕНИЕ

Настоящее техническое задание распространяется на разработку системы учета успеваемости студентов, предназначенной для сбора и хранения информации о ходе сдачи экзаменационной сессии. Предполагается, что использовать данную систему будут сотрудники деканата, декан и его заместители. Во время сессии необходимо получение оперативной информации о ходе ее сдачи студентами, однако выполнение такого контроля вручную требует значительного времени. Автоматизированная система учета успеваемости позволит улучшить качество контроля сдачи сессии со стороны куратора и деканата и обеспечит получение сведений о динамике работы каждого студента, группы и курса в целом. Кроме того, хранение информации о сдаче сессий в течение всего времени обучения позволит осуществлять автоматическую генерацию справок о прослушанных курсах и приложений к диплому выпускника!

2. ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ

Система разрабатывается на основании учебного плана.

3. НАЗНАЧЕНИЕ

Система предназначена для хранения и обработки сведений об успеваемости студентов учебных групп факультета в течение всего срока обучения. Обработанные сведения об успеваемости студентов могут быть использованы для оценки успеваемости каждого студента, группы, курса и факультета в целом.

4. ТРЕБОВАНИЯ К ПРОГРАММЕ ИЛИ ПРОГРАММНОМУ ИЗДЕЛИЮ

4.1. Требования к функциональным характеристикам.

4.1.1. Система должна обеспечивать возможность выполнения следующих функций:

- инициализацию системы (ввод списков групп, перечней изучаемых дисциплин в соответствии с учебными планами и т. п.);

- ввод и коррекцию текущей информации о ходе сдачи сессии конкретными студентами;
- хранение информации об успеваемости в течение времени обучения студента;
- получение сведений о текущем состоянии сдачи сессии студентами.

4.1.2. Исходные данные:

- списки студентов учебных групп;
- учебные планы кафедр – перечень предметов и контрольных мероприятий по каждому предмету;
- расписания сессий;
- текущие сведения о сдаче сессии каждым студентом.

4.1.3. Результаты:

- итоги сдачи сессии конкретным студентом;
- итоги сдачи сессии студентами конкретной группы;
- процент успеваемости по всем студентам группы при сдаче конкретного предмета в целом на текущий момент;
- проценты успеваемости по всем группам специальности на текущий момент;
- проценты успеваемости по всем группам курса на текущий момент;
- проценты успеваемости по всем курсам и в целом по факультету на текущий момент;
- список задолжников группы на текущий момент;
- список задолжников курса на текущий момент.

4.2. Требования к надежности.

4.2.1. Предусмотреть контроль вводимой информации.

4.2.2. Предусмотреть блокировку некорректных действий пользователя при работе с системой.

4.2.3. Обеспечить целостность хранимой информации.

4.3. Требования к составу и параметрам технических средств.

4.3.1. Система должна работать на IBM совместимых персональных компьютерах.

4.3.2. Минимальная конфигурация:

- тип процессора – Pentium и выше;
- объем оперативного запоминающего устройства – 32 Мб и более.

4.4. Требования к информационной и программной совместимости: Система должна работать под управлением семейства операционных систем Win 32 (Windows 95, Windows 98, Windows 2000, Windows NT и т. п.).

5. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

5.1. Разрабатываемые программные модули должны быть самодокументированы, т. е. тексты программ должны содержать все необходимые комментарии.

5.2. Программная система должна включать справочную информацию о работе и подсказки пользователю.

5.3. В состав сопровождающей документации должны входить:

5.3.1. Пояснительная записка на 25–30 листах, содержащая описание разработки.

5.3.2. Руководство системного программиста.

5.3.3. Руководство пользователя.

5.3.4. Графическая часть на трех листах формата А1:

5.3.4.1. Схема структурная программной системы.

5.3.4.2. Диаграмма компонентов данных.

5.3.4.3. Формы интерфейса пользователя.

7.3.3 Пример разработки и внедрения информационно-аналитической системы

Согласно ГОСТ 34.601–90 «Автоматизированные системы. Стадии создания» выделяют следующие основные стадии создания и этапы разработки автоматизированной системы (АС):

- Формирование требований к АС.
- Разработка концепции АС.
- Техническое задание.
- Эскизный проект.
- Технический проект.
- Рабочая документация.
- Ввод в действие.
- Сопровождение АС.

Как правило, этапы стадий формирования требований к автоматизированной системе объединяют с этапами разработки технического задания, а этапы разработки концепции – с этапами эскизного проектирования. Также к приведенным стадиям создания АС добавляют стадию подготовки к началу проекта.

Ниже представлены стадии создания информационно-аналитической системы – корпоративного хранилища данных. Также дана оценка длительности каждого из этапов внедрения системы.

Стадия I. Организация проекта

Этапы работ	Результат	Дней
Заключение контракта	Контракт на разработку системы заключен	10
Согласование процедур управления	Процедуры управления проектом и устав проекта согласованы	5
Сбор команды проекта	Команда проекта сформирована	5

Стадия II. Формирование требований и разработка технического задания

Этапы работ	Результат	Дней
Системно-аналитическое обследование объекта автоматизации	Проведены интервью с функциональными и IT-специалистами Заказчика. Собрана информация: о требуемых отчетах, показателя, измерениях; возможных системах-источниках данных; системном ландшафте; сетевом окружении	20
Анализ и обработка полученной информации	Сформированы: альбом показателей; альбом измерений и иерархий; альбом отчетности; определены источники данных для каждого показателя	20
Разработка концептуальной модели данных	Концептуальная модель данных	15
Разработка технического задания	Техническое задание и приложения к нему	20
Согласование и утверждение	Согласованное и утвержденное техническое задание	10

Стадия III. Эскизный проект

Этапы работ	Результат	Дней
Определение общей функциональной и технической архитектур	Общее описание функциональной и технической архитектур	10
Разработка логической модели данных	Логическая модель данных. Определен состав сущностей области постоянного хранения (System of Record, Summary Area, Data Marts). Окончательно сформирован состав реализуемых показателей, измерений и отчетов	20
Обоснование выбора программного обеспечения и технической инфраструктуры	ПО выбрано. Утвержден план формирования необходимой технической инфраструктуры	10
Разработка предварительного регламента	Предварительные регламенты взаимодействия согласованы с разработчиками	20

Этапы работ	Результат	Дней
взаимодействия информационных систем	и администраторами систем-источников данных. Определен состав сущностей области временного хранения (Staging Area) и области обмена данными (Data Exchange Interface)	
Эскизное проектирование процессов ETL	Общее описание процессов извлечения данных, алгоритмов трансформации, загрузки и агрегации данных	10
Эскизное проектирование интерфейсов пользователя	Общее описание интерфейсов ввода и предоставления данных	5
Оформление эскизного проекта	Пояснительная записка к эскизному проекту	15
Согласование и утверждение	Согласованная и утвержденная пояснительная записка к эскизному проекту	5

Стадия IV. Технический проект

Этапы работ	Результат	Дней
Определение функциональной и технической архитектур	Описание функциональной архитектуры (включая описание каждой функции, задачи, методов реализации). Описание технической архитектуры (включая описание размещения технических средств, условий эксплуатации, режима функционирования, организации резервного копирования и т. д.)	15
Формирование плана развертывание системного ландшафта	План развертывания системного ландшафта	10
Разработка физической модели данных	Физическая модель данных – описание таблиц БД, индексов, секций и других объектов БД	10
Разработка и согласование регламентов взаимодействия информационных систем	Согласованный и утвержденный набор регламентов взаимодействия, включающих описание интерфейсов, периодичности и т. п.	10
Проектирование процессов ETL	Описание процессов извлечения данных, алгоритмов трансформации и обеспечения качества данных, процессов загрузки и агрегации данных	20

Этапы работ	Результат	Дней
Проектирование интерфейсов пользователя	Описание бизнес-слоя данных, интерфейсов ввода и предоставления данных, разграничения прав доступа	15
Оформление технического проекта	Пояснительная записка к техническому проекту	20
Согласование и утверждение	Согласованная и утвержденная пояснительная записка к техническому проекту	5

Стадия V. Рабочая документация

Этапы работ	Результат	Дней
Разработка рабочей документации на систему и на её части	Разработаны следующие документы: Ведомость эксплуатационных документов. Ведомость машинных носителей информации. Паспорт. Общее описание системы. Технологическая инструкция. Руководство пользователя. Описание технологического процесса обработки данных (включая телеобработку). Инструкция по формированию и ведению базы данных (набора данных). Состав выходных данных (сообщений). Каталог базы данных. Программа и методика испытаний (ПИМ). Спецификация. Описание программ. Текст программ	40
Разработка или адаптация программ	Развернуты экземпляры БД. Созданы необходимые объекты БД. Разработаны процессы ETL и процессы обеспечения качества данных. Выставлено расписание запуска процессов.	60

Этапы работ	Результат	Дней
	Реализованы дополнительные приложения. Реализованы витрины данных и отчетность. Настроены профили пользователей и прав доступа	
Согласование и утверждение	Согласованная и утвержденная рабочая документация	15

Стадия VI. Ввод в действие

Этапы работ	Результат	Дней
Подготовка объекта автоматизации к вводу системы в действие	Создание у Заказчика службы сопровождения системы (при необходимости)	10
Подготовка персонала	Обучение пользователей и администраторов системы	10
Строительно-монтажные работы	Оборудование смонтировано в выделенном для этого помещении и подключено к каналам передачи данных	20
Комплектация системы поставляемыми изделиями	Закупка и завоз необходимого оборудования и программного обеспечения. Для снятия рисков поставки данный этап обычно выполняется на предыдущей стадии	20–100
Пусконаладочные работы	Прошла наладка технических и программных средств. ПО системы перенесено в зону тестирования/промышленной эксплуатации. Настроена система резервного копирования. Проведена загрузка исторических данных в систему. Запущены процессы извлечения данных из систем-источников	20
Проведение предварительных испытаний	Испытания системы на работоспособность и соответствие техническому заданию в соответствии с ПИМ проведены. Устранение неисправностей и внесение изменений в документацию в соответствии с протоколом испытаний проведено.	10

Этапы работ	Результат	Дней
	Акт приёмки системы в опытную эксплуатацию оформлен и подписан	
Проведение опытной эксплуатации	Опытная эксплуатация проведена. Устранение неисправностей, доработка ПО и дополнительная наладка технических средств проведены	10–60
Проведение приёмочных испытаний		5
Завершение работ		5

Стадия VII. Сопровождение

Этапы работ	Результат	Дней
Выполнение работ в соответствии с гарантийными обязательствами	Выявленные недостатки системы устранены	365
Послегарантийное обслуживание	Система работает стабильно, без сбоев. Выявленные недостатки устранены	–

8 ПРИНЦИПЫ ПРОЕКТИРОВАНИЯ ПРОГРАММНЫХ СИСТЕМ

8.1 Общие сведения о проектировании программных систем

Одна из основных проблем при создании больших и сложных систем, в том числе ПО, – это проблема сложности. Сложная программная система должна быть разделена на небольшие подсистемы, каждую из которых можно разрабатывать независимо (в какой-то степени) от других. Декомпозиция является главным способом преодоления сложности разработки ПО.

Принципы декомпозиции:

– слабая связанность – low coupling (количество связей между отдельными подсистемами должно быть минимальным);

– сильное сцепление – high cohesion (связность отдельных частей внутри каждой подсистемы должна быть максимальной).

Структура ПО должна удовлетворять следующим требованиям:

– каждая подсистема должна инкапсулировать свое содержимое (скрывать его от других подсистем);

– каждая подсистема должна иметь четко определенный интерфейс с другими подсистемами.

Два основных подхода к декомпозиции систем: функционально-модульный, основанный на функциональной декомпозиции, при которой структура системы описывается в терминах иерархии ее функций и иерархии структур данных; объектно-ориентированный, использующий объектную декомпозицию, при которой структура системы описывается в терминах объектов и связей между ними, а поведение системы описывается в терминах обмена сообщениями между объектами. Подходы имеют много общего. Достоинством второго подхода является то, что есть единая иерархия и нет необходимости отслеживать соответствие между двумя иерархиями функционально-модульного подхода.

В рамках обоих подходов используются наглядные графические модели (схемы и диаграммы) для описания архитектуры ПО с различных точек зрения. Примеры: блок-схемы, диаграммы «сущность – связь», диаграммы классов. Модель ПО – это формализованное описание системы ПО на определенном уровне абстракции. Каждая модель описывает конкретный аспект системы, использует набор диаграмм или формальных описаний и документов заданного формата, а также отражает точку зрения и является объектом деятельности различных людей с конкретными интересами, ролями или задачами. Модели служат полезным инструментом анализа проблем, обмена информацией между всеми заинтересованными сторонами, проектирования ПО. Моделирование способствует более полному усвоению требований, улучшению качества системы и повышению степени ее управляемости.

Визуальное моделирование – это способ восприятия проблем с помощью зримых абстракций, воспроизводящих понятия и объекты реального мира. Моделирование осуществляется при помощи языка моделирования, который включает в себя: элементы модели; нотацию (систему обозначений); руководство по использованию.

Моделирование не является целью разработки ПО. Диаграммы – это лишь наглядные изображения. Причины, побуждающие прибегать к их использованию:

1. Графические модели помогают получить общее представление о системе, сказать о том, какого рода абстракции существуют в системе и какие ее части нуждаются в дальнейшем уточнении.

2. Графические модели образуют внешнее представление системы и объясняют, что эта система будет делать, тем самым облегчают общение с заказчиком.

3. Графические модели облегчают изучение методов проектирования, в частности объектно-ориентированных методов.

В процессе создания ПО используются следующие виды моделей:

модели деятельности организации (или модели бизнес-процессов):

– модели «AS-IS» («как есть»), отражающие существующее положение;

– модели «AS-TO-BE» («как должно быть»), отражающие представление о новых процессах и технологиях работы организации;

модели проектируемого ПО, которые строятся на основе модели «AS-TO-BE», уточняются и детализируются до необходимого уровня.

Состав моделей, используемых в каждом конкретном проекте, и степень их детальности в общем случае зависят от следующих факторов: сложности проектируемой системы; необходимой полноты ее описания; знаний и навыков участников проекта; времени, отведенного на проектирование.

Для облегчения труда разработчиков и автоматизированного выполнения некоторых рутинных действий используются CASE-средства (Computer Aided Software Engineering). В настоящее время CASE-средства обеспечивают поддержку большинства процессов жизненного цикла ПО, что позволяет говорить о CASE-технологиях разработки ПО. CASE-технология – это совокупность методов проектирования ПО и инструментальных средств для моделирования предметной области, анализа моделей на всех стадиях ЖЦ ПО и разработки ПО.

8.2 Понятие архитектуры программного обеспечения

Архитектура ПО – набор ключевых правил, определяющих организацию системы:

– совокупность структурных элементов системы и связей между ними;

– поведение элементов системы в процессе их взаимодействия;

– иерархию подсистем, объединяющих структурные элементы.

Архитектура ПО многомерна, поскольку различные специалисты работают с её различными аспектами. Различные представления архитектуры служат различным целям:

– представление функциональных возможностей ПО;

– представление логической организации ПО;

– представление физической структуры программных компонент, входящих в состав ПО;

– представление структуры потоков управления и аспектов параллельной работы ПО;

– описание физического размещения компонент ПО по узлам вычислительной системы.

Архитектурное представление – это модель системы с определенной точки зрения, в которой отражены лишь существенные аспекты и опущено все, что несущественно при данном взгляде на систему. Архитектурно значимый элемент – это элемент, значительно влияющий на структуру системы, её функциональность, производительность, надежность, защищенность, возможность развития. Подсистемы, их интерфейсы, процессы и потоки управления являются архитектурно значимыми элементами.

Существуют стандартные архитектурные модели, такие как OSI/ISO. Для удобства повторного использования и передачи знаний между разработчиками некоторые архитектурные решения оформляются в виде образцов или паттернов проектирования. Паттерн – это описание проблемы, возникающей при проектировании архитектуры ПО, и способа решения этой проблемы. Каждый паттерн содержит: имя паттерна; описание проблемной области и ситуаций, в которых можно использовать паттерн; шаблон проектного решения; описание последствий применения паттерна.

8.3 Объектная модель

В основе объектно-ориентированного подхода (ООП) лежит объектная декомпозиция, при этом статическая структура ПО описывается в терминах объектов и связей между ними, а динамический аспект ПО описывается в терминах обмена сообщениями между объектами. Каждый объект системы обладает своим собственным поведением, моделирующим поведение объекта реального мира.

Объектная модель является естественным способом представления реального мира. Она является концептуальной основой ООП. Основными принципами ее построения являются:

– абстрагирование;

– инкапсуляция;

– модульность;

– иерархия.

Абстрагирование – это выделение наиболее важных, существенных характеристик некоторого объекта, которые отличают его от всех других видов объектов, и игнорирование менее важных или незначительных деталей. Абстрагирование позволяет управлять сложностью системы, концентрируясь на существенных свойствах объекта. Выбор правильного набора абстракций для заданной предметной области представляет собой главную задачу объектно-ориентированного проектирования. Объекты и классы – основные абстракции предметной области.

Инкапсуляция – локализация свойств и поведения в рамках единственной абстракции (рассматриваемой как «черный ящик»), скрывающей реализацию за общедоступным интерфейсом. Инкапсуляция – это отделение внутреннего устройства объекта от его внешнего поведения. Объектный подход предполагает, что внутренние ресурсы объекта скрыты от внешней среды. Абстрагирование и инкапсуляция являются взаимодополняющими принципами.

Модульность – это декомпозиция системы в виде набора внутренне сильно сцепленных, но слабо связанных между собой подсистем (модулей). Модульность снижает сложность системы, позволяя выполнять независимую разработку отдельных модулей.

Иерархия – это упорядоченная система абстракций, задающая их расположение по уровням. Основными видами иерархических структур сложных систем являются структура классов и структура объектов. Иерархия классов строится по наследованию, а иерархия объектов – по агрегации.

К основным понятиям объектно-ориентированного подхода (элементам объектной модели) относятся: объект; класс; атрибут; операция; полиморфизм; наследование; компонент; связь.

Объект – это сущность предметной области или программной системы, имеющий четко определяемое поведение. Любой объект обладает состоянием, поведением и индивидуальностью. Состояние объекта определяется значениями его свойств (атрибутов) и связями с другими объектами, оно может меняться со временем. Поведение определяет действия объекта и его реакцию на запросы от других объектов. Поведение представляется с помощью набора сообщений, воспринимаемых объектом (операций, которые может выполнять объект). Индивидуальность – это свойства объекта, отличающие его от всех других объектов.

Структура и поведение схожих объектов определяют общий для них класс. Класс – это множество объектов, связанных общностью свойств, поведения, связей и семантики. Любой объект является экземпляром класса. Определение классов и объектов – одна из самых сложных задач объектно-ориентированного проектирования.

Атрибут – поименованное свойство класса, определяющее диапазон допустимых значений, которые могут принимать экземпляры данного свойства. Атрибуты могут быть скрыты от других классов, это определяет видимость

атрибута: public (общий, открытый); private (закрытый, секретный); protected (защищенный).

Определенное воздействие одного объекта на другой с целью вызвать соответствующую реакцию называется операцией или посылкой сообщения. Операция – это реализация услуги, которую можно запросить у любого объекта данного класса. Операции реализуют связанное с классом поведение, его обязанности. Описание операции включает четыре части: имя; список параметров; тип возвращаемого значения; видимость.

Понятие полиморфизма может быть интерпретировано, как способность класса принадлежать более чем одному типу. Полиморфизм – это способность скрывать множество различных реализаций под единственным общим интерфейсом. Интерфейс – это совокупность операций, определяющих набор услуг класса или компонента. Интерфейс не определяет внутреннюю структуру, все его операции открыты.

Наследование – это построение новых классов, на основе существующих с возможностью добавления или переопределения свойств (атрибутов) и поведения (операций).

Компонент – это относительно независимая и замещаемая часть системы, выполняющая четко определенную функцию в контексте заданной архитектуры. Виды компонентов: компонент исходного кода; компонент времени выполнения; исполняемый компонент.

Между элементами объектной модели существуют различные виды связей:

– ассоциация – это семантическая связь между классами;

– агрегация – более сильный тип связи между целым и его частями;

– зависимость – связь между двумя элементами модели, при которой изменения в спецификации одного элемента могут повлечь за собой изменения в другом элементе;

– обобщение – связь «тип – подтип».

Связи характеризуются: направлением; именем и ролевыми именами участников связи; мощностью.

8.4 Анализ и проектирование программного обеспечения

Целью объектно-ориентированного анализа является трансформация функциональных требований к ПО в предварительный системный проект и создание стабильной основы архитектуры системы.

Объектно-ориентированный анализ включает два вида деятельности:

– архитектурный анализ;

– анализ вариантов использования.

Архитектурный анализ выполняется архитектором системы и включает в себя:

– утверждение общих стандартов (соглашений) моделирования и документирования системы;

– предварительное выявление архитектурных механизмов (механизмов анализа);

– формирование набора основных абстракций предметной области (классов анализа);

– формирование начального представления архитектурных уровней.

Соглашения моделирования определяют:

– используемые диаграммы и элементы модели;

– правила их применения;

– соглашения по именованию элементов модели;

– организацию модели (пакеты).

Архитектурные механизмы отражают нефункциональные требования к системе (надежность, безопасность, хранение данных в конкретной среде, интерфейсы с внешними системами и т. д.) и их реализацию в архитектуре

системы. Архитектурные механизмы представляют собой набор типовых решений, или образцов, принятых в качестве стандарта данного проекта.

Идентификация основных абстракций заключается в определении набора классов системы (классов анализа) на основе описания предметной области и спецификации требований к системе (в частности, глоссария проекта).

Архитектурные уровни образуют иерархию уровней представления любой крупной системы. Почти в любой системе можно выделить следующие уровни:

– прикладной, содержащий набор компонентов, реализующих основную функциональность системы;

– бизнес-уровень, включающий набор компонентов, специфичных для конкретной предметной области;

– промежуточный (middleware), куда входят платформо-независимые сервисы;

– системный, содержащий компоненты вычислительной и сетевой инфраструктуры.

Анализ вариантов использования выполняется проектировщиками и включает в себя:

– идентификацию классов, участвующих в реализации потоков событий;

– определение обязанностей классов;

– определение атрибутов и ассоциаций классов;

– унификацию классов анализа.

В потоках событий варианта использования выявляются классы трех типов:

– граничные классы, являющиеся посредниками при взаимодействии с внешними объектами;

– классы-сущности, представляющие собой основные абстракции (понятия) разрабатываемой системы;

– управляющие классы, обеспечивающие координацию поведения объектов в системе.

Классы анализа отражают функциональные требования к системе и моделируют объекты предметной области. Совокупность классов анализа представляет собой начальную концептуальную модель системы.

В процессе анализа конкретного варианта использования в первую очередь строится диаграмма последовательности, описывающая основной поток событий и его подчиненные потоки. Для каждого альтернативного потока событий строится отдельная диаграмма последовательности.

Обязанности каждого класса определяются исходя из сообщений на диаграммах взаимодействия, и документируются в классах в виде «операций анализа». В процессе проектирования каждая «операция анализа» преобразуется в одну или более операций класса, которые в дальнейшем будут реализованы в коде системы.

При построении диаграмм взаимодействия возникают проблемы правильного распределения обязанностей между классами. Для их решения существует ряд образцов: Information Expert, Creator, Low Coupling, High Cohesion.

Атрибуты классов анализа определяются исходя из знаний о предметной области, требований к системе, глоссария и бизнес-модели. В процессе анализа атрибуты определяются только для классов-сущностей. Связи между классами определяются в два этапа. Начальный набор связей определяется на основе анализа кооперативных диаграмм. Если два объекта обмениваются сообщениями, между ними должна быть ассоциация. На втором этапе исходя из знаний о предметной области создаются ассоциации между классами-сущностями.

Унификация классов анализа заключается в проверке выполнения следующих условий:

– имя и описание каждого класса анализа должно отражать сущность его роли в системе;

– классы с одинаковым поведением, или представляющие одно и то же явление, должны объединяться;

– классы-сущности с одинаковыми атрибутами должны объединяться (даже если их поведение отличается).

По результатам унификации классов должны быть модифицированы описания вариантов использования.

Проектирование архитектуры системы выполняется архитектором системы и включает в себя:

– идентификацию архитектурных решений и механизмов, необходимых для проектирования системы;

– анализ взаимодействий между классами анализа, выявление подсистем и интерфейсов;

– формирование архитектурных уровней;

– проектирование структуры потоков управления;

– проектирование конфигурации системы.

В процессе проектирования определяются детали реализации архитектурных механизмов, обозначенных в процессе анализа. Например, уточняется способ хранения данных, реализация дублирования для повышения надежности системы и т. п.

Декомпозиция системы на подсистемы реализует принцип модульности. Первым действием архитектора при выявлении подсистем является преобразование классов анализа в проектные классы. По каждому классу анализа принимается одно из двух решений:

– класс анализа отображается в проектный класс, если он простой или представляет единственную логическую абстракцию;

– сложный класс анализа может быть разбит на несколько классов, преобразован в пакет или в подсистему.

Несколько классов могут быть объединены в подсистему если:

– классы имеют функциональную связь (участвуют в реализации варианта использования и взаимодействуют только друг с другом);

– совокупность классов реализует функциональность, которая может быть удалена из системы или заменена на альтернативную;

– классы сильно связаны;

– классы размещены на одном узле вычислительной сети.

Примеры возможных подсистем: подсистема безопасности, защиты данных, архивирования; подсистема пользовательского интерфейса или интерфейса с внешними системами; коммуникационное ПО, доступ к базам данных.

При создании подсистем в модели выполняются следующие преобразования:

– объединяемые классы помещаются в специальный пакет с именем подсистемы и стереотипом <<subsystem>>;

– спецификации операций классов, образующих подсистему, выносятся в интерфейс подсистемы – класс со стереотипом <<Interface>>;

– характер использования операций интерфейса и порядок их выполнения документируется с помощью диаграмм взаимодействия, которые вместе с диаграммой классов подсистемы объединяются в кооперацию с именем интерфейса и стереотипом <<interface realization>>;

– в подсистеме создается класс-посредник со стереотипом <<subsystem proxy>>, управляющий реализацией операций интерфейса.

В процессе анализа было принято предварительное решение о выделении архитектурных уровней. В процессе проектирования все элементы системы должны быть распределены по данным уровням. С точки зрения модели это означает распределение проектных классов по пакетам, соответствующим архитектурным уровням.

Проектирование структуры потоков управления выполняется при наличии в системе параллельных процессов (параллелизма). Цель проектирования – выявление существующих в системе процессов, характера их взаимодействия, создания, уничтожения и отображения в среду реализации.

Процесс – это ресурсоемкий поток управления, который может выполняться параллельно с другими потоками. Он выполняется в независимом адресном пространстве и в случае высокой сложности может разделяться на два или более потоков.

Поток (нить) – это облегченный поток управления, который может выполняться параллельно с другими потоками в рамках одного и того же процесса в общем адресном пространстве.

Реализация процессов и потоков обеспечивается средствами операционной системы.

Для моделирования структуры потоков управления используются так называемые активные классы – классы со стереотипами <<process>> и <<thread>>. Активный класс владеет собственным процессом или потоком и может инициировать управляющие воздействия. Связи между процессами моделируются как зависимости. Потоки могут существовать только внутри процессов, поэтому связи между процессами и потоками моделируются как композиции.

Если создаваемая система является распределенной, то необходимо спроектировать ее конфигурацию в вычислительной среде, т. е. описать вычислительные ресурсы, коммуникации между ними и использование ресурсов различными системными процессами.

Распределенная сетевая конфигурация системы моделируется с помощью диаграммы размещения.

Распределение процессов, составляющих структуру потоков управления, по узлам сети производится с учетом следующих факторов:

– используемые образцы распределения (трехзвенная архитектура клиент-сервер, «толстый клиент», «тонкий клиент», «точка-точка» и т. д.);

– время отклика;

– минимизация сетевого трафика;

– мощность узла;

– надежность оборудования и коммуникаций.

Проектирование элементов системы выполняется проектировщиками и включает:

– уточнение описания вариантов использования;

– проектирование классов;

– проектирование баз данных.

Уточнение описания вариантов использования заключается в модификации их диаграмм взаимодействия и диаграмм классов с учетом вновь появившихся на шаге проектирования классов и подсистем, а также проектных механизмов.

Проектирование классов включает следующие действия:

– детализация проектных классов;

– уточнение операций и атрибутов;

– моделирование состояний для классов;

– уточнение связей между классами.

Каждый граничный класс преобразуется в некоторый набор классов, в зависимости от своего назначения. Это может быть набор элементов пользовательского интерфейса, зависящий от возможностей среды разработки, или набор классов, реализующий системный или аппаратный интерфейс.

Классы-сущности с учетом соображений производительности и защиты данных могут разбиваться на ряд классов. Основанием для разбиения является наличие в классе атрибутов с различной частотой использования или видимостью. Такие атрибуты, как правило, выделяются в отдельные классы.

Что касается управляющих классов, то классы, реализующие простую передачу информации от граничных классов к сущностям, могут быть удалены. Сохраняются классы, выполняющие существенную работу по управлению потоками событий (управление транзакциями, распределенная обработка и т. д.).

Полученные в результате уточнения классы подлежат непосредственной реализации в коде системы.

Обязанности классов, определенные в процессе анализа и документированные в виде «операций анализа», преобразуются в операции, которые будут реализованы в коде. При этом:

– каждой операции присваивается краткое имя, характеризующее ее результат;

– определяется полная сигнатура операции;

– создается краткое описание операции, включая смысл всех ее параметров;

– определяется видимость операции: public, private или protected;

– определяется область действия операции: операция объекта или операция класса.

Уточнение атрибутов классов заключается в следующем:

– задается его тип атрибута и значение по умолчанию (необязательно);

– задается видимость атрибутов: public, private или protected;

– при необходимости определяются производные (вычисляемые) атрибуты.

Если в системе присутствуют объекты со сложным поведением, то строят диаграммы состояний. Построение диаграмм состояний может оказать следующее воздействие на описание классов:

– события могут отображаться в операции класса;

– особенности конкретных состояний могут повлиять на детали выполнения операций;

– описание состояний и переходов может помочь при определении атрибутов класса.

В процессе проектирования связи между классами подлежат уточнению. Ассоциации между граничными и управляющими классами преобразуются в зависимости. Агрегации, обладающие свойствами композиции, преобразуются в связи композиции. Связи обобщения могут преобразовываться в ситуациях с так называемой метаморфозой подтипов, когда объект суперкласса может менять свой подтип.

Проектирование баз данных производится, если используется реляционная БД, при этом классы-сущности объектной модели отображаются в таблицы реляционной БД. Совокупность таблиц и связей между ними может быть представлена в виде диаграммы классов, которая, по существу, является диаграммой «сущность – связь». Набор правил, применяемых при отображении классов в таблицы БД, таков:

1. Каждая простая сущность, не являющаяся подтипом и не имеющая подтипов, превращается в таблицу. Имя сущности становится именем таблицы.

2. Каждый атрибут становится возможным столбцом с тем же именем; может выбираться более точный формат.

3. Уникальный идентификатор сущности превращается в первичный ключ таблицы.

4. Если тип бинарной связи между сущностями – один-к-одному и класс принадлежности обеих сущностей является обязательным, то из двух связанных сущностей формируется одна таблица.

5. Если тип бинарной связи – один-к-одному и класс принадлежности одной сущности является обязательным, а другой – необязательным, то формируются две таблицы и идентификатор сущности, для которой класс принадлежности является необязательным, добавляется в качестве атрибута

в таблицу, выделенную для сущности с обязательным классом принадлежности.

6. Если тип бинарной связи – один-к-одному и класс принадлежности ни одной сущности не является обязательным, то формируются три таблицы: по одной для каждой сущности и одна для связи.

7. Если тип бинарной связи – один-ко-многим и класс принадлежности сущности с мощностью «n» является обязательным, то формируются две таблицы, при этом идентификатор каждой сущности должен служить первичным ключом соответствующей таблицы. Кроме того, идентификатор сущности с мощностью «1» добавляется в качестве атрибута в таблицу, выделенную для сущности с мощностью «n».

8. Если тип бинарной связи – один-ко-многим и класс принадлежности сущности с мощностью «n» является необязательным, см. правило 6.

9. Если тип бинарной связи – многие-ко-многим, см. правило 6.

10. Для представления N-арной связи требуется N+1 таблица. Например, в случае тернарной связи формируются четыре таблицы, по одной для каждой сущности и одна для связи. Таблица связи будет иметь среди своих атрибутов ключи от каждой сущности.

11. Для связи «супертип-подтип» возможны два способа преобразования:

а) все подтипы в одной таблице;

б) для каждого подтипа – отдельная таблица.

При применении способа (а) для супертипа создается таблица, а для подтипов могут создаваться представления (view). В таблицу добавляется по крайней мере один столбец, содержащий код типа.

При использовании способа (б) для каждого подтипа супертип воссоздается с помощью операции объединения (UNION) – из всех таблиц подтипов выбираются общие столбцы – столбцы супертипа.

8.5 Технологические аспекты проектирования программного обеспечения

В соответствии со стандартом ISO/IEC 12207 все процессы жизненного цикла программного обеспечения разделены на три группы:

– основные процессы (приобретение, поставка, разработка, эксплуатация, сопровождение);

– вспомогательные процессы, обеспечивающие выполнение основных процессов (документирование, управление конфигурацией, обеспечение качества, верификация, аттестация, совместная оценка, аудит, разрешение проблем);

– организационные процессы (управление, создание инфраструктуры, усовершенствование, обучение).

1. СОДЕРЖАНИЕ ОСНОВНЫХ ПРОЦЕССОВ

1.1. Процесс приобретения состоит из действий и задач заказчика, приобретающего программное обеспечение.

1.2. Процесс поставки охватывает действия и задачи, выполняемые поставщиком, который снабжает заказчика программным продуктом или услугой.

1.3. Процесс разработки предусматривает действия и задачи, выполняемые разработчиком и охватывает работы по созданию программного обеспечения и его компонентов в соответствии с заданными требованиями, включая оформление проектной и эксплуатационной документации, подготовку материалов, необходимых для проверки работоспособности и соответствующего качества программных продуктов, материалов, необходимых для организации обучения персонала и т. д.

1.4. Процесс эксплуатации охватывает действия и задачи оператора – организации, эксплуатирующей систему.

1.5. Процесс сопровождения предусматривает действия и задачи, выполняемые сопровождающей организацией (службой сопровождения). Процесс сопровождения охватывает подготовительную работу, анализ проблем и запросов на модификацию программного обеспечения, модификацию программного обеспечения, проверку и приемку, перенос программного обеспечения в другую среду, снятие программного обеспечения с эксплуатации.

2. СОДЕРЖАНИЕ ВСПОМОГАТЕЛЬНЫХ ПРОЦЕССОВ

2.1. Процесс документирования предусматривает формализованное описание информации, созданной в течение жизненного цикла программного обеспечения.

2.2. Процесс управления конфигурацией предполагает применение административных и технических процедур на всем протяжении жизненного цикла программного обеспечения для определения состояния компонентов программного обеспечения в системе управления модификациями программного обеспечения, описания и подготовки отчетов о состоянии компонентов программного обеспечения и запросов на модификацию, обеспечения полноты, совместимости и корректности компонентов программного обеспечения, управления хранением и поставкой программного обеспечения.

2.3. Процесс обеспечения качества обеспечивает соответствующие гарантии того, что программное обеспечение и процессы его жизненного цикла соответствуют заданным требованиям и утвержденным планам.

2.4. Процесс верификации состоит в определении того, что программные продукты, являющиеся результатами некоторого действия, полностью удовлетворяют требованиям или условиям, обусловленным предшествующими действиями.

2.5. Процесс аттестации предусматривает определение полноты соответствия заданных требований и созданной системы или программного продукта их конкретному функциональному назначению.

2.6. Процесс совместной оценки предназначен для оценки состояния работ по проекту и программного обеспечения, создаваемого при выполнении данных работ (действий). Он сосредоточен в основном на контроле планирования и управления ресурсами, персоналом, аппаратурой и инструментальными средствами проекта.

2.7. Процесс аудита представляет собой определение соответствия требованиям, планам и условиям договора.

2.8. Процесс разрешения проблем предусматривает анализ и решение проблем (включая обнаруженные несоответствия) независимо от их происхождения или источника, которые обнаружены в ходе разработки, эксплуатации, сопровождения или других процессов.

3. СОДЕРЖАНИЕ ОРГАНИЗАЦИОННЫХ ПРОЦЕССОВ

3.1. Процесс управления состоит из действий и задач, которые могут выполняться любой стороной, управляющей своими процессами. Данная сторона (менеджер) отвечает за управление выпуском продукта, управление проектом и управление задачами соответствующих процессов, таких, как приобретение, поставка, разработка, эксплуатация, сопровождение и др.

3.2. Процесс создания инфраструктуры охватывает выбор и поддержку (сопровождение) технологии, стандартов и инструментальных средств, выбор и установку аппаратных и программных средств, используемых для разработки, эксплуатации или сопровождения программного обеспечения.

3.3. Процесс усовершенствования предусматривает оценку, измерение, контроль и усовершенствование процессов жизненного цикла программного обеспечения.

3.4. Процесс обучения охватывает первоначальное обучение и последующее постоянное повышение квалификации персонала. Должны быть разработаны и представлены методические материалы, необходимые для обучения пользователей в соответствии с учебным планом.

Технология проектирования программного обеспечения определяется как совокупность технологических операций проектирования (рис. 8.1) в их последовательности и взаимосвязи, приводящая к разработке проекта программного обеспечения.



Рис. 8.1 – Контекст технологической
операции проектирования

Современная технология проектирования программного обеспечения информационных систем должна обеспечивать:

– соответствие стандарту ISO/IEC 12207 (поддержка всех процессов жизненного цикла программного обеспечения);

– гарантированное достижение целей разработки информационных систем в рамках установленного бюджета, с заданным качеством и в установленное время;

– возможность декомпозиции проекта на составные части, разрабатываемые группами исполнителей ограниченной численности (3–7 человек), с последующей интеграцией составных частей;

– минимальное время получения работоспособного программного обеспечения информационных систем.

8.6 Объектная декомпозиция

Объектной декомпозицией называют процесс представления предметной области задачи в виде совокупности функциональных элементов (объектов), обменивающихся в процессе выполнения программы входными воздействиями (сообщениями). Иначе можно дать определение так: процесс представления предметной области задачи в виде совокупности объектов, обменивающихся сообщениями, называется объектной декомпозицией. При объектной декомпозиции между объектами устанавливаются отношения:

1) использования – первый объект (активный) передает сообщение другому (пассивному), между ними могут быть объекты посредники;

2) включения – если объект является результатом декомпозиции более сложного объекта.

Объектная декомпозиция, так же, как и процедурная, может применяться многократно или быть многоуровневой. Это значит, что каждый объект может рассматриваться как система, которая состоит из элементов, взаимодействующих друг с другом через передачу сообщений. При многоуровневой декомпозиции на каждом уровне мы получаем объекты с более простым поведением, что позволяет разрабатывать системы повышенной сложности по частям.

Объект – элемент системы, объединяющий в себе данные и операции над ними, обладающий свойствами наследования, инкапсуляции и полиморфизма.

Каждый выделяемый объект предметной области должен уметь выполнять некоторые действия, зависящие от полученных сообщений и параметров самого объекта.

Совокупность значений параметров объекта называют его состоянием, а совокупность реакций на получаемые сообщения – поведением.

Параметры состояния и элементы поведения объектов определяются условием задачи.

В процессе решения задачи объект, получив некоторое сообщение, выполняет заранее определенные действия, например, может изменить собственное состояние, выполнить некоторые вычисления, нарисовать окно или график и, в свою очередь, сформировать сообщения другим объектам. Таким образом, процессом решения задачи управляет последовательность сообщений. Передавая эти сообщения от объекта к объекту, система выполняет необходимые действия.

Принципиально можно выделить 2 вида разбиения предметной области на составляющие элементы:

Алгоритмическая декомпозиция (основные элементы программы – строительные блоки – алгоритмы).

Объектная декомпозиция (основные элементы программы – виды абстракций (классы) и представители этих классов (объекты)).

В соответствии с алгоритмической декомпозицией (см. рис. 8.2) предметной области мы при анализе задачи пытаемся понять, какие алгоритмы необходимо разработать для ее решения, каковы спецификации этих алгоритмов (вход, выход) и как эти алгоритмы связаны друг с другом. В языках программирования данный подход в полной мере поддерживается средствами модульного программирования (библиотеки, модули, подпрограммы).

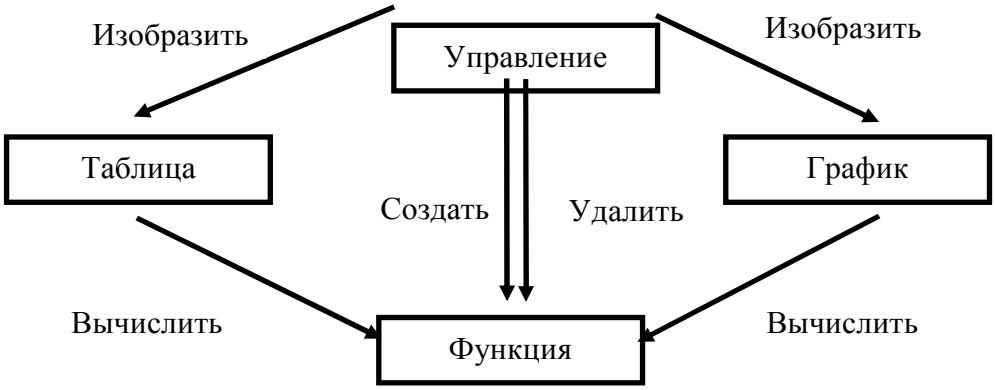


Рис. 8.2 – Объектная декомпозиция программы построения таблиц и графиков

В рамках объектной декомпозиции мы пытаемся выделить основные содержательные элементы задачи, разбить их на типы (классы). Далее для каждого класса абстракций мы определяем его свойства (данные) и поведение (операции), а также, как эти классы абстракций взаимодействуют друг с другом.

На сегодняшний день объектный подход и его основы – объектная модель и объектная декомпозиция – поддерживаются современными объектно-ориентированными языками программирования (C++, Java, C#…).

Пример объектной декомпозиции контроля качества средств доставки сводится к контролю количественных и качественных характеристик материально-технического обеспечения учебного процесса, как это показано на рисунке 8.3.

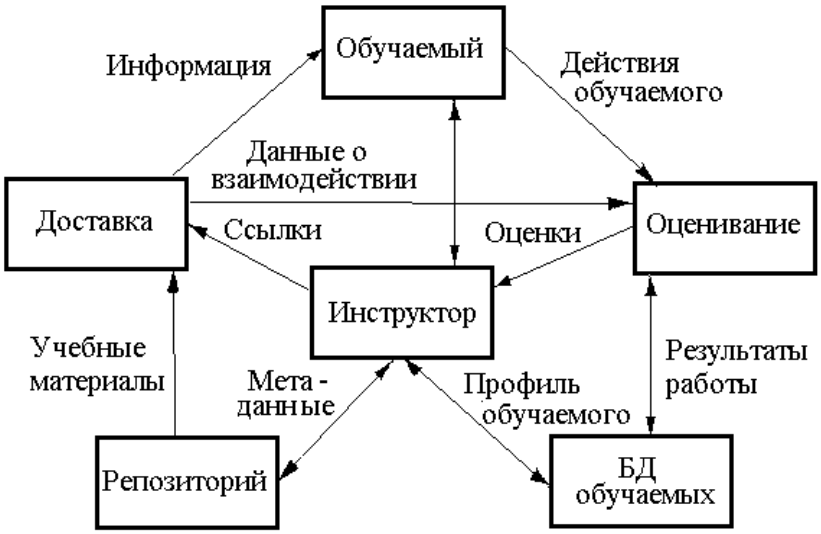


Рис. 8.3 – Объектная декомпозиция процесса оценки качества средств доставки

Другой пример объектной и структурной декомпозиции, связанный с описанием игрового движка (рис. 8.4 и 8.5). Игровой процесс можно смоделировать, используя два объекта: «Интерфейс» (блок управления, который получает и интерпретирует команды пользователя) и «Игра» (реализация игрового процесса), каждый со своими параметрами (рис. 8.4).

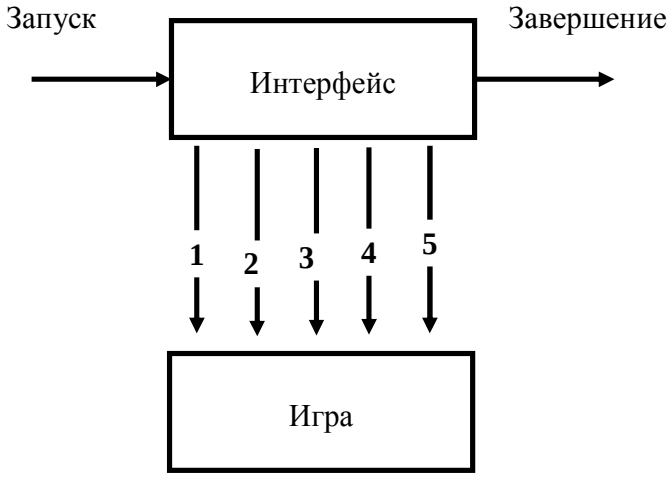


Рис. 8.4 – Объектная декомпозиция игрового процесса (1 – начать игру, 2 – проверка, попал ли по стене или планке, 3 – проверка, попал ли по кирпичу, 4 – проверка, не закончилась ли игра, 5 – отобразить)

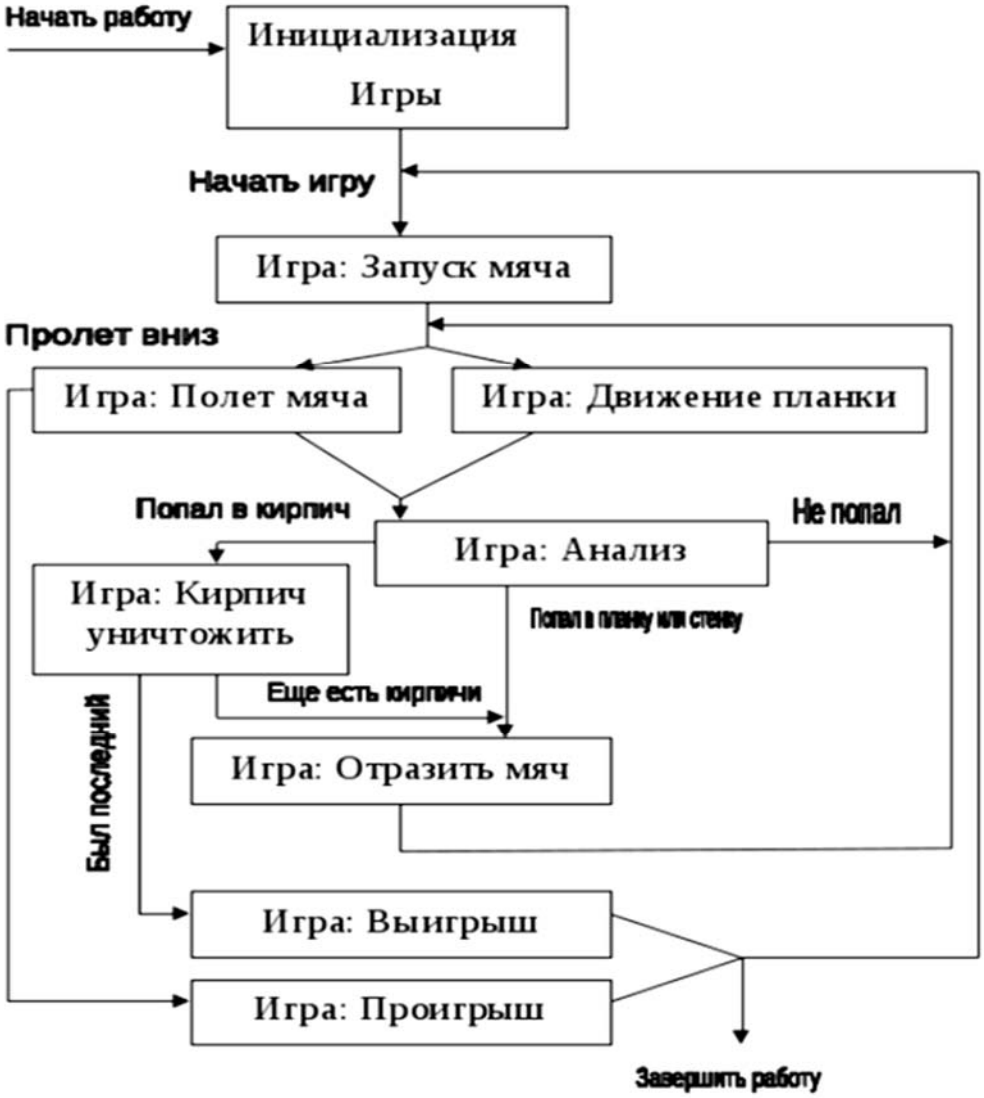


Рис. 8.5 – Структурная декомпозиция программного продукта

Рекомендации по выполнению объектной декомпозиции:

1. Для сложных систем объектная декомпозиция должна выполняться поэтапно: на первом этапе – объектная декомпозиция всей системы, на последующих – декомпозиция объектов как подсистем.

2. При декомпозиции системы в целом в качестве объектов могут выделяться элементы двух типов:

– элементы интерфейса пользователя (окна меню, окна сообщений, окна форм ввода-вывода и т. д.);

– средства хранения, организации и преобразования данных (базы данных, файлы, протоколы, структуры данных и т. д.).

При этом для каждого объекта должно определяться множество получаемых и передаваемых сообщений и основные характеристики.

3. Процесс декомпозиции прекращается при получении объектов, которые могут быть достаточно просто реализованы, т. е. имеют четко определенную структуру и поведение.

9 МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ УЧЕБНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЫ

В процессе выполнения учебно-исследовательской работы студенты должны выполнить основные элементы (более детально изложены пункты задания в п. 9.1 и 9.2):

1. Исследовать предметную область, сформулировать задание на собственный проект, выявить программы аналоги, определить их функциональные характеристики и подготовить обзор.

2. Изучить ГОСТы, связанные с разработкой программного обеспечения.

3. Разработать развернутое техническое задание (ТЗ) на программный продукт.

4. На основе ТЗ выполнить разработку программного продукта на языке UML (диаграмма классов, диаграмма использования, диаграмма деятельности).

5. Выбрать структуры данных для реализации предметной области программного продукта.

6. Разработать интерфейс пользователя.

7. Выбрать язык и среду программирования.

8. Разработать алгоритмы и реализовать их в выбранной среде проектирования.

9. Выбрать стратегию тестирования и разработать тесты.

10. Выполнить тестирование и отладку.

11. Разработать техническую документацию и руководство пользователя.

Выполнение учебно-исследовательской работы осуществляется в несколько этапов. В конце каждого этапа студент демонстрирует преподавателю результаты выполнения работы. Основные этапы учебно-исследовательской работы и представляемые преподавателю результаты приведены в таблице 9.1.

Таблица 9.1 – Этапы выполнения учебно-исследовательской работы

Этап	Содержание	Дисциплина	Представляемые результаты. Вид контроля
1	Пункты 1, 2	УПД-1	Ознакомление с методологией исследования, нормативной документацией, литературными источниками. Выбор темы исследования. Ознакомление с российскими и зарубежными разработками по выбранной теме. Анализ и подготовка результатов обзора по выбору языка и среды программирования. Постановка задачи по теме и составление объектной декомпозиции. Зачет с оценкой
2	Пункт 3	УПД-2	Ознакомление с требованиями к подготовке технического задания. Написание технического задания. Зачет с оценкой
3	Пункты 4 – 6	УПД-3	Разработка классов приложения и их взаимодействия. Описание прототипа программного продукта на языке UML. Зачет с оценкой
4	Пункты 7 – 11	УПД-4	Описание программного кода продукта. Разработка приложения. Проведение тестирования. Пояснительная записка о тестировании. Техническая документация и руководство пользователя. Зачет с оценкой

Выбор предметной области для проведения исследования осуществляется студентом самостоятельно. Желательно и приветствуется, чтобы тема проекта была связана с работой студента на производстве или в сфере услуг, что послужит залогом разработки оригинального, а главное актуального проекта.

Учебно-исследовательская работа предполагает последовательно-поочерёдное выполнение работ в соответствии с замечаниями на каждом этапе. Не следует отправлять на проверку следующую работу, не получив оценку за предыдущую и указания по следующей.

Формой контроля выполненной работы на каждом этапе является отчёт, который дополняется информацией из предыдущих отчётов в соответствии с указаниями.

9.1 Задания для выполнения учебно-исследовательской работы

9.1.1 Задания для УПД-1

Задания

1. Знакомство с методологическими аспектами подготовки и проведению научных исследований (см. гл. 1 и 2).

2. Изучение нормативной документации в области разработки ПО (см. приложения Б и Д).

3. Выбор предметной области для проведения исследования.

4. Изучение литературных и интернет-источников, анализ и написание обзора.

5. Формулирование задачи в соответствии с выбранной темой исследования.

6. Составление объектной декомпозиции (см. гл. 8 и 9).

7. Подготовка отчета (см. гл. 3 и 4).

Формой контроля выполненной работы является отчет.

Примерное содержание отчета по УПД-1

Титульный лист

Содержание

Реферат

Введение

1. Теоретическая часть

1.1 Обзор предметной области рассматриваемой задачи

1.2 Описание задачи как объекта исследования

1.3 Выбор методологии разработки программного обеспечения

1.4 Составление объектной декомпозиции (структуры ПО)

Список использованных источников

Приложения

9.1.2 Задания для УПД-2

Задания

1. Знакомство с требованиями по подготовке технического задания.

2. Написание технического задания согласно ГОСТу (см. гл. 7 и приложение В).

3. Подготовка отчета (см. гл. 3 и 4).

Формой контроля выполненной работы является отчет.

Примерное содержание отчета по УПД-2

Титульный лист

Содержание

Реферат

Введение

1. Теоретическая часть

1.1 Обзор предметной области рассматриваемой задачи

1.2 Описание задачи как объекта исследования

1.3 Выбор методологии разработки программного обеспечения

1.4 Составление объектной декомпозиции (структуры ПО)

2. Техническое задание на разработку ПО

2.1 Нормативная документация разработки ПО

2.2 Выбор основных элементов реализации ПО

2.3 Техническое задание

Список использованных источников

Приложения

9.1.3 Задания для УПД-3

Задания

1. Разработка классов приложения и их взаимодействие (см. гл. 6).

2. Разработка UML-диаграммы классов (см. гл. 6).

3. Подготовка отчета (см. гл. 3 и 4).

Разрабатываемая в рамках учебно-исследовательской работы программа (проект), решающая поставленную задачу, должна использовать технологию объектно-ориентированного программирования.

К проекту (отчету) необходимо оформить программную документацию согласно регламентной документации в виде технического задания – формулировки решаемой задачи, требований к программе, языка программирования и др. В рамках этой программы следует спроектировать структуру классов и взаимодействие на языке UML.

Формой контроля выполненной работы является отчет.

Примерное содержание отчета по УПД-3

Титульный лист

Содержание

Реферат

Введение

1. Теоретическая часть

1.1 Обзор предметной области рассматриваемой задачи

1.2 Описание задачи как объекта исследования

1.3 Выбор методологии разработки программного обеспечения

1.4 Составление объектной декомпозиции (структуры ПО)

2. Техническое задание на разработку ПО

2.1 Нормативная документация разработки ПО

2.2 Выбор основных элементов реализации ПО

2.3 Техническое задание

3. Практическая часть

3.1 Объектно-ориентированная технология разработки ПО

3.2 Анализ требований на UML

3.3 Интерфейс ПО. Классы интерфейса на UML

3.4 Классы обработки данных на UML

3.5 Классы управления задачами на UML

Список использованных источников

Приложения

Теоретические основы

Наибольшее распространение среди методов анализа ПО получил метод, учитывающий:

– информационную модель системы – объектную декомпозицию;

– модель состояний объектов в информационной модели системы, определяемую набором правил поведения, предписаний и законов;

– модель процессов, которая отображает процессы и действия, совершающиеся в системе и обеспечивающие прохождение моделей состояний через жизненные циклы – получение, порождение и уничтожение событий в системе.

Согласно этого методу ПО анализируется в три этапа: информационное моделирование, моделирование состояний, моделирование процессов. В результате их выполнения создается система в виде совокупности этих моделей. Информационная модель отображает ПО как мир объектов с характеристиками и атрибутами.

При переходе от этого этапа к этапу моделирования состояний для объектов информационной модели определяются связи объектов и их поведение. Создается модель состояний, которая отображает динамику состояния объектов системы и их поведение. На третьем этапе определяются действия и процессы, которые порождают события. Действия имеют функциональную природу. Цель моделирования процессов состоит в том, чтобы расчленить процессы на действия, которые вместе взятые определяют функциональное содержание системы. Рассмотрим модели метода подробнее.

Под информационной моделью понимается совокупность объектов (сущностей) ПО, их характеристик (атрибутов) и связей между ними. Она создается по принципу реляционной модели данных, т. е. представления данных в виде отношений между ними.

Анализ ПО состоит в выявлении объектов, предоставлении им уникальных и значимых названий, соответствующих смысловым понятиям в этой предметной области. В качестве объектов могут выступать:

1) реальные предметы мира ПО как абстракции фактически существующих физических объектов ПО;

2) роли как абстракции целей или назначения человека, части организации;

3) взаимодействия – объекты, получаемые путем установления отношений между другими объектами или частями системы;

4) спецификации, используемые для представления правил, критериев и ограничений на применение объектов в системе.

Таким образом, элементами информационной модели объектной декомпозиции могут быть объекты, их атрибуты и идентификаторы, а также связи между объектами. Для объектов ПО определяются их характерные признаки или свойства, называемые атрибутами. Каждый атрибут – это абстракция одной характеристики объекта, которая присуща всем представителям класса объектов. Для классов объектов выбираются уникальные

имена, устанавливаются атрибуты и связи. Атрибут получает имя, уникальное в рамках класса. Различаются описательные, указывающие и вспомогательные атрибуты.

Описательный атрибут устанавливает реальную характеристику, которая может определяться одним из таких возможных способов:

1) заданием числового диапазона;

2) перечислением возможных значений, которые может принимать атрибут;

3) ссылкой на документ, который определяет возможные значения;

4) заданием правил генерации допустимых значений.

Указывающий атрибут задает форму, назначение, перечисление или ссылку. Дополнительный атрибут задает дополнительные значения, которые может принимать атрибут объекта.

Идентификаторы объекта содержат один или несколько атрибутов, значения которых позволяют однозначно выделить экземпляр объекта в данном классе (например, табельный номер сотрудника, номер паспорта и др.). Ссылка на некоторый атрибут может уточняться именем класса, задаваемым через точку, а атрибуты – отношениями, которые определяются по следующим правилам:

1) каждый объект – экземпляр класса или более чем одного класса, обладает одним значением своего атрибута;

2) идентификатор может составляться из нескольких имен атрибутов (через точку), первое имя относится к классу, остальные к имени объекта.

Связи объектов устанавливаются между объектами одного или другого класса и характеризуются количеством экземпляров объектов, которые одновременно могут принимать участие в этих связях.

В информационной модели объектной декомпозиции (см. рис. 9.1) связи между объектами изображаются стрелками, указывающими направление связи. Возле рамки объекта, принимающего участие в связи, на линии

стрелки указывается роль, которую этот объект поддерживает в данной связи. Связь 1:1 обозначается двунаправленной стрелкой, имеющей по одному «наконечнику» с каждой стороны; связь 1:N представляется стрелкой, имеющей два «наконечника» со стороны объекта, который состоит в связи с несколькими объектами; и, наконец, по два «наконечника» с каждой стороны имеет стрелка, означающая связь N:M.

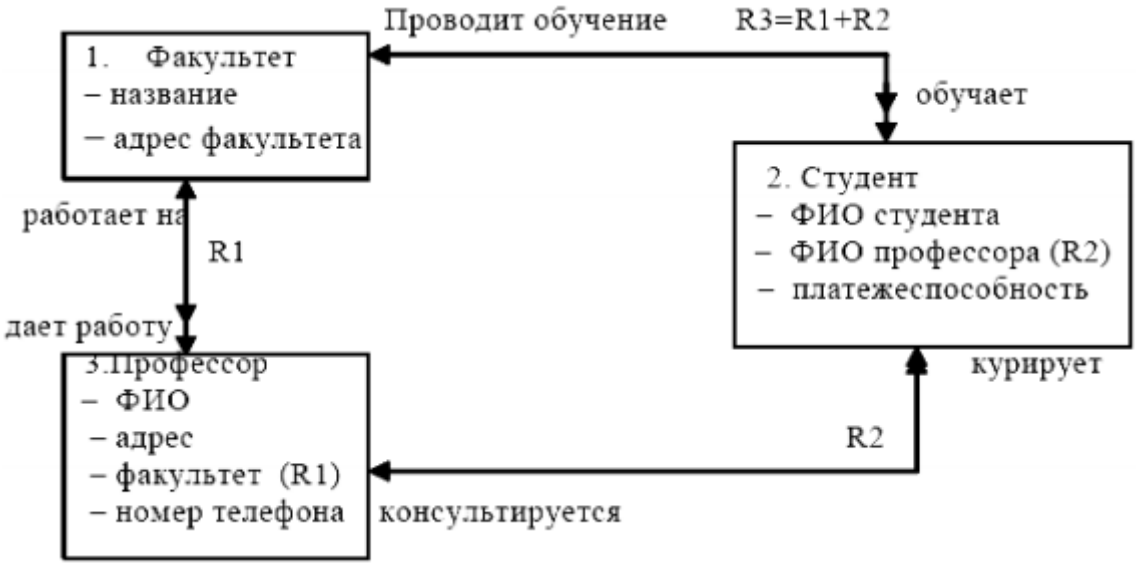


Рис. 9.1 – Пример объектной декомпозиции
разрабатываемой программы

Модель состояний – предназначена для отображения динамического поведения и изменения состояний каждого из объектов информационной модели и жизненного цикла поведения объектов. Состояние в модели – это положение или ситуация объекта, определяемая правилами и линией поведения. Событие – это инцидент, который заставляет объект переходить из одного состояния в другое. Экземпляры класса имеют поведение, которое определяется:

1) состоянием, зависящим от текущих значений отдельных его атрибутов;

2) состоянием, изменяемым в результате выполненных над объектами действий;

3) состоянием ПО, зависящим от совокупности состояний ее объектов;

4) некоторыми процессами и действиями, которые изменяют жизненный цикл состояния объекта.

Построение модели состояний начинается после выделения в информационной модели отдельных объектов, обладающих динамическим поведением (например, изменение состояния с течением времени), создания экземпляра объекта или его уничтожения после прекращения существования (например, электрическая лампочка перегорает, тем самым закончился ее ЖЦ). В данном методе предусмотрены две нотации для представления динамических аспектов поведения объектов: диаграмма перехода состояний и таблица перехода в состояния.

При построении модели состояний для каждого объекта информационной модели определяется следующее:

1) множество состояний, в которых объект может находиться;

2) множество инцидентов или событий, которые побуждают экземпляры класса изменять свое состояние;

3) правила перехода объекта из зафиксированного состояния на новое состояние при условии, что произойдет некоторое событие из множества событий;

4) действие на каждое из состояний выполняется при переходе в новое состояние.

Эта информация представляется в диаграмме перехода состояний (рис. 9.1) исходя из следующих условий:

1) каждое состояние, определенное для класса объектов, получает номер, уникальный идентификатор (ID) и название;

2) состояние обозначается рамкой, содержащей номер и название;

3) переход от состояния к состоянию изображается направленной дугой, помеченной меткой и названием события, обусловившего переход;

4) начальное состояние обозначается стрелкой, направленной к соответствующей рамке, и является состоянием, которое экземпляр объекта приобретает после своего создания;

5) заключительное состояние жизни экземпляра объекта (продолжение или разрушение) обозначается пунктирной рамкой;

6) указание на действия, которые должны быть выполнены экземпляром объекта для перехода в другое состояние.

Изменение состояния экземпляра класса объектов осуществляется при выполнении таких действий:

1) обработка информации, переданной в систему, что может повлиять на некоторое событие;

2) изменение поведения атрибута объекта;

3) вычисление атрибута;

4) генерация некоторой операции для одного из экземпляров класса объектов;

5) генерация события, сообщение о котором передается объекту, внешнему по отношению к данному;

6) прием сообщения о событии от внешних объектов;

7) взаимодействие с таймером, измеряющим время, истечение которого приводит к созданию некоторого события.

Спецификация – подробное описание системы, которое полностью определяет ее цель и функциональные возможности. Различают:

– словесные спецификации на естественном языке;

– модельные спецификации;

– формальные спецификации.

Словесные спецификации на естественном языке как раз и вызывают массу проблем, поскольку создаются разными специалистами на «их языке».

Другим видом спецификаций являются формальные спецификации. Действительно, описание спецификации с помощью строгого математического языка было бы чудесным решением всех проблем, т. к. сам способ записи исключал бы малейшие неоднозначности. Да, в математике есть, например, алгебра высказываний, с помощью которой можно пытаться создавать технические задания на разработку некоторых приложений. Проблема кроется в слове «некоторых». Понятно, что формальная спецификация является, по сути, математической моделью задачи и потому для вычислительных задач все выглядит достаточно просто. Формализация же задач из других областей знаний может оказаться более сложной и трудоемкой проблемой, чем разработка самого приложения ввиду отсутствия четкой математической модели.

На рисунке 9.2 представлена формализация процесса обучения студента, которая сразу позволяет представить любому человеку все необходимые для обучения действия и возможные их варианты.

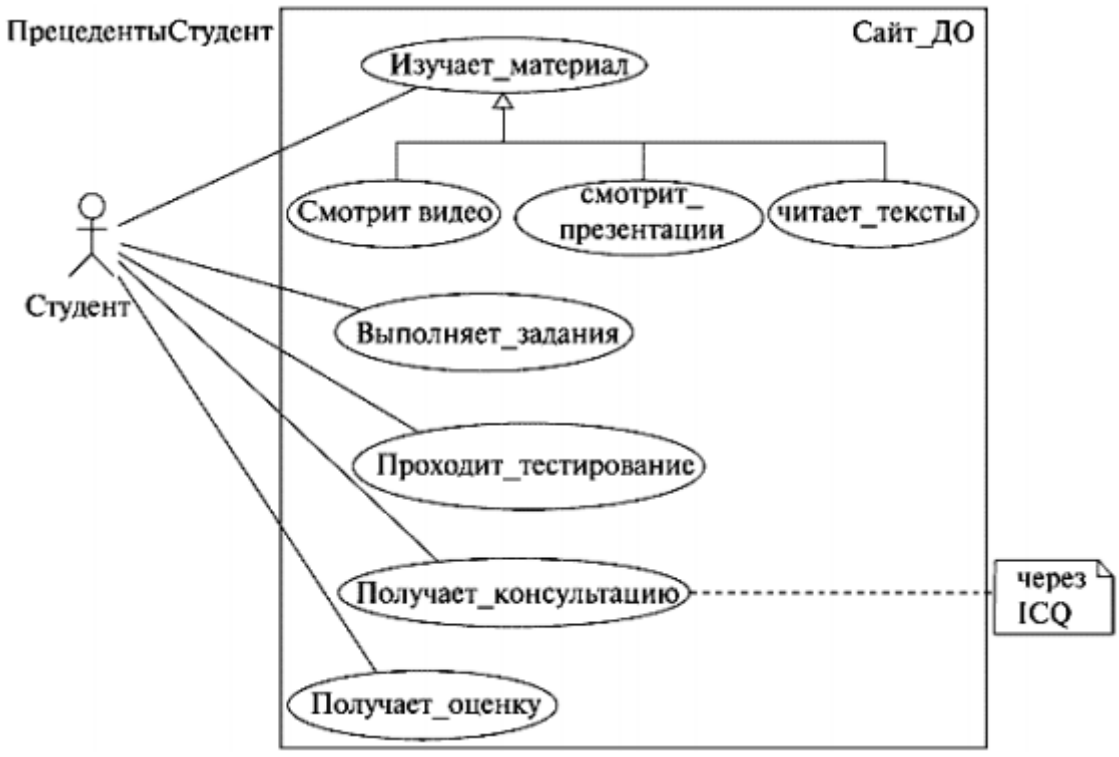


Рис. 9.2 – Формализация процесса обучения студента

Язык моделирования UML позволяет строить модели программных систем, по которым потом можно производить генерацию программного кода проектируемого программного обеспечения. Немаловажной чертой UML-модели является тот факт, что они сами по себе уже являются документами (и весьма понятными, даже для неспециалиста, см. рис. 9.2). Причем любой элемент на любой диаграмме может быть снабжен поясняющим текстовым комментарием. Таким образом, построение набора диаграмм уже является процессом документирования будущей программной системы.

Следует пояснить более детально основные сущностные элементы предыдущего параграфа.

Система – совокупность взаимосвязанных управляемых подсистем, объединенных общей целью функционирования. Подсистема – это элемент системы, функционирование которой не зависит от других подсистем. Программная система структурируется в виде совокупности относительно независимых подсистем. Также определяются взаимодействия между подсистемами.

Модель – это некий (материальный или нет) объект, отображающий лишь наиболее значимые для данной задачи характеристики системы. Модели бывают разные – материальные и нематериальные, искусственные и естественные, декоративные и математические и др., например, школьный глобус является моделью земного шара.

Диаграмма – это графическое представление множества элементов. Обычно изображается в виде графа с вершинами (сущностями) и ребрами (отношениями). Примеров диаграмм можно привести множество: блок-схема, схемы монтажа различного оборудования, дерево файлов и каталогов на диске и многое другое.

При проектировании ПО с помощью диаграмм можно визуализировать программную систему с различных точек зрения. Одна из диаграмм,

например, может описывать взаимодействие пользователя с системой, другая – изменение состояний системы в процессе ее работы, третья – взаимодействие между собой элементов системы и т. д. Сложную систему можно и нужно представить в виде набора небольших и почти независимых моделей-диаграмм, причем ни одна из них не является достаточной для описания системы и получения полного представления о ней, поскольку каждая из них фокусируется на каком-то определенном аспекте функционирования системы и выражает разный уровень абстракции. Другими словами, каждая модель соответствует некоторой определенной, особой точке зрения на проектируемую систему.

Диаграммы – лишь средство визуализации модели, и эти два понятия следует различать. Лишь весь набор диаграмм составляет модель системы и наиболее полно ее описывает, но не одна диаграмма, вырванная из контекста.

Виды диаграмм UML:

– диаграмма прецедентов;
– диаграмма классов;
– диаграмма объектов;
– диаграмма последовательностей;
– диаграмма взаимодействия;
– диаграмма состояний;
– диаграмма активности;
– диаграмма развертывания.

Одной из самых важных диаграмм, определяющих работу всей программной системы, является Диаграмма классов (class diagram). Классы – это строительные блоки любой объектно-ориентированной системы. Они

представляют собой описание совокупности объектов с общими атрибутами, операциями, отношениями и семантикой. При проектировании объектно-ориентированных систем диаграммы классов обязательны.

Классы используются в процессе анализа предметной области для составления словаря предметной области разрабатываемой системы. Это могут быть как абстрактные понятия предметной области, так и классы, на которые опирается разработка и которые описывают программные или аппаратные сущности.

Диаграммы классов применяются при проектировании ПО, то есть в процессе разработки новой системы, и при обратном проектировании – описании существующих и используемых систем.

В примере диаграммы классов (см. рис. 9.3) показан базовый класс и операции наследования элементов бытовой техники.



Рис. 9.3 – Диаграмма классов описания
элементов бытовой техники

Рассмотрим другой, более сложный пример диаграммы классов (см. рис. 9.4).



Рис. 9.4 – Диаграмма классов автоматизации работы учебного учреждения

Данная диаграмма классов описывает предметную область задачи автоматизации работы учебного учреждения. Текстом показывается смысловая функция операции, а цифрами указаны кратности на концах связей (количественное описание).

Более сложная диаграмма классов (см. рис. 9.5), более предназначенная для программирования конкретной, а не обобщенной задачи, как, например, представленной на рисунке 9.2.



Рис. 9.5 – Диаграмма классов прохождения студентом курса

Разработчик программного обеспечения в первую очередь обращает внимание на объекты предметной области, а программист же концентрируется на поведении этих объектов, пользуясь классами, к которым они принадлежат. Класс на диаграмме изображается в виде прямоугольника, разделенного горизонтальными линиями на три части. В первой части указывается название класса. Как правило, имя класса состоит из одного, максимум двух слов. Вторая часть содержит перечень атрибутов класса, которые характеризуют тот или иной объект этого класса в модели предметной области. Третья часть содержит перечень операций, отражающих его поведение в модели предметной области (рис. 9.6).

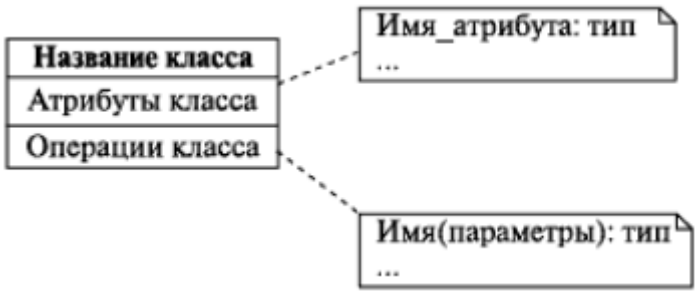


Рис. 9.6 – Формальное описание класса

На первом этапе разработки ПО пользователю необязательно знать структуру класса, т. е. набор атрибутов (данных) и операций (методов, работающих с этими данными). Сокрытие от пользователя внутреннего устройства объектов называется инкапсуляцией. Инкапсуляция – это защита отдельных элементов объекта, не затрагивающих существенных характеристик его как целого.

В программировании инкапсуляция обеспечивается с помощью т. н. модификаторов видимости. С их помощью можно ограничить доступ к атрибутам и операциям объекта со стороны других объектов. Так, если атрибут или операция описаны с модификатором private, то доступ к ним можно

получить только из операции, определенной в том же классе. Если же атрибут или операция описаны с модификатором видимости public, то к ним можно получить доступ из любой части программы. Модификатор protected разрешает доступ только из операций этого же класса и классов, создаваемых на его основе.

В языках программирования могут встречаться модификаторы видимости, ограничивающие доступ на более высоком уровне, например, к классам или их группам, однако смысл инкапсуляции от этого не изменяется. В UML атрибуты и операции с модификаторами доступа обозначаются специальными символами слева от их имен.

Доступ к элементам класса возможен через интерфейс. Интерфейс отражает внешние проявления объекта, показывает, каким образом осуществляется взаимодействие с ним, скрывая остальные детали, не имеющие отношения к процессу взаимодействия. Интерфейс всегда реализуется некоторым классом, который в таком случае называют классом, поддерживающим интерфейс. Как мы уже говорили ранее, один и тот же объект может иметь несколько интерфейсов. Это означает, что класс этого объекта реализует все операции этих интерфейсов.

Сложное ПО может состоять из многих десятков, сотен классов, взаимодействующих между собой. В таком наборе возможны пересечения, взаимные области действия, поэтому надо искать атрибуты, операции и обязанности, общие для двух или более классов из данной совокупности. Это позволит избежать ненужного дублирования структуры и функциональности объектов. Обобщенные элементы выносятся в некоторый общий суперкласс, а если такого не существует, то надо создать такой класс (см. рис. 9.7).



Рис. 9.7 – Формирование суперкласса
средствами UML

Примером суперкласса может быть описание класса Фигура, которая может быть представлена точкой, кругом или прямоугольником (см. рис. 9.8).

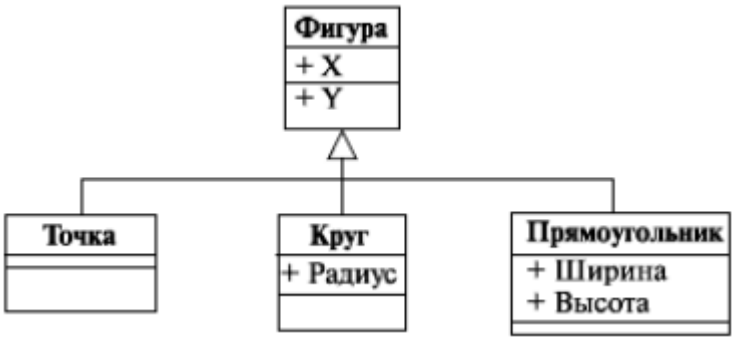


Рис. 9.8 – Описание суперкласса «Фигура»
средствами UML

Суперкласс «Фигура», имеет два атрибута – координаты центра, а все остальные классы на этой диаграмме связаны с классом «Фигура» отношением обобщения, т. е. в них нужно доопределить только недостающие атрибуты – радиус, ширину и высоту. Атрибуты, описывающие координаты центра, эти классы имеют изначально как потомки класса «Фигура» – они их наследуют.

Программные объекты разных классов могут поддерживать один и тот же интерфейс именно так, как того ожидает пользователь. Примером тому может служить рассмотренная выше диаграмма с геометрическими фигурами (см. рис. 9.8). Все рассмотренные фигуры имеют, например, операцию рисования на экране. С точки зрения пользователя, в каждом случае это одно и то же действие. Однако реализованы эти операции по-разному – ведь процедура изображения прямоугольника сильно отличается от подобной процедуры для круга.

Это стало возможно благодаря одному из основных принципов ООП – полиморфизму. Полиморфизм является основой для реализации механизма интерфейсов в языках программирования.

Отношения между классами – важная тема программирования. Ни один из объектов окружающего нас мира не существует сам по себе. Каждый из нас связан с массой других людей разнообразными родственными, профессиональными и другими связями, предполагающими различные типы отношений, или, в конечном случае, жизнью на одной планете. Точно так же и классы связаны между собой. И чтобы в полной мере овладеть ООП, нам необходимо понять суть этих отношений и научиться их идентифицировать.

Программные объекты находятся в определенных отношениях друг с другом. Один из типов таких отношений – это зависимость. Зависимость возникает тогда, когда реализация класса одного объекта зависит от спецификации операций класса другого объекта. И если изменится спецификация операций этого класса, нам неминуемо придется вносить изменения и в зависимый класс.

Приведем пример, иногда попадаются видеофайлы которые не удается воспроизвести. Ответ ясен, на компьютере не установлены соответствующие кодеки. Отсюда вывод, что медиаплеер, который вы используете,

зависит от кодека. Эта зависимость между классами изображается в UML (см. рис. 9.9).



Рис. 9.9 – Зависимость классов в UML

Стоит отметить, что зависимости на диаграммах изображают далеко не всегда, а только в тех случаях, когда их отображение является важным для понимания модели. Часто зависимости лишь подразумеваются, т. к. логически следуют из природы классов.

Другой вид отношений между объектами – это ассоциация. Это просто связь между объектами, по которой можно между ними перемещаться. Ассоциация может иметь имя, показывающее природу отношений между объектами, при этом в имени может указываться направление чтения связи при помощи треугольного маркера. Однонаправленная ассоциация может изображаться стрелкой. Проиллюстрируем сказанное примерами (см. рис. 9.10).

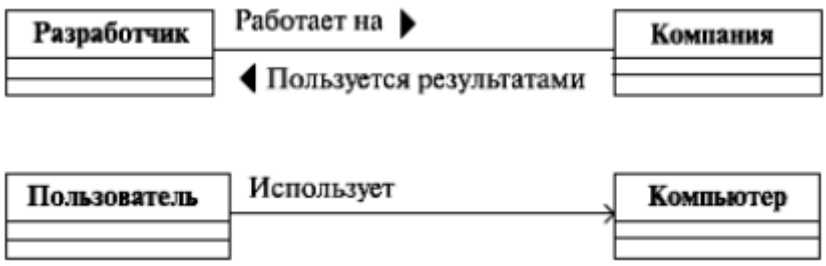


Рис. 9.10 – Отношения между объектами в UML

Кроме направления ассоциации, мы можем указать на диаграмме роли, которые каждый класс играет в данном отношении, и кратность, то есть количество объектов, связанных отношением (см. рис. 9.11).

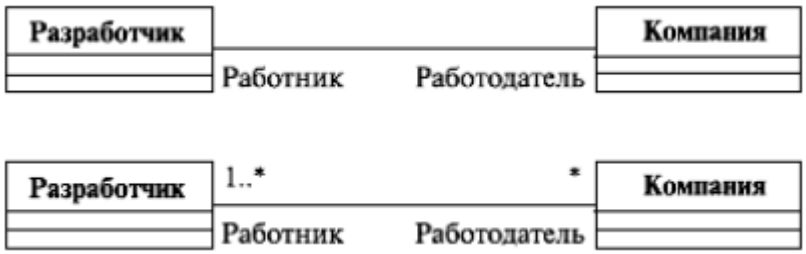


Рис. 9.11 – Направления ассоциации при отображении роли

В отношении между двумя классами сама ассоциация тоже может иметь свойства и, следовательно, тоже может быть представлена в виде класса (например, см. рис. 9.12).



Рис. 9.12 – Свойства ассоциации

Действительно, перед началом трудовых отношений работник и работодатель подписывают между собой контракт, который имеет такие атрибуты, как, например, описание работ, сроки их выполнения, порядок оплаты и т. д.

9.1.4 Задания для УПД-4

Задания

1. Ознакомиться с литературными и интернет-источниками, провести анализ и подготовить обзор по программам аналогам, близким к разрабатываемой (приводится описание и таблица, где по строкам расположены программы, а по столбцам их функциональные характеристики).

2. Изучить информацию в литературных и интернет-источниках, провести анализ и написание обзора по выбору языка и среды программирования.

3. На основе разработанной объектной декомпозиции и технического задания произвести разработку программного обеспечения (например, с использованием Visual Studio C++ 2010).

4. Провести тестирование программы (подготовить тесты и правила проведения тестов согласно ГОСТ (см. приложение Д)).

5. Подготовить документацию на применение программы (см. гл. 3 и 4; приложения А, Б, Г).

6. Подготовить отчет по проделанной работе за семестр (см. гл. 3 и 4).

Формой контроля выполненной работы за семестр является программный проект, включающий файлы разработанного программного кода, и отчет.

Примерное содержание отчета по тестированию ПО

Титульный лист

Содержание

Введение

1. Описание структуры разработанной программы

1.1 UML-диаграммы (use-case, data flow, classes)

1.2 Методы хранения данных

1.3 Основные функции программы

2. Тестирование программного обеспечения

2.1 Описание входных и выходных данных программы (тест)

2.2 Результаты тестирования

Список использованных источников

Приложения

Требования к отчетам по дисциплине
«Учебно-проектная деятельность»

Обучающемуся для получения аттестации по данной дисциплине необходимо в каждом семестре проводить необходимые учебно-исследовательские работы и оформлять отчет по полученным результатам, согласно заданиям и требованиям, указанным для каждого семестра.

Формой аттестации в каждом семестре является зачет с оценкой (дифференцированный зачет).

Отчет оформляется в соответствии с образовательным стандартом университета ОС ТУСУР 01–2021 «Работы студенческие по направлениям подготовки и специальностям технического профиля. Общие требования и правила оформления» (режим доступа: https://regulations.tusur.ru/documents/70).

9.2 Примеры заданий по проекту УИР

Пример 1

Разработать информационную систему, которая автоматизирует процесс резервирования билетов. Система должна: (а) содержать расписание рейсов (город вылета, прилета, авиакомпания, номер рейса, тип самолета, дни недели выполнения рейса) и иметь возможность обновлять его; (б) список типов самолетов (тип самолета, план мест в салоне, например 12 рядов по 4 места в ряду: ABCD), какие места бизнес- и экономического класса, места для курящих и некурящих); (в) показать наличие рейсов между парой

городов (например, между Москвой и Петербургом); (г) список блюд, предлагаемых пассажирам (название блюда, его код); (д) иметь возможность резервировать места (и отказаться от места) для пассажира на данный рейс с заказом блюд на питание из заданного списка; (е) выдать упорядоченный список пассажиров по заданному признаку: номер рейса, дата вылета, заказанное блюдо; (ж) выдавать список пассажиров, отказавшихся от места; (з) выдавать список блюд и их количество, заказанное пассажирами.

Пример 2

Разработать информационную систему, которая автоматизирует деятельность поликлиники. Система содержит: списки врачей (ФИО, номера, специальности врача, дату приема на работу, номер кабинета); списки больных (ФИО, номер медкарты, год рождения, пол); списки болезней (название, ее номер, признак группы болезни); списки посещений (номер больного, номер врача, номер болезни, дата). Система должна проверять правильность приема врачом больного (номер группы болезни должен совпасть со специальностью врача) и выполняет следующие функции: (а) вводит данные по персоналу врачей; (б) вводит данные о больных, (в) регистрирует посещения; (г) выдает справки о посещении больного врачом по дате, месяцу, году; (д) выдает справки о приеме врачом больных за день, месяц, год; (ж) выдает справки по врачам данной специальности (например, терапевты) о числе принятых больных за месяц, год.

Пример 3

Разработать информационную систему, которая автоматизирует учебный процесс и обрабатывает данные о преподавателях, учебных дисциплинах и группах. Система содержит: (а) список преподавателей (ФИО, кафедра, должность, номер преподавателя, дата приема на работу); (б) список дисциплин (название, код, семестр, специальность); (в) список дисциплин

кафедры (код, номер преп., количество часов). Система должна: (г) учитывать возможность чтения одной дисциплины несколькими преподавателями, и наоборот; (д) учитывать возможность изучения группой нескольких дисциплин, и наоборот; (ж) иметь средства для ввода и модификации всех перечисленных данных; (з) выдавать справки по преподавателю (какие дисциплины он читает, для каких групп и сколько часов); (е) по запросу, указав код дисциплины, получить список преподавателей, ее читающих.

Пример 4

Разработать информационную систему, которая автоматизирует деятельность авторемонтной мастерской. Система должна: (а) содержать сведения по рабочим мастерской (имя, фамилия, номер, специальность (кузовщик, электрик, слесарь и т. д.), дата приема на работу); (б) сведения о находящихся в ремонте машинах (тип, регистрационный номер, дату поступления в ремонт, характер неисправности, порядковый номер, присвоенный в мастерской, дату окончания ремонта (если он завершен), стоимость завершенного ремонта; (в) сведения о ремонте (порядковый номер, тип ремонта, список рабочих, ремонтирующих машину); (г) выдавать справки о доходах мастерской за месяц, год; (д) справки о машинах, находящихся в ремонте и по типу ремонта; (е) список рабочих, участвующих в ремонте данной машины.

Пример 5

Разработать информационную систему, которая автоматизирует деятельность магазина. Система должна: (а) содержать сведения об отделах магазина (название, номер, имя заведующего); (б) сведения о сотрудниках (имя, фамилия, номер, дата поступления на работу, номер отдела, должность, специализация); (в) сведения о товарах, поступающих в магазин (название товара, его номер, отдел, в котором он продается, стоимость,

количество единиц); (г) продажи товара – товар может продаваться в нескольких отделах (номер товара, номер отдела, дата продажи, номер продавца); (д) выдавать справки о продажах за месяц, год; (е) выдавать справки о продажах в отделе, заданного вида товара, заданного продавца за определенный период времени всех товаров; (ж) справки о количестве единиц данного вида товара.

Пример 6

Разработать информационную систему штурманского обеспечения полетов. Система должна: (а) содержать данные о промежуточных пунктах маршрута (ППМ) (название, номер ППМ, координаты ППМ – географическую широту и долготу) (смотри справочник); (б) данные о самолете (тип самолета (например, Ту-154), его бортовой номер (например, 85123), скорость взлета (320 км), скорость посадки (280), крейсерскую скорость (850 км), взлетную массу); (в) список трасс (код трассы, номер ППМ1, номер ППМ2, номер ППМ3 и т. д.); (г) выдавать план полета (по коду трассы рассчитать маршрут полета по трассе, состоящей из ППМ). Расстояние между ППМ рассчитывается по географическим координатам. Предполагается, что полет проходит по отрезкам прямых линий между ППМ. План полета самолета указанного типа – это таблица, содержащая список ППМ, расстояния между ними и итогового расстояния.

Пример 7

Разработать информационную систему, которая автоматизирует деятельность гостиницы. Система должна: (а) содержать данные о номерах: (порядковый номер, количество мест в номере, этаж, удобства (телевизор, холодильник, телефон и т. д.), стоимость (с завтраком или без); (б) данные о клиентах (имя, фамилия, адрес, номер, им занимаемый, дата приезда, дата

отъезда, состояние оплаты (оплачено все, часть); (в) выдавать справки о свободных номерах и о конкретном номере; (г) справки о доходах гостиницы за год, месяц, день (задаваемые); (д) средства регистрации клиента и предложенные ему номера.

Пример 8

Разработать информационную систему, которая автоматизирует деятельность отдела персонала. Система должна: (а) содержать сведения о сотрудниках (имя, фамилия, номер отдела, номер должности, дату приема, семейное положение, образование, пол, адрес); (б) список должностей (название, номер должности, вилку оклада (например, 15 000–20 000 руб.)); (в) список отделов (название, номер отдела, руководитель); (г) содержать список перемещений сотрудников: приказ (MEMO поле), номер приказа, старая должность, новая должность, дата перемещения, старый оклад, новый оклад; (д) выдавать справки о перемещениях за год, месяц, день; (е) выдавать справки о заданном сотруднике: личные данные, перемещения.

Пример 9

Разработать информационную систему, которая автоматизирует деятельность библиотеки. Система должна: (а) содержать данные о книгах (название книги, автор, название издательства, год издания, номер библиотечный, номер тематики, количество экземпляров); (б) издателях (название 241 издательства, адрес, место издания); (в) читателях (фамилия, адрес, город, номер карты в библиотеке); (г) выдачах (номер карты в библиотеке, номер библиотечный, дата выдачи); (д) выдавать справки по количеству взятых книг по заданной тематике, за заданный год; (е) выдавать справки по заданному читателю и его книгам; (ж) выдавать справки по рейтингу книг по данной тематике; (з) справку по данной книге и взявшим ее читателям.

Пример 10

Разработать информационную систему, которая автоматизирует деятельность компании-оператора связи. Система содержит: списки абонентов (ФИО, пол, дата рождения, паспортные данные); списки телефонных номеров (номер, дата регистрации, активен или нет, регион, тариф, абонент); список подключенных опций к телефонному номеру (телефонный номер, опция, активна или нет); список операций по номеру (номер телефона, тип операции, дата проведения, стоимость). Система должна позволять: (а) добавлять/редактировать/удалять данные об абонентах; (б) добавлять/редактировать/удалять номера телефонов для каждого абонента; (в) добавлять или удалять данные о подключенных опциях к номеру телефона; (г) добавлять или удалять данные об операциях по номеру телефона; (д) формировать отчёт об операциях по номеру телефона за период с сортировкой по дате операции, который содержит: Ф.И.О. абонента, тип операции, дата операции, стоимость; (е) формировать отчёт о тарифах (только активных или всех) с сортировкой по названию тарифа и должен содержать: название тарифа, количество подключенных к нему телефонных номеров.

Пример 11

Разработать информационную систему для контроля качества услуг, оказываемых строительными компаниями. Система должна: (а) позволять создавать учетные записи как для заказчиков, так и для подрядчиков; (б) выводить открытую информацию о запрашиваемом заказчике или подрядчике (Ф.И.О., название организации, контакты и т. д.); (в) предоставлять информацию обо всех успешных и провальных работах, выполненных подрядчиком; (г) позволять создавать новые строительные задачи заказчиком (например: поклейка обоев, заливка пола бетоном); (д) разбивать строительные задачи на более мелкие подзадачи; (е) устанавливать требуемое количество

фото или видео материалов для успешного закрытия подзадачи; (ж) предоставлять выбранному подрядчику доступ к просмотру задачи и всех связанных подзадач; (з) позволять подрядчику загружать требуемые фото или видео материалы для закрытия подзадачи; (и) предоставлять заказчику возможность вынести положительный или отрицательный вердикт по предоставленным материалам с указанием комментариев; (к) хранить всю историю обмена данными между заказчиком и подрядчиком.

Пример 12

Разработать информационную систему учета коммунальных платежей. Система должна: (а) содержать сведения об оказываемых коммунальных услугах (название, номер, тариф, способ начисления); (б) сведения о потребителях (имя, фамилия, номер, адрес); (в) сведения о поставщиках услуг (название, номер, список поставляемых услуг); (г) сведения о факте оказания коммунальной услуги (номер услуги, номер потребителя, номер поставщика, месяц и год, объем оказанной услуги); (д) выдавать справки о суммарном потреблении выбранной услуги для потребителей, проживающих по указанному адресу; (е) формировать квитанцию на оплату коммунальных услуг для выбранного потребителя за указанный месяц и год.

Пример 13

Разработать информационную систему, которая автоматизирует учет ПК и оргтехники в организации. Система должна: (а) содержать сведения по отделам организации (название, номер); (б) сведения о видах ПК и оргтехники (название, производитель, номер, характеристики); (в) сведения о находящихся на балансе единицах ПК и оргтехники (инвентарный номер, номер вида техники, номер отдела, дата установки, балансная стоимость); (г) сведения о списанных единицах ПК и оргтехники (инвентарный номер, номер вида техники, дата списания, балансная стоимость); (д) выдавать

справки о движении указанной единицы техники (перемещение по отделам вплоть до списания); (е) справки о балансной стоимости эксплуатируемой техники; (ж) список техники, эксплуатируемой в указанном отделе организации.

Пример 14

Разработать информационную систему, которая автоматизирует учет лицензионного ПО организации. Система должна: (а) содержать сведения по отделам организации (название, номер); (б) сведения об установленных в организации ПК (название, инвентарный номер, номер отдела); (в) сведения об используемом в организации ПО (название, разработчик, номер); (г) сведения об установке ПО (номер ПО, инвентарный номер ПК, лицензионный ключ, дата установки, дата удаления – если ПО в эксплуатации, поле пустое); (д) выдавать справки об установленных экземплярах ПО указанного вида; (е) список ПО, установленного на указанном ПК; (ж) список ПО, установленного в указанном отделе.

Пример 15

Разработать информационную систему, которая автоматизирует учет транспорта на автотранспортном предприятии. Система должна: (а) содержать сведения о марках автомобилей (название, номер, производитель, класс, объем кузова, грузоподъемность, объем бака, расход горючего, скорость движения); (б) сведения о водителях предприятия (ФИО, номер, стаж); (в) сведения об имеющихся транспортных средствах (марка, гос. номер, количество горючего, статус); (г) сведения о маршрутных точках (название, номер, географические координаты); (д) сведения о путевых листах (номер листа, номер транспортного средства, номер водителя, список маршрутных точек, дата и время выезда); (е) выдавать справки расходе горючего для указанного транспортного средства; (ж) списки транспортных средств, имеющих указанный статус – на стоянке, в пути, в ремонте, на тех. обслуживании, списано.

ЛИТЕРАТУРА

1. Брукс, Фредерик. Мифический человеко-месяц или как создаются программные комплексы : пер. с англ. / Фредерик Брукс. – СПб. : Символ-Плюс, 1999. – 304 с.

2. Марко, Том. Вальсируя с Медведями. Управление рисками в проектах по разработке программного обеспечения / Том де Марко, Тимоти Листер. – М. : Компания p.m.Office, 2005. – 188 с.

3. Орлов, С. А. Технологии разработки программного обеспечения : учебник / С. А. Орлов. – СПб. : Питер, 2002. – 464 с.

4. Системный анализ и принятие решений. Словарь-справочник. – М. : Высшая школа, 2004. – 616 с.

5. UML. Основы. – М. : Символ-Плюс, 2005. – 415 с.

6. Сайт «Госстандарт». – Режим доступа: www.gost.ru/

7. Крутов, В. И. Основы научных исследований / В. И. Крутов, И. М. Грушко, В. В. Попов. – М. : Высшая школа, 1980. – 400 с.

8. Кузнецов, И. Н. Подготовка и оформление рефератов, курсовых и дипломных работ / И. Н. Кузнецов. – Минск, 2000. – 131 с.

9. Сачек, М. Г. Научно-исследовательская работа студентов / М. Г. Сачек. – М., 1989. – 112 с.

10. Кузнецов, И. Н. Научное исследование. Методика проведения и оформление / И. Н. Кузнецов. – М., 2004. – 432 с.

11. Информационные технологии управления : учеб. пособие / под ред. Ю. М. Черкасова. – М. : ИНФРА-М, 2001. – 216 с.

12. Фаулер, М. UML в кратком изложении / М. Фаулер, К. Скотт. – М. : Мир, 1999. – 191 с.

13. Грейди, Буч. UML. Руководство пользователя / Буч Грейди, Джеймс Рамбо, Айвар Джекобсон. – М. : ДМК, 2000. – 432 с.

14. Фаулер М. UML. Основы : пер. с англ. / М. Фаулер, К. Скотт. – СПб. : Символ-Плюс, 2002. – 192 с.

ПРИЛОЖЕНИЕ А

ГОСТ 19.701–90 (ИСО 5807–85)

Схемы алгоритмов, программ, данных и систем, обозначения условные и правила выполнения

Настоящий стандарт распространяется на условные обозначения (символы) в схемах алгоритмов, программ, данных и систем и устанавливает правила выполнения схем, используемых для отображения различных видов задач обработки данных и средств их решения. Стандарт не распространяется на форму записей и обозначений, помещаемых внутри символов или рядом с ними и служащих для уточнения выполняемых ими функций.

Требования стандарта являются обязательными.

1. ОБЩИЕ ПОЛОЖЕНИЯ

1.1. Схемы алгоритмов, программ, данных и систем (далее – схемы) состоят из имеющих заданное значение символов, краткого пояснительного текста и соединяющих линий.

1.2. Схемы могут использоваться на различных уровнях детализации, причем число уровней зависит от размеров и сложности задачи обработки данных. Уровень детализации должен быть таким, чтобы различные части и взаимосвязь между ними были понятны в целом.

1.3. В настоящем стандарте определены символы, предназначенные для использования в документации по обработке данных, и приведено руководство по условным обозначениям для применения их: 1) в схемах данных; 2) схемах программ; 3) схемах работы системы; 4) схемах взаимодействия программ; 5) схемах ресурсов системы.

1.4. В стандарте используются следующие понятия: 1) основной символ – символ, используемый в тех случаях, когда точный тип (вид) процесса или носителя данных неизвестен или отсутствует необходимость в описании фактического носителя данных; 2) специфический символ – символ, используемый в тех случаях, когда известен точный тип (вид) процесса или носителя данных или когда необходимо описать фактический носитель данных; 3) схема – графическое представление определения, анализа или метода решения задачи, в котором используются символы для отображения операций, данных, потока, оборудования и т. д.

2. ОПИСАНИЕ СХЕМ

2.1. Схема данных.

2.1.1. Схемы данных отображают путь данных при решении задач и определяют этапы обработки, а также различные применяемые носители данных.

2.1.2. Схема данных состоит: 1) из символов данных (символы данных могут также указывать вид носителя данных); 2) символов процесса, который следует выполнить над данными (символы процесса могут также указывать функции, выполняемые вычислительной машиной); 3) символов линий, указывающих потоки данных между процессами и (или) носителями данных; 4) специальных символов, используемых для облегчения написания и чтения схемы.

2.1.3. Символы данных предшествуют и следуют за символами процесса. Схема данных начинается и заканчивается символами данных (за исключением специальных символов, указанных в п. 3.4).

2.2. Схема программы

2.2.1. Схемы программ отображают последовательность операций в программе.

2.2.2. Схема программы состоит: 1) из символов процесса, указывающих фактические операции обработки данных (включая символы, определяющие путь, которого следует придерживаться с учетом логических условий); 2) линейных символов, указывающих поток управления; 3) специальных символов, используемых для облегчения написания и чтения схемы.

2.3. Схема работы системы

2.3.1. Схемы работы системы отображают управление операциями и поток данных в системе.

2.3.2. Схема работы системы состоит: 1) из символов данных, указывающих на наличие данных (символы данных могут также указывать вид носителя данных); 2) символов процесса, указывающих операции, которые следует выполнить над данными, а также определяющих логический путь, которого следует придерживаться; 3) линейных символов, указывающих потоки данных между процессами и (или) носителями данных, а также поток управления между процессами; 4) специальных символов, используемых для облегчения написания и чтения блок-схемы.

2.4. Схема взаимодействия программ.

2.4.1. Схемы взаимодействия программ отображают путь активаций программ и взаимодействий с соответствующими данными. Каждая программа в схеме взаимодействия программ показывается только один раз (в схеме работы системы программа может изображаться более чем в одном потоке управления).

2.4.2. Схема взаимодействия программ состоит из: 1) символов данных, указывающих на наличие данных; 2) символов процесса, указывающих на операции, которые следует выполнить над данными; 3) линейных символов, отображающих поток между процессами и данными, а также инициации процессов; 4) специальных символов, используемых для облегчения написания и чтения схемы.

2.5. Схема ресурсов системы.

2.5.1. Схемы ресурсов системы отображают конфигурацию блоков данных и обрабатывающих блоков, которая требуется для решения задачи или набора задач.

2.5.2. Схема ресурсов системы состоит: 1) из символов данных, отображающих входные, выходные и запоминающие устройства вычислительной машины; 2) символов процесса, отображающих процессоры (центральные процессоры, каналы и т. д.); 3) линейных символов, отображающих передачу данных между устройствами ввода-вывода и процессорами, а также передачу управления между процессорами; 4) специальных символов, используемых для облегчения написания и чтения схемы.

Примеры выполнения схем приведены в приложении.

3. ОПИСАНИЕ СИМВОЛОВ

3.1. Символы данных.

3.1.1. Основные символы данных.

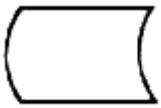
3.1.1.1. Данные.

Символ отображает данные, носитель данных не определен.



3.1.1.2. Запоминаемые данные.

Символ отображает хранимые данные в виде, пригодном для обработки, носитель данных не определен.



3.1.2. Специфические символы данных.

3.1.2.1. Оперативное запоминающее устройство

Символ отображает данные, хранящиеся в оперативном запоминающем устройстве.



3.1.2.2. Запоминающее устройство с последовательным доступом.

Символ отображает данные, хранящиеся в запоминающем устройстве с последовательным доступом (магнитная лента, кассета с магнитной лентой, магнитофонная кассета).



3.1.2.3. Запоминающее устройство с прямым доступом.

Символ отображает данные, хранящиеся в запоминающем устройстве с прямым доступом (магнитный диск, магнитный барабан, гибкий магнитный диск).



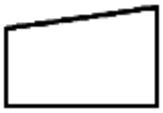
3.1.2.4. Документ.

Символ отображает данные, представленные на носителе в удобочитаемой форме (машинограмма, документ для оптического или магнитного считывания, микрофильм, рулон ленты с итоговыми данными, бланки ввода данных).



3.1.2.5. Ручной ввод.

Символ отображает данные, вводимые вручную во время обработки с устройств любого типа (клавиатура, переключатели, кнопки, световое перо, полоски со штриховым кодом).



3.1.2.6. Карта.

Символ отображает данные, представленные на носителе в виде карты (перфокарты, магнитные карты, карты со считываемыми метками, карты с отрывным ярлыком, карты со сканируемыми метками).



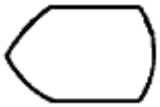
3.1.2.7. Бумажная лента.

Символ отображает данные, представленные на носителе в виде бумажной ленты.



3.1.2.8. Дисплей.

Символ отображает данные, представленные в человекочитаемой форме на носителе в виде отображающего устройства (экран для визуального наблюдения, индикаторы ввода информации).

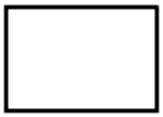


3.2. Символы процесса.

3.2.1. Основные символы процесса.

3.2.1.1. Процесс.

Символ отображает функцию обработки данных любого вида (выполнение определенной операции или группы операций, приводящее к изменению значения, формы или размещения информации или к определению, по которому из нескольких направлений потока следует двигаться).



3.2.2. Специфические символы процесса.

3.2.2.1. Предопределенный процесс.

Символ отображает предопределенный процесс, состоящий из одной или нескольких операций или шагов программы, которые определены в другом месте (в подпрограмме, модуле).



3.2.2.2. Ручная операция.

Символ отображает любой процесс, выполняемый человеком.



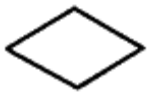
3.2.2.3. Подготовка.

Символ отображает модификацию команды или группы команд с целью воздействия на некоторую последующую функцию (установка переключателя, модификация индексного регистра или инициализация программы).



3.2.2.4. Решение.

Символ отображает решение или функцию переключательного типа, имеющую один вход и ряд альтернативных выходов, один и только один из которых может быть активизирован после вычисления условий, определенных внутри этого символа. Соответствующие результаты вычисления могут быть записаны по соседству с линиями, отображающими эти пути.

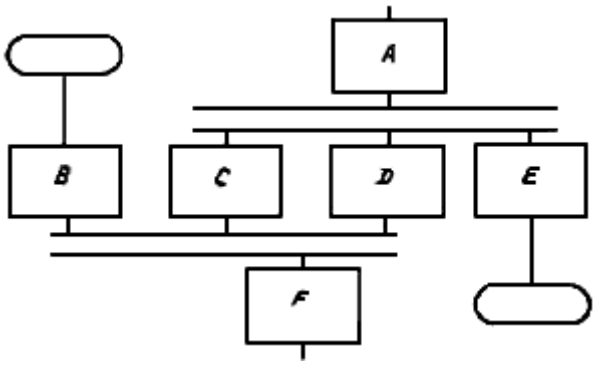


3.2.2.5. Параллельные действия.

Символ отображает синхронизацию двух или более параллельных операций.



Пример

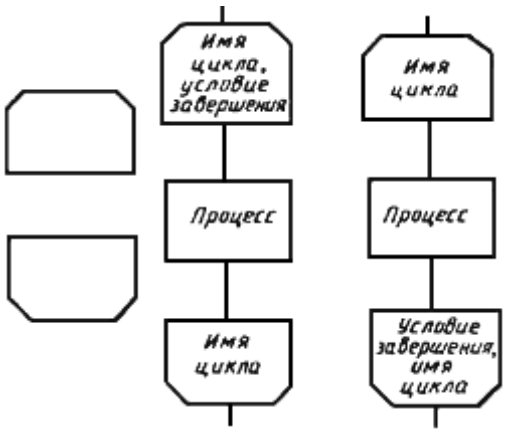


Примечание. Процессы C, D и E не могут начаться до тех пор, пока не завершится процесс A; аналогично процесс F должен ожидать завершения процессов B, C и D, однако процесс C может начаться и (или) завершиться прежде, чем соответственно начнется и (или) завершится процесс D.

3.2.2.6. Граница цикла.

Символ, состоящий из двух частей, отображает начало и конец цикла. Обе части символа имеют один и тот же идентификатор. Условия для инициализации, приращения, завершения и т. д. помещаются внутри символа в начале или в конце в зависимости от расположения операции, проверяющей условие.

Пример



3.3. Символы линий.

3.3.1. Основной символ линий.

3.3.1.1. Линия.

Символ отображает поток данных или управления.

При необходимости или для повышения удобочитаемости могут быть добавлены стрелки-указатели.

3.3.2. Специфические символы линий.

3.3.2.1. Передача управления.

Символ отображает непосредственную передачу управления от одного процесса к другому, иногда с возможностью прямого возвращения к инициирующему процессу после того, как инициированный процесс завершит свои функции. Тип передачи управления должен быть назван внутри символа (например, запрос, вызов, событие).



3.3.2.2. Канал связи.

Символ отображает передачу данных по каналу связи.

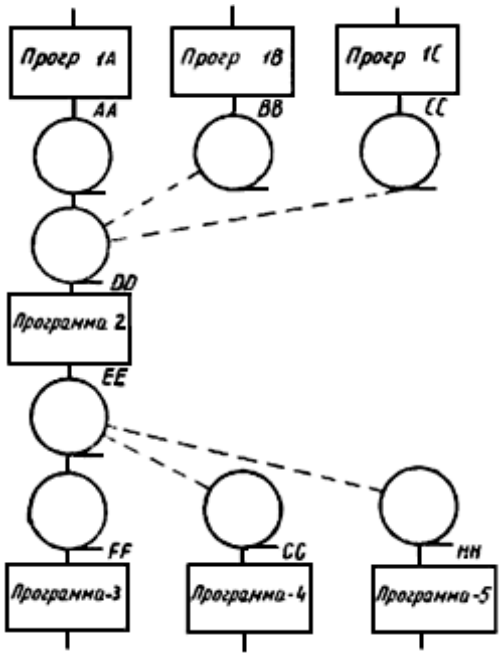


3.3.2.3. Пунктирная линия.

Символ отображает альтернативную связь между двумя или более символами. Кроме того, символ используют для обведения аннотированного участка.

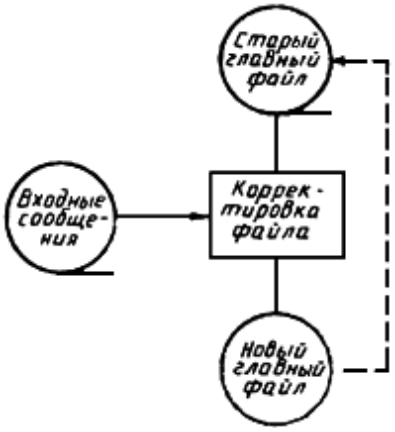
Пример 1

Если один из ряда альтернативных выходов используют в качестве входа в процесс либо когда выход используется в качестве входа в альтернативные процессы, эти символы соединяют пунктирными линиями.



Пример 2

Выход, используемый в качестве входа в следующий процесс, может быть соединен с этим входом с помощью пунктирной линии.



3.4. Специальные символы.

3.4.1. Соединитель.

Символ отображает выход в часть схемы и вход из другой части этой схемы и используется для обрыва линии и продолжения ее в другом месте. Соответствующие символы-соединители должны содержать одно и то же уникальное обозначение.



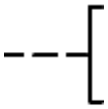
3.4.2. Терминатор.

Символ отображает выход во внешнюю среду и вход из внешней среды (начало или конец схемы программы, внешнее использование и источник или пункт назначения данных).

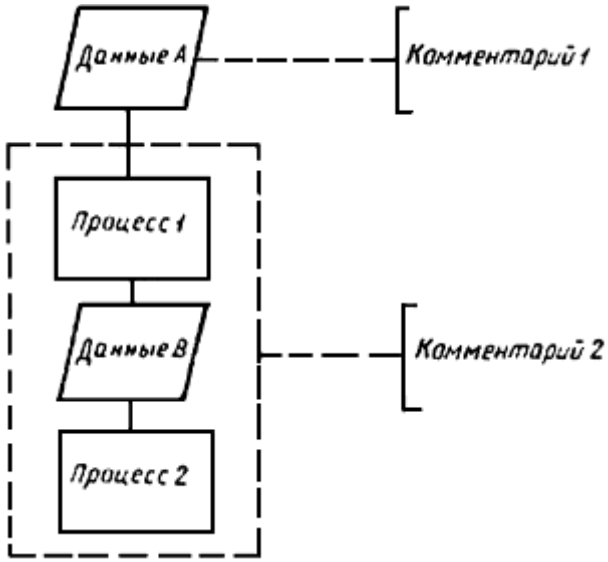


3.4.3. Комментарий.

Символ используют для добавления описательных комментариев или пояснительных записей в целях объяснения или примечаний. Пунктирные линии в символе комментария связаны с соответствующим символом или могут обводить группу символов. Текст комментариев или примечаний должен быть помещен около ограничивающей фигуры.

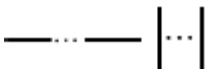


Пример

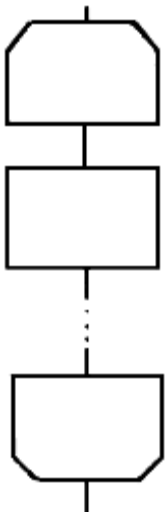


3.4.4. Пропуск.

Символ (три точки) используют в схемах для отображения пропуска символа или группы символов, в которых не определены ни тип, ни число символов. Символ используют только в символах линии или между ними. Он применяется главным образом в схемах, изображающих общие решения с неизвестным числом повторений.



Пример



4. ПРАВИЛА ПРИМЕНЕНИЯ СИМВОЛОВ И ВЫПОЛНЕНИЯ СХЕМ

4.1. Правила применения символов.

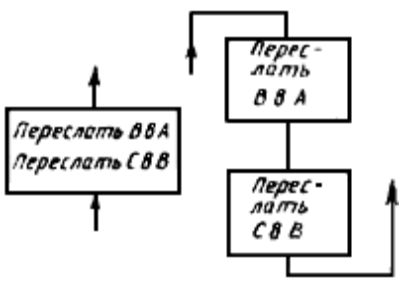
4.1.1. Символ предназначен для графической идентификации функции, которую он отображает, независимо от текста внутри этого символа.

4.1.2. Символы в схеме должны быть расположены равномерно. Следует придерживаться разумной длины соединений и минимального числа длинных линий.

4.1.3. Большинство символов задумано так, чтобы дать возможность включения текста внутри символа. Формы символов, установленные настоящим стандартом, должны служить руководством для фактически используемых символов. Не должны изменяться углы и другие параметры, влияющие на соответствующую форму символов. Символы должны быть, по возможности, одного размера. Символы могут быть вычерчены в любой ориентации, но, по возможности, предпочтительной является горизонтальная ориентация. Зеркальное изображение формы символа обозначает одну и ту же функцию, но не является предпочтительным.

4.1.4. Минимальное количество текста, необходимого для понимания функции данного символа, следует помещать внутри данного символа. Текст для чтения должен записываться слева направо и сверху вниз независимо от направления потока.

Пример

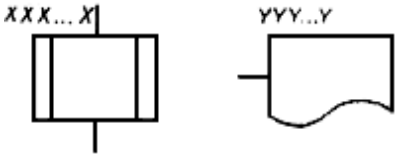


Если объем текста, помещаемого внутри символа, превышает его размеры, следует использовать символ комментария.

Если использование символов комментария может запутать или разрушить ход схемы, текст следует помещать на отдельном листе и давать перекрестную ссылку на символ.

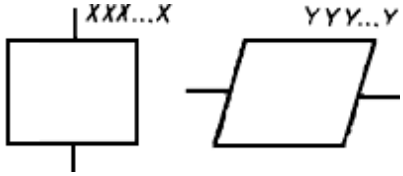
4.1.5. В схемах может использоваться идентификатор символов. Это связанный с данным символом идентификатор, который определяет символ для использования в справочных целях в других элементах документации (например, в листинге программы). Идентификатор символа должен располагаться слева над символом.

Пример



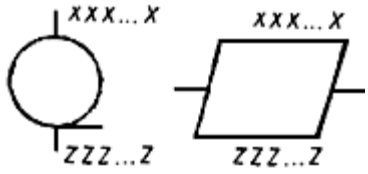
4.1.6. В схемах может использоваться описание символов – любая другая информация, например для отображения специального применения символа с перекрестной ссылкой или для улучшения понимания функции как части схемы. Описание символа должно быть расположено справа над символом.

Пример



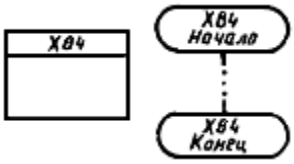
4.1.7. В схемах работы системы символы, отображающие носители данных, во многих случаях представляют способы ввода-вывода. Для использования в качестве ссылки на документацию текст на схеме для символов, отображающих способы вывода, должен размещаться справа над символом, а текст для символов, отображающих способы ввода, – справа под символом.

Пример



4.1.8. В схемах может использоваться подробное представление, которое обозначается с помощью символа с полосой для процесса или данных. Символ с полосой указывает, что в этом же комплекте документации в другом месте имеется более подробное представление. Символ с полосой представляет собой любой символ, внутри которого в верхней части проведена горизонтальная линия. Между этой линией и верхней линией символа помещен идентификатор, указывающий на подробное представление данного символа. В качестве первого и последнего символа подробного представления должен быть использован символ указателя конца. Первый символ указателя конца должен содержать ссылку, которая имеется также в символе с полосой.

Символ с полосой **Подробное представление**

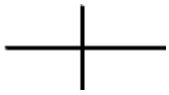


4.2. Правила выполнения соединений.

4.2.1. Потоки данных или потоки управления в схемах показываются линиями. Направление потока слева направо и сверху вниз считается стандартным. В случаях, когда необходимо внести большую ясность в схему (например, при соединениях), на линиях используются стрелки. Если поток имеет направление, отличное от стандартного, стрелки должны указывать это направление.

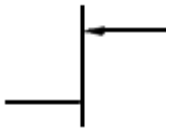
4.2.2. В схемах следует избегать пересечения линий. Пересекающиеся линии не имеют логической связи между собой, поэтому изменения направления в точках пересечения не допускаются.

Пример



4.2.3. Две или более входящие линии могут объединяться в одну исходящую линию. Если две или более линии объединяются в одну линию, место объединения должно быть смещено.

Пример



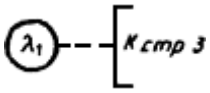
4.2.4. Линии в схемах должны подходить к символу либо слева, либо сверху, а исходить либо справа, либо снизу. Линии должны быть направлены к центру символа.

4.2.5. При необходимости линии в схемах следует разрывать для избегания излишних пересечений или слишком длинных линий, а также, если схема состоит из нескольких страниц. Соединитель в начале разрыва называется внешним соединителем, а соединитель в конце разрыва – внутренним соединителем.

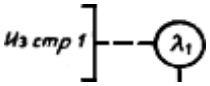
4.2.6. Ссылки к страницам могут быть приведены совместно с символом комментария для их соединителей.

Пример

Внешний соединитель



Внутренний соединитель

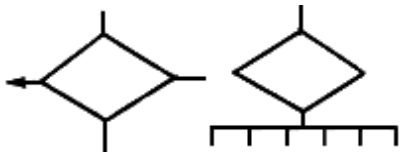


4.3. Специальные условные обозначения.

4.3.1. Несколько выходов.

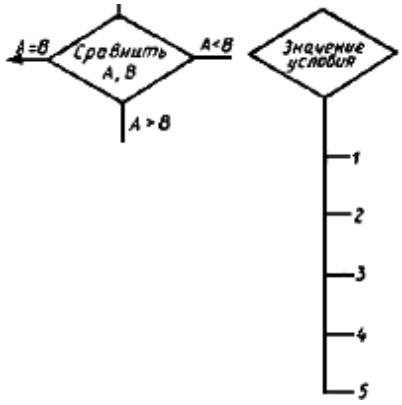
4.3.1.1. Несколько выходов из символа следует показывать: 1) несколькими линиями от данного символа к другим символам; 2) одной линией от данного символа, которая затем разветвляется в соответствующее число линий.

Примеры



4.3.1.2. Каждый выход из символа должен сопровождаться соответствующими значениями условий, чтобы показать логический путь, который он представляет, с тем чтобы эти условия и соответствующие ссылки были идентифицированы.

Примеры



4.3.2. Повторяющееся представление.

4.3.2.1. Вместо одного символа с соответствующим текстом могут быть использованы несколько символов с перекрытием изображения, каждый из которых содержит описательный текст (использование или формирование нескольких носителей данных или файлов, производство множества копий печатных отчетов или форматов перфокарт).

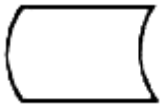
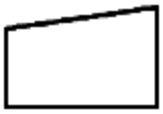
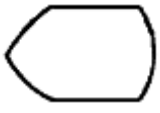
4.3.2.2. Когда несколько символов представляют упорядоченное множество, это упорядочение должно располагаться от переднего (первого) к заднему (последнему).

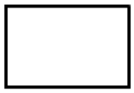
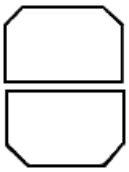
4.3.2.3. Линии могут входить или исходить из любой точки перекрытых символов, однако требования п.4.2.4 должны соблюдаться. Приоритет или последовательный порядок нескольких символов не изменяется посредством точки, в которой линия входит или из которой исходит.

Пример



5. ПРИМЕНЕНИЕ СИМВОЛОВ

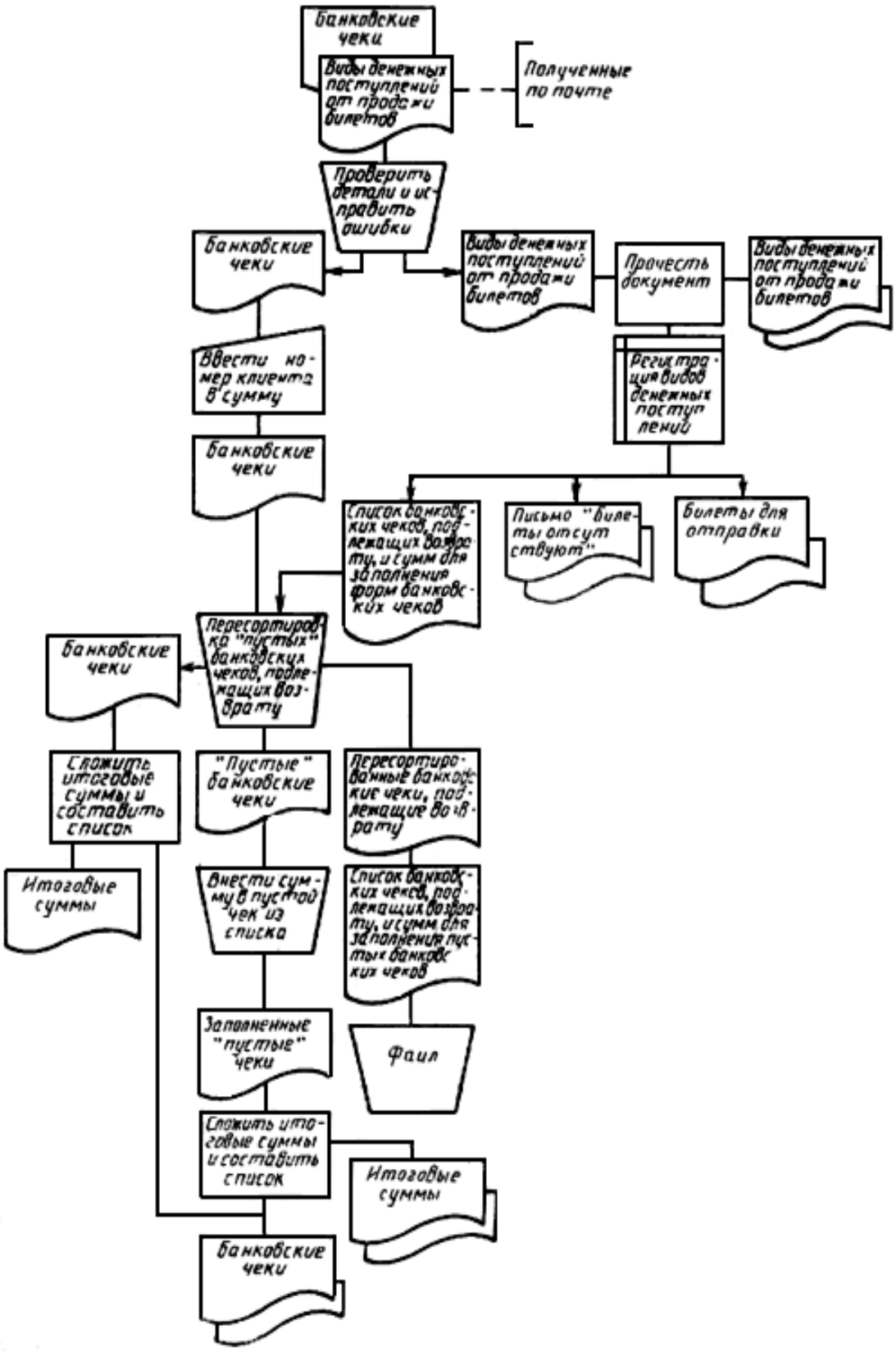
Символ	Наименование символа	Схема данных	Схема программы	Схема работы системы	Схема взаимодействия программ	Схема ресурсов системы
Символы данных *Основные*						
	Данные	+	+	+	+	+
	Запоминаемые данные	+	–	+	+	+
Специфические						
	Оперативное запоминающее устройство	+	–	+	+	+
	Запоминающее устройство с последовательной выборкой	+	–	+	+	+
	Запоминающее устройство с прямым доступом	+	–	+	+	+
	Документ	+	–	+	+	+
	Ручной ввод	+	–	+	+	+
	Карта	+	–	+	+	+
	Бумажная лента	+	–	+	+	+
	Дисплей	+	–	+	+	+

Символ	Наименование символа	Схема данных	Схема программы	Схема работы системы	Схема взаимодействия программ	Схема ресурсов системы
Символы процесса *Основные*						
	Процесс	+	+	+	+	+
Специфические						
	Предопределенный процесс	–	+	+	+	–
	Ручная операция	+	–	+	+	–
	Подготовка	+	+	+	+	–
	Решение	–	+	+	–	–
	Параллельные действия	–	+	+	+	–
	Граница цикла	–	+	+	–	–
Символы линий *Основные*						
	Линия	+	+	+	+	+
Специфические						
	Передача управления	–	–	–	+	–
	Канал связи	+	–	+	+	+
	Пунктирная линия	+	+	+	+	+
Специальные символы						
	Соединитель	+	+	+	+	+
	Терминатор	+	+	+	–	–
	Комментарий	+	+	+	+	+
	Пропуск	+	+	+	+	+

Примечание. Знак «+» указывает, что символ используют в данной схеме, знак «–» – не используют.

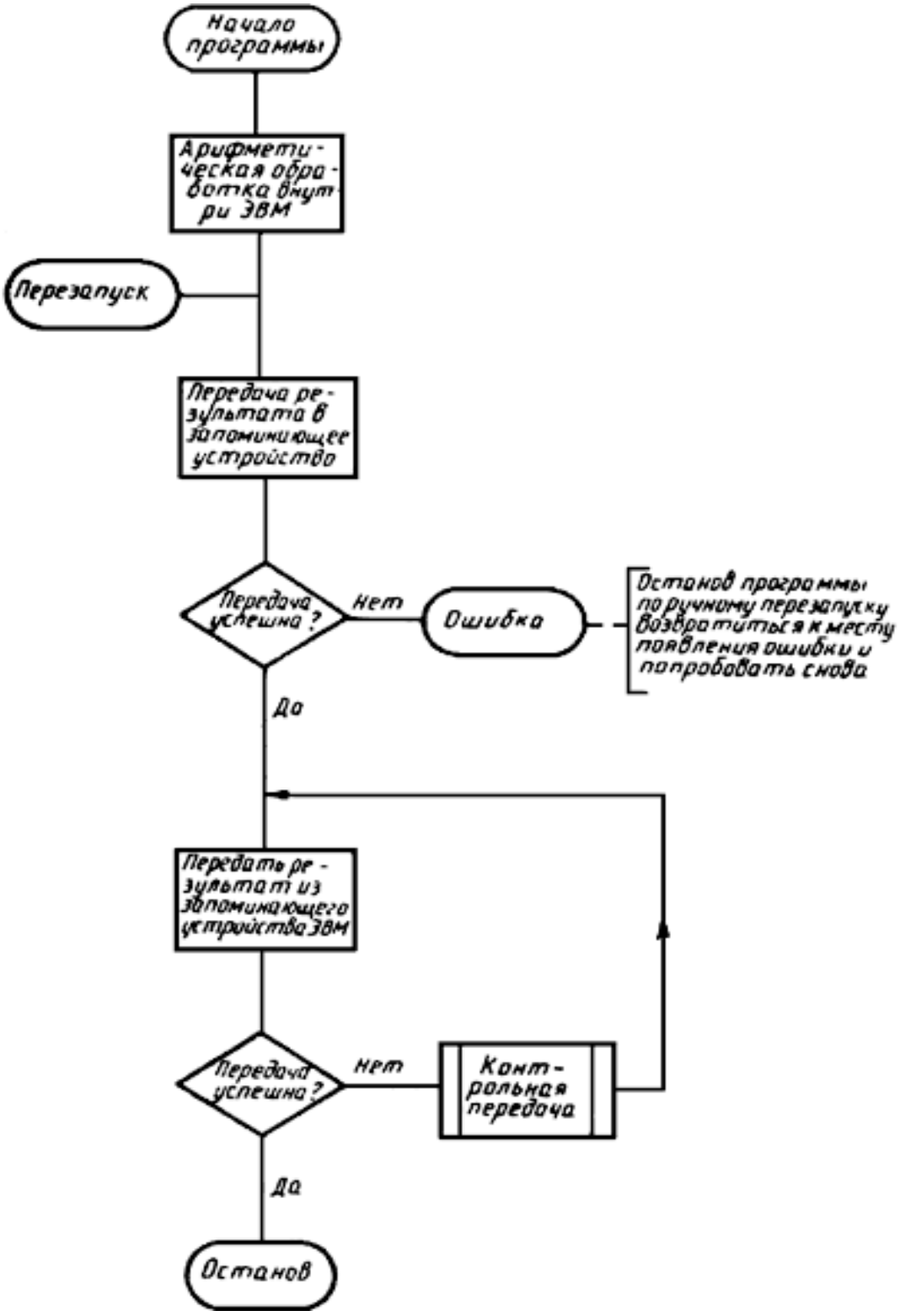
Приложение «Примеры выполнения схем»
Справочное

1. Схема данных

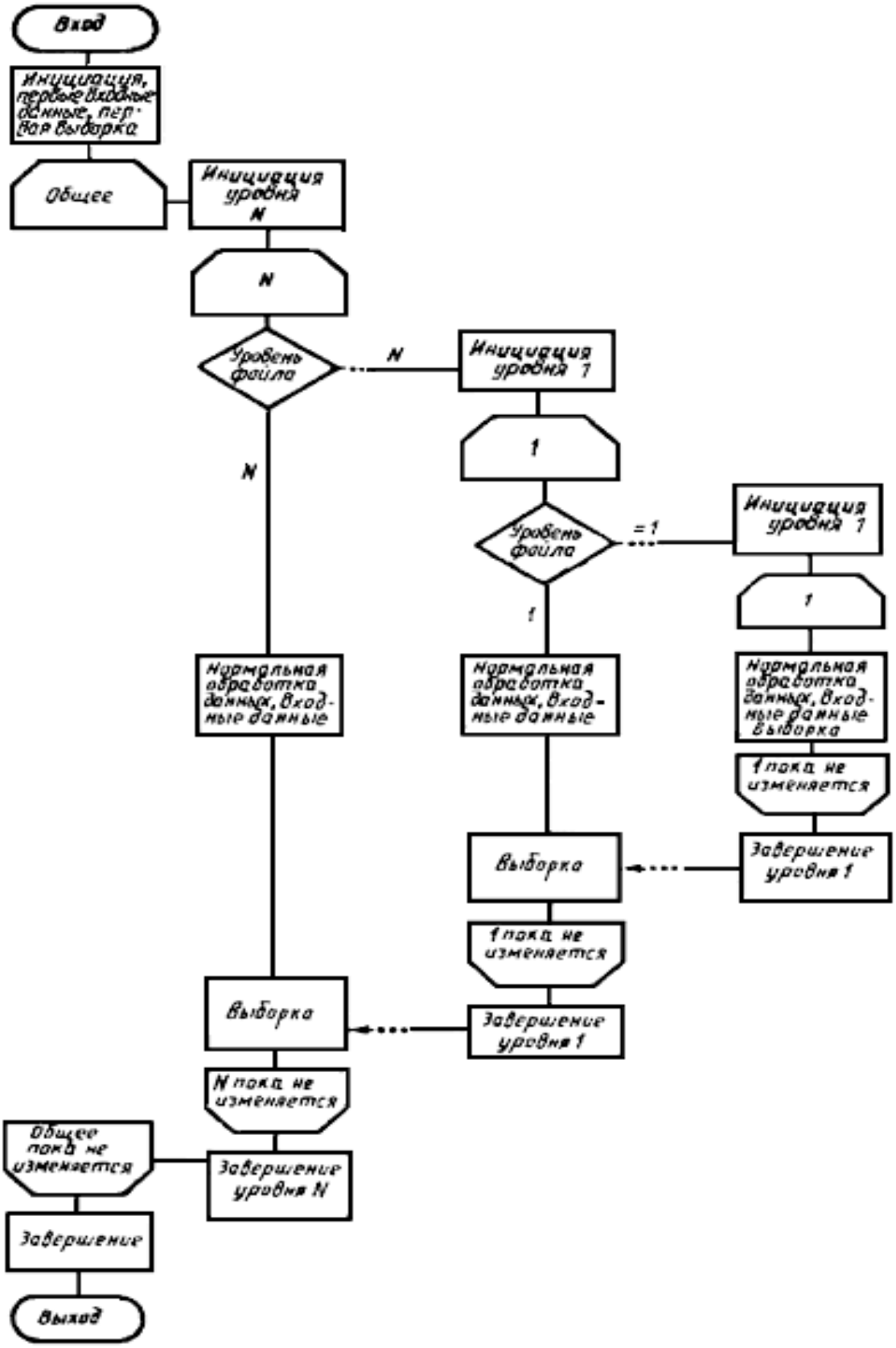


2. Схемы программы

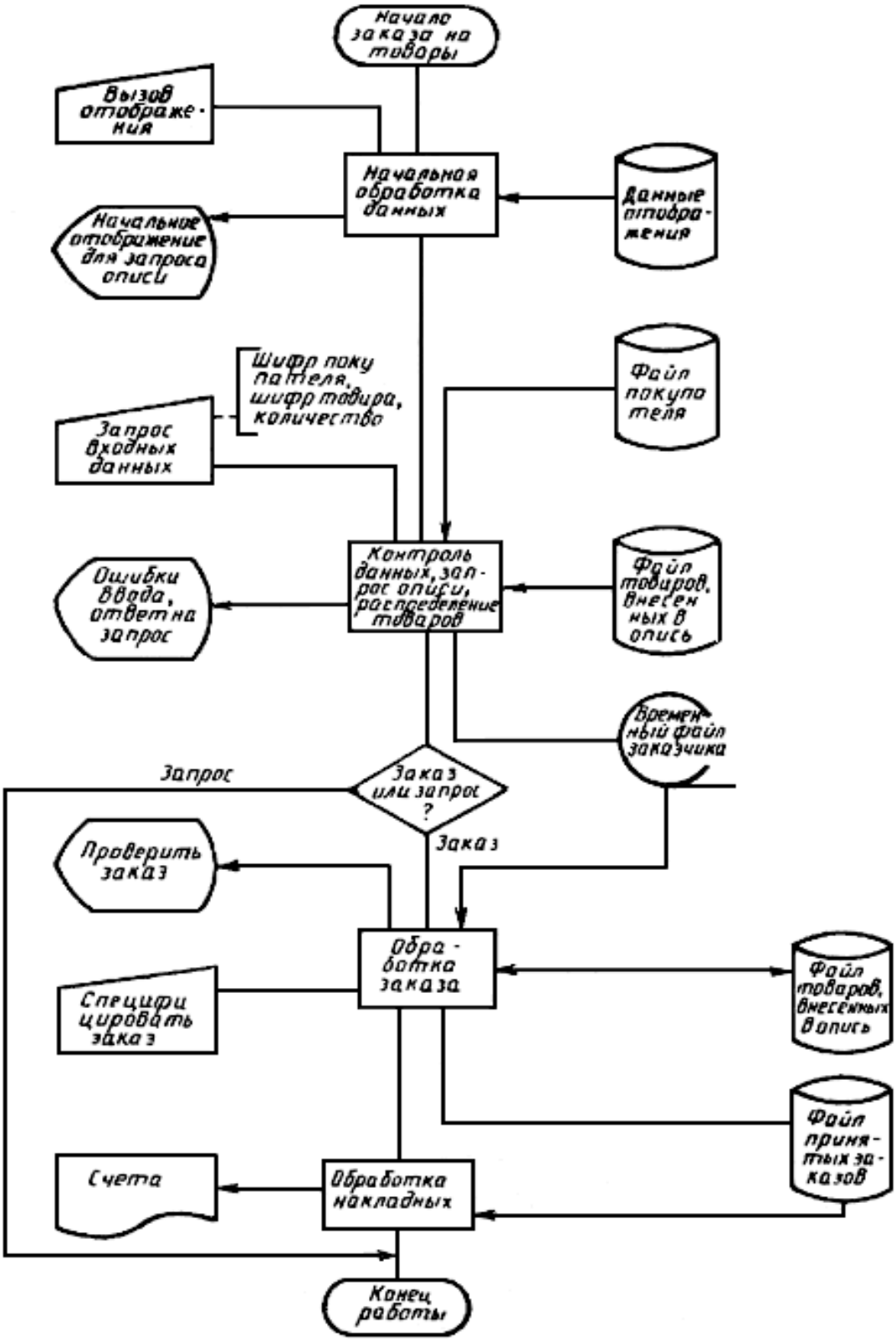
Пример 1



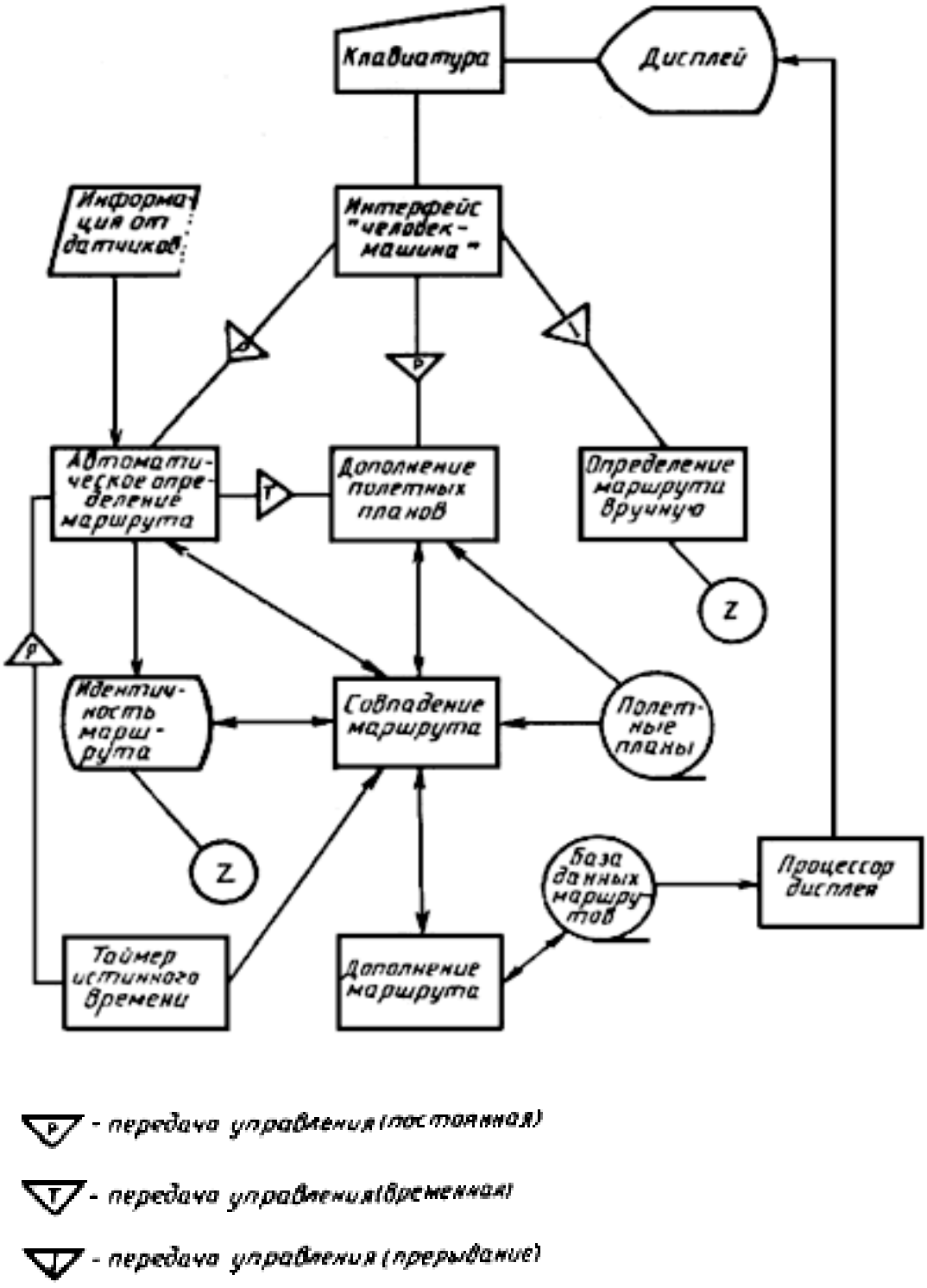
Пример 2


Вход
Инициация, первые входные данные, первая выборка
Общее
Инициация уровня N
N
Уровень файла
N
Инициация уровня 1
1
Уровень файла
=1
Инициация уровня 1
1
N
1
Нормальная обработка данных, входные данные
Нормальная обработка данных, входные данные
Нормальная обработка данных, входные данные выборка
1 пока не изменяется
Выборка
Завершение уровня 1
1 пока не изменяется
Выборка
Завершение уровня 1
N пока не изменяется
Общее пока не изменяется
Завершение уровня N
Завершение
Выход

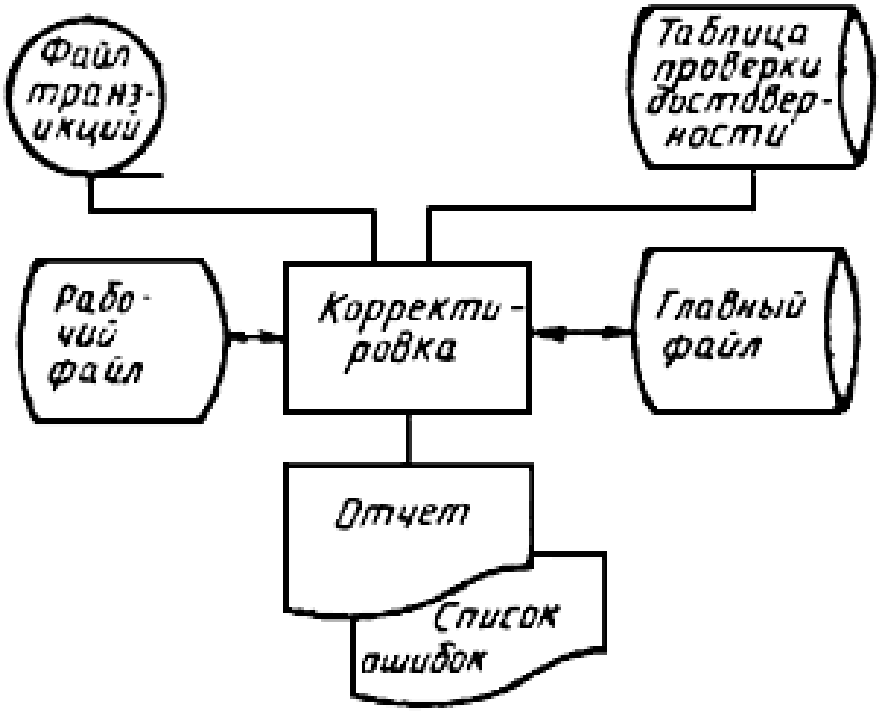
3. Схема работы системы



4. Схема взаимодействия программ



5. Схема ресурсов системы



ПРИЛОЖЕНИЕ Б

ГОСТ 34.601–90

Комплекс стандартов на автоматизированные системы.

Автоматизированные системы. Стадии создания

Настоящий стандарт распространяется на автоматизированные системы (АС), используемые в различных видах деятельности (исследование, проектирование, управление и т. п.), включая их сочетания, создаваемые в организациях, объединениях и на предприятиях (далее – организациях).

Стандарт устанавливает стадии и этапы создания АС. В приложении 1 приведено содержание работ на каждом этапе.

1. ОБЩИЕ ПОЛОЖЕНИЯ

1.1. Процесс создания АС представляет собой совокупность упорядоченных во времени, взаимосвязанных, объединённых в стадии и этапы работ, выполнение которых необходимо и достаточно для создания АС, соответствующей заданным требованиям.

1.2. Стадии и этапы создания АС выделяются как части процесса создания по соображениям рационального планирования и организации работ, заканчивающихся заданным результатом.

1.3. Работы по развитию АС осуществляют по стадиям и этапам, применяемым для создания АС.

1.4. Состав и правила выполнения работ на установленных настоящим стандартом стадиях и этапах определяют в соответствующей документации организаций, участвующих в создании конкретных видов АС.

Перечень организаций, участвующих в работах по созданию АС, приведён в приложении 2.

2. СТАДИИ И ЭТАПЫ СОЗДАНИЯ АС

2.1. Стадии и этапы создания АС в общем случае приведены в таблице. Б.2.1.

Таблица Б.2.1 – Стадии и этапы создания автоматизированной системы

Стадии	Этапы работ
1. Формирование требований к АС	1.1. Обследование объекта и обоснование необходимости создания АС. 1.2. Формирование требований пользователя к АС. 1.3. Оформление отчёта о выполненной работе и заявки на разработку АС (тактико-технического задания)

Окончание табл. Б.2.1

Стадии	Этапы работ
2. Разработка концепции АС	2.1. Изучение объекта. 2.2. Проведение необходимых научно-исследовательских работ. 2.3. Разработка вариантов концепции АС, удовлетворяющего требованиям пользователя. 2.4. Оформление отчёта о выполненной работе
3. Техническое задание	Разработка и утверждение технического задания на создание АС
4. Эскизный проект	4.1. Разработка предварительных проектных решений по системе и её частям. 4.2. Разработка документации на АС и её части
5. Технический проект	5.1. Разработка проектных решений по системе и её частям. 5.2. Разработка документации на АС и её части. 5.3. Разработка и оформление документации на поставку изделий для комплектования АС и (или) технических требований (технических заданий) на их разработку. 5.4. Разработка заданий на проектирование в смежных частях проекта объекта автоматизации
6. Рабочая документация	6.1. Разработка рабочей документации на систему и её части. 6.2. Разработка или адаптация программ
7. Ввод в действие	7.1. Подготовка объекта автоматизации к вводу АС в действие. 7.2. Подготовка персонала. 7.3. Комплектация АС поставляемыми изделиями (программными и техническими средствами, программно-техническими комплексами, информационными изделиями). 7.4. Строительно-монтажные работы. 7.5. Пусконаладочные работы. 7.6. Проведение предварительных испытаний. 7.7. Проведение опытной эксплуатации. 7.8. Проведение приёмочных испытаний
8. Сопровождение АС	8.1. Выполнение работ в соответствии с гарантийными обязательствами. 8.2. Послегарантийное обслуживание

2.2. Стадии этапы, выполняемые организациями-участниками работ по созданию АС, устанавливаются в договорах и техническом задании на основе настоящего стандарта.

Допускается исключить стадию «Эскизный проект» и отдельные этапы работ на всех стадиях, объединять стадии «Технический проект» и «Рабочая документация» в одну стадию «Технорабочий проект». В зависимости от специфики создаваемых АС и условий их создания допускается

выполнять отдельные этапы работ до завершения предшествующих стадий, параллельное во времени выполнение этапов работ, включение новых этапов работ.

ПРИЛОЖЕНИЕ 1 (справочное)

СОДЕРЖАНИЕ РАБОТ

1. На этапе 1.1. «Обследование объекта и обоснование необходимости создания в АС» общем случае проводят:

а) сбор данных об объекте автоматизации и осуществляемых видах деятельности;

б) оценку качества функционирования объекта и осуществляемых видах деятельности, выявление проблем, решение которых возможно средствами автоматизации;

в) оценку (технико-экономической, социальной и т. д.) целесообразности создания АС.

2. На этапе 1.2. «Формирование требований пользователя к АС» проводят:

а) подготовку исходных данных для формирования требований АС (характеристика объекта автоматизации, описание требований к системе, ограничения допустимых затрат на разработку, ввод в действие и эксплуатацию, эффект, ожидаемый от системы, условия создания и функционирования системы);

б) формулировку и оформление требований пользователя к АС.

3. На этапе 1.3. «Оформление отчёта о выполненной работе и заявки на разработку АС (технико-технического задания)» проводят оформление отчета о выполненных работах на данной стадии и оформление заявки на разработку АС (тактико-технического задания) или другого заменяющего её документа с аналогичным содержанием.

4. На этапах 2.1. «Изучение объекта» и 2.2. «Проведение научно-исследовательских работ» организация-разработчик проводит детальное изучение объекта автоматизации и необходимые научно-исследовательские работы (НИР), связанные с поиском путей и оценкой возможности реализации требований пользователя, оформляют и утверждают отчёты о НИР.

5. На этапе 2.3. «Разработка вариантов концепции АС и выбор варианта концепции АС, удовлетворяющего требованиям пользователя» в общем случае, проводят разработку альтернативных вариантов концепции создаваемой АС и планов их реализации; оценку необходимых ресурсов на их реализацию и обеспечение функционирования; оценку преимуществ и недостатков каждого варианта; определение порядка оценки качества и условий приёмки системы; оценку эффектов, получаемых от системы.

6. На этапе 2.4. «Оформление отчёта о выполненной работе» подготавливают и оформляют отчет, содержащий описание выполненных работ на стадии описания и обоснования предлагаемого варианта концепции системы.

7. На этапе 3.1. «Разработка и утверждение технического задания на создание АС» проводят разработку, оформление, согласование и утверждение технического задания на АС и, при необходимости, технических заданий на части АС.

8. На этапе 4.1. «Разработка предварительных проектных решений по системе и её частям» определяются: функции АС; функции подсистем, их цели и эффекты; состав комплексов задач и отдельных задач; концепция информационной базы, её укрупнённая структура; функции системы управления базой данных; состав вычислительной системы; функции и параметры основных программных средств.

9. На этапе 5.1. «Разработка проектных решений по системе и её частям» обеспечивают разработку общих решений по системе и её частям, функционально-алгоритмической структуре системы, по функциям персонала и организационной структуре, по структуре технических средств, по алгоритмам решения задач и применяемым языкам, по организации и ведению информационной базы, системе классификации и кодирования информации, по программному обеспечению.

10. На этапах 4.2. и 5.2. «Разработка документации на АС и её части» проводят разработку, оформление, согласование и утверждение документации в объёме, необходимом для описания полной совокупности принятых проектных решений и достаточном для дальнейшего выполнения работ по созданию АС. Виды документов – по ГОСТ 34.201–89.

11. На этапе 5.3. «Разработка и оформление документации на поставку изделий для комплектования АС и (или) технических требований (технических заданий) на их разработку» проводят: подготовку и оформление документации на поставку изделий для комплектования АС; определение технических требований и составление ТЗ на разработку изделий, не изготовляемых серийно.

12. На этапе 5.4 «Разработка заданий на проектирование в смежных частях проекта объекта автоматизации» осуществляют разработку, оформление, согласование и утверждение заданий на проектирование в смежных частях проекта объекта автоматизации для проведения строительных, электротехнических, санитарно-технических и других подготовительных работ, связанных с созданием АС.

13. На этапе 6.1 «Разработка рабочей документации на систему и её части» осуществляют разработку рабочей документации, содержащей все необходимые и достаточные сведения для обеспечения выполнения работ по вводу АС в действие и её эксплуатации, а также для поддержания уровня эксплуатационных характеристик (качества) системы в соответствии с принятыми проектными решениями, её оформление, согласование и утверждение. Виды документов по ГОСТ 34.201–89.

14. На этапе 6.2 «Разработка или адаптация программ» проводят разработку программ и программных средств системы, выбор, адаптацию

и (или) привязку приобретаемых программных средств, разработку программной документации в соответствии с ГОСТ 19.101.

15. На этапе 7.1 «Подготовка объекта автоматизации к вводу АС в действие» проводят работы по организационной подготовке объекта автоматизации к вводу АС в действие, в том числе:

- реализацию проектных решений по организационной структуре АС;
- обеспечение подразделений объекта управления инструктивно-методическими материалами;
- внедрение классификаторов информации.

16. На этапе 7.2 «Подготовка персонала» проводят обучение персонала и проверку его способности обеспечить функционирование АС.

17. На этапе 7.3 «Комплектация АС поставляемыми изделиями (программными и техническими средствами, программно-техническими комплексами, информационными изделиями)» обеспечивают получение комплектующих изделий серийного и единичного производства, материалов и монтажных изделий, проводят входной контроль их качества.

18. На этапе 7.4 «Строительно-монтажные работы» проводят:

- выполнение работ по строительству специализированных зданий (помещений) для размещения технических средств и персонала АС;
- сооружение кабельных каналов;
- выполнение работ по монтажу технических средств и линий связи;
- испытание смонтированных технических средств;
- сдачу технических средств для проведения пусконаладочных работ.

19. На этапе 7.5 «Пусконаладочные работы» проводят:

- автономную наладку технических и программных средств,
- загрузку информации в базу данных и проверку системы её ведения;
- комплексную наладку всех средств системы.

20. На этапе 7.6 «Проведение предварительных испытаний» осуществляют:

а) испытания АС на работоспособность и соответствие техническому заданию в соответствии с программой и методикой предварительных испытаний;

б) устранение неисправностей и внесение изменений в документацию на АС, в том числе эксплуатационную в соответствии с протоколом испытаний;

в) оформление акта о приёмке АС в опытную эксплуатацию.

21. На этапе 7.7 «Проведение опытной эксплуатации» проводят:

- опытную эксплуатацию АС;
- анализ результатов опытной эксплуатации АС;
- доработку (при необходимости) программного обеспечения АС;
- дополнительную наладку (при необходимости) технических средств АС;
- оформление акта о завершении опытной эксплуатации.

22. На этапе 7.8 «Проведение приёмочных испытаний» проводят:

а) испытания на соответствие техническому заданию в соответствии с программой и методикой приёмочных испытаний;

б) анализ результатов испытания АС и устранение недостатков, выявленных при испытаниях;

в) оформление акта о приёмке АС в постоянную эксплуатацию.

23. На этапе 8.1 «Выполнение работ в соответствии с гарантийными обязательствами» осуществляются работы по устранению недостатков, выявленных при эксплуатации АС в течение установленных гарантийных сроков, внесению необходимых изменений в документацию по АС.

24. На этапе 8.2 «Послегарантийное обслуживание» осуществляют работы:

а) по анализу функционирования системы;

б) выявлению отклонений фактических эксплуатационных характеристик АС от проектных значений;

в) установлению причин этих отклонений;

г) устранению выявленных недостатков и обеспечению стабильности эксплуатационных характеристик АС;

д) внесению необходимых изменений в документацию на АС.

ПРИЛОЖЕНИЕ (справочное) 2

ПЕРЕЧЕНЬ ОРГАНИЗАЦИЙ, УЧАСТВУЮЩИХ В РАБОТАХ ПО СОЗДАНИЮ АС

1. Организация-заказчик (пользователь), для которой создаются АС и которая обеспечивает финансирование, приемку работ и эксплуатацию АС, а также выполнение отдельных работ по созданию АС.

2. Организация-разработчик, которая осуществляет работы по созданию АС, представляет заказчику совокупность научно-технических услуг на разных стадиях и этапах создания, а также разрабатывает и поставляет различные программные и технические средства АС.

3. Организация-поставщик, которая изготавливает и поставляет программные и технические средства по заказу разработчика или заказчика.

4. Организация-генпроектировщик объекта автоматизации.

5. Организации-проектировщики различных частей проекта объекта автоматизации для проведения строительных, электротехнических, санитарно-технических и других подготовительных работ, связанных с созданием АС.

6. Организации строительные, монтажные, наладочные и другие.

Примечания:

а) в зависимости от условий создания АС возможны различные совмещения функций заказчика, разработчика, поставщика и других организаций, участвующих в работах по созданию АС;

б) стадии и этапы выполняемых ими работ по созданию АС определяются на основании настоящего стандарта.

ПРИЛОЖЕНИЕ В

ГОСТ 19.201–78

Техническое задание. Требования к содержанию и оформлению

Настоящий стандарт устанавливает порядок построения и оформления технического задания на разработку программы или программного изделия для вычислительных машин, комплексов и систем независимо от их назначения и области применения.

Стандарт полностью соответствует СТ СЭВ 1627–79.

1. ОБЩИЕ ПОЛОЖЕНИЯ

1.1. Техническое задание оформляют в соответствии с ГОСТ 19.106–78 на листах формата 11 и 12 по ГОСТ 2.301–68, как правило, без заполнения полей листа. Номера листов (страниц) проставляются в верхней части листа над текстом.

1.2. Лист утверждения и титульный лист оформляют в соответствии с ГОСТ 19.104–78.

Информационную часть (аннотацию и содержание), лист регистрации изменений допускается в документ не включать.

1.3. 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.

1.4. 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;
- порядок контроля и приемки;
- в техническое задание допускается включать приложения.

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

2. СОДЕРЖАНИЕ РАЗДЕЛОВ

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

2.2. В разделе «Основания для разработки» должны быть указаны:

- документ (документы), на основании которых ведется разработка;
- организация, утвердившая этот документ, и дата его утверждения;
- наименование и (или) условное обозначение темы разработки.

2.3. В разделе «Назначение разработки» должно быть указано функциональное и эксплуатационное назначение программы или программного изделия.

2.4. 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;
- требования к маркировке и упаковке;
- требования к транспортированию и хранению;
- специальные требования.

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

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

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

2.4.4. В подразделе «Требования к составу и параметрам технических средств» указывают необходимый состав технических средств с указанием их основных технических характеристик.

2.4.5. 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.

2.4.6. В подразделе «Требования к маркировке и упаковке» в общем случае указывают требования к маркировке программного изделия, варианты и способы упаковки.

2.4.7. В подразделе «Требования к транспортированию и хранению» должны быть указаны для программного изделия условия транспортирования, места хранения, условия хранения, условия складирования, сроки хранения в различных условиях.

2.5а. В разделе «Требования к программной документации» должен быть указан предварительный состав программной документации и, при необходимости, специальные требования к ней.

2.5. 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.

2.6. 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, сроки разработки и определяют исполнителей.

2.7. В разделе «Порядок контроля и приемки» должны быть указаны виды испытаний и общие требования к приемке работы.

2.8. В приложениях к техническому заданию, при необходимости, приводят:

- перечень научно-исследовательских и других работ, обосновывающих разработку;
- схемы алгоритмов, таблицы, описания, обоснования, расчеты и другие документы, которые могут быть использованы при разработке;

другие источники разработки.

ПРИЛОЖЕНИЕ Г

ГОСТ 7.32–2001

Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления

1 Область применения

Настоящий стандарт устанавливает общие требования к структуре и правилам оформления научных и технических отчетов, а также правила для тех случаев, когда единая процедура оформления будет содействовать обмену информацией, совершенствуя обработку отчета в информационной системе.

Настоящий стандарт распространяется на отчеты о фундаментальных, поисковых, прикладных научно-исследовательских работах (НИР) по всем областям науки и техники, выполняемых научно-исследовательскими, проектными, конструкторскими организациями, высшими учебными заведениями, научно-производственными и производственными объединениями, промышленными предприятиями, акционерными обществами и другими организациями.

Положения настоящего стандарта могут быть использованы при подготовке отчета в других сферах научной деятельности.

2 Общие положения

2.1 Отчет о НИР – научно-технический документ, который содержит систематизированные данные о научно-исследовательской работе, описывает состояние научно-технической проблемы, процесс и/или результаты научного исследования.

2.2 По результатам выполнения НИР составляется заключительный отчет о работе в целом. Кроме того, по отдельным этапам НИР могут быть составлены промежуточные отчеты, что отражается в Техническом задании на НИР и в календарном плане выполнения НИР.

2.3 Ответственность за достоверность данных, содержащихся в отчете, и за соответствие его требованиям настоящего стандарта несет организация-исполнитель.

2.4 Отчет о НИР подлежит обязательному нормоконтролю в организации-исполнителе. При проведении нормоконтроля рекомендуется руководствоваться ГОСТ 2.111.

3 Структурные элементы отчета

Структурными элементами отчета о НИР являются:

– титульный лист;
– список исполнителей;
– реферат;
– содержание;
– определения;
– обозначения и сокращения;
– введение;
– основная часть;
– заключение;
– список использованных источников;
– приложения.

Обязательные структурные элементы выделены полужирным шрифтом. Остальные структурные элементы включают в отчет по усмотрению исполнителя НИР с учетом требований 4 и 5.

4 Требования к содержанию структурных элементов отчета

4.1 Титульный лист

4.1.1 Титульный лист является первой страницей отчета о НИР и служит источником информации, необходимой для обработки и поиска документа.

4.1.2 На титульном листе приводят следующие сведения:

– наименование вышестоящей организации;
– наименование организации-исполнителя НИР;
– индекс Универсальной десятичной классификации (УДК);
– коды Высших классификационных группировок Общероссийского классификатора промышленной и сельскохозяйственной продукции для НИР (ВКГОКП), предшествующих постановке продукции на производство;
– номера, идентифицирующие отчет;
– грифы согласования и утверждения;
– наименование работы;
– наименование отчета;
– вид отчета (заключительный, промежуточный);
– номер (шифр) работы;
– должности, ученые степени, ученые звания, фамилии и инициалы руководителей организации-исполнителя НИР, руководителей НИР;
– место и дату составления отчета.

4.1.3 Если отчет о НИР состоит из двух и более частей, то каждая часть должна иметь свой титульный лист, соответствующий титульному листу первой части и содержащий сведения, относящиеся к данной части.

4.1.4 Титульный лист следует оформлять в соответствии с 5.10.

4.2 Список исполнителей

4.2.1 В список исполнителей должны быть включены фамилии и инициалы, должности, ученые степени, ученые звания руководителей НИР, ответственных исполнителей, исполнителей и соисполнителей, принимавших творческое участие в выполнении работы.

4.2.2 Если отчет выполнен одним исполнителем, то его должность, ученую степень, ученое звание, фамилию и инициалы следует указывать на титульном листе отчета.

4.2.3 Список исполнителей следует оформлять в соответствии с 5.11.

4.3 Реферат

4.3.1 Общие требования к реферату на отчет о НИР – по ГОСТ 7.9.

4.3.2 Реферат должен содержать:

– сведения об объеме отчета, количестве иллюстраций, таблиц, приложений, количестве частей отчета, количестве использованных источников;

– перечень ключевых слов;

– текст реферата.

4.3.2.1 Перечень ключевых слов должен включать от 5 до 15 слов или словосочетаний из текста отчета, которые в наибольшей мере характеризуют его содержание и обеспечивают возможность информационного поиска. Ключевые слова приводятся в именительном падеже и печатаются прописными буквами в строку через запятые.

4.3.2.2 Текст реферата должен отражать:

– объект исследования или разработки;

– цель работы;

– метод или методологию проведения работы;

– результаты работы и их новизну;

– основные конструктивные, технологические и технико-эксплуатационные характеристики;

– степень внедрения;

– рекомендации по внедрению или итоги внедрения результатов НИР;

– область применения;

– экономическую эффективность или значимость работы;

– прогнозные предположения о развитии объекта исследования.

Если отчет не содержит сведений по какой-либо из перечисленных структурных частей реферата, то в тексте реферата она опускается, при этом последовательность изложения сохраняется.

4.4 Содержание (оглавление)

4.4.1 Содержание включает введение, наименование всех разделов, подразделов, пунктов (если они имеют наименование), заключение, список

использованных источников и наименование приложений с указанием номеров страниц, с которых начинаются эти элементы отчета о НИР.

4.4.2 При составлении отчета, состоящего из двух и более частей, в каждой из них должно быть свое содержание. При этом в первой части помещают содержание всего отчета с указанием номеров частей, в последующих – только содержание соответствующей части. Допускается в первой части вместо содержания последующих частей указывать только их наименования.

4.4.3 В отчете о НИР объемом не более 10 страниц содержание допускается не составлять.

4.5 Определения

4.5.1 Структурный элемент «Определения» содержит определения, необходимые для уточнения или установления терминов, используемых в НИР.

4.5.2 Перечень определений начинают со слов: «В настоящем отчете о НИР применяют следующие термины с соответствующими определениями».

4.6 Обозначения и сокращения

4.6.1 Структурный элемент «Обозначения и сокращения» содержит перечень обозначений и сокращений, применяемых в данном отчете о НИР.

4.6.2 Допускается определения, обозначения и сокращения приводить в одном структурном элементе «Определения, обозначения и сокращения».

4.7 Введение

4.7.1 Введение должно содержать оценку современного состояния решаемой научно-технической проблемы, основание и исходные данные для разработки темы, обоснование необходимости проведения НИР, сведения о планируемом научно-техническом уровне разработки, о патентных исследованиях и выводы из них, сведения о метрологическом обеспечении НИР. Во введении должны быть показаны актуальность и новизна темы, связь данной работы с другими научно-исследовательскими работами.

4.7.2 Во введении промежуточного отчета по этапу НИР должны быть приведены цели и задачи этапа исследований, их место в выполнении НИР в целом.

4.7.3 Во введении заключительного отчета о НИР помещают перечень наименований всех подготовленных промежуточных отчетов по этапам и их инвентарные номера.

4.8 Основная часть

4.8.1 В основной части отчета приводят данные, отражающие сущность, методику и основные результаты выполненной НИР.

4.8.2 Основная часть должна содержать:

а) выбор направления исследований, включающий обоснование направления исследования, методы решения задач и их сравнительную оценку, описание выбранной общей методики проведения НИР;

б) процесс теоретических и (или) экспериментальных исследований, включая определение характера и содержания теоретических исследований, методы исследований, методы расчета, обоснование необходимости проведения экспериментальных работ, принципы действия разработанных объектов, их характеристики;

в) обобщение и оценку результатов исследований, включающих оценку полноты решения поставленной задачи и предложения по дальнейшим направлениям работ, оценку достоверности полученных результатов и технико-экономической эффективности их внедрения и их сравнение с аналогичными результатами отечественных и зарубежных работ, обоснование необходимости проведения дополнительных исследований, отрицательные результаты, приводящие к необходимости прекращения дальнейших исследований.

4.8.3 Представление в отчете данных о свойствах веществ и материалов проводятся по ГОСТ 7.54, единицы физических величин – по ГОСТ 8.417.

4.9 Заключение

Заключение должно содержать:

– краткие выводы по результатам НИР или отдельных ее этапов;

– оценку полноты решений поставленных задач;

– разработку рекомендаций и исходных данных по конкретному использованию результатов НИР;

– результаты оценки технико-экономической эффективности внедрения;

– результаты оценки научно-технического уровня выполненной НИР в сравнении с лучшими достижениями в данной области.

4.10 Список использованных источников

Список должен содержать сведения об источниках, использованных при составлении отчета. Сведения об источниках приводятся в соответствии с требованиями ГОСТ 7.1.

4.11 Приложения

4.11.1 В приложения рекомендуется включать материалы, связанные с выполненной НИР, которые по каким-либо причинам не могут быть включены в основную часть.

В приложения могут быть включены:

– промежуточные математические доказательства, формулы и расчеты;

– таблицы вспомогательных цифровых данных;

– протоколы испытаний;

– описание аппаратуры и приборов, применяемых при проведении экспериментов, измерений и испытаний;

– заключение метрологической экспертизы;

– инструкции, методики, разработанные в процессе выполнения НИР;

– иллюстрации вспомогательного характера;

– копии технического задания на НИР, программы работ, договора или другого исходного документа для выполнения НИР;

– протокол рассмотрения выполненной НИР на научно-техническом совете;

– акты внедрения результатов НИР и др.

4.11.2 В приложения к отчету о НИР, предшествующему постановке продукции на производство, должен быть включен проект технического задания на разработку (модернизацию) продукции или документ (заявка, протокол, контракт и др.), содержащий обоснованные технико-экономические требования к продукции.

4.11.3 В приложения к отчету о НИР, в составе которой предусмотрено проведение патентных исследований, должен быть включен отчет о патентных исследованиях, оформленный по ГОСТ 15.011, библиографический список публикаций и патентных документов, полученных в результате выполнения НИР, – по ГОСТ 7.1.

4.11.4 Приложения следует оформлять в соответствии с 5.14.

5 Правила оформления отчета

5.1 Общие требования

5.1.1 Изложение текста и оформление отчета выполняют в соответствии с требованиями настоящего стандарта. Страницы текста отчета о НИР и включенные в отчет иллюстрации и таблицы должны соответствовать формату А4 по ГОСТ 9327. Допускается применение формата А3 при наличии большого количества таблиц и иллюстраций данного формата.

5.1.2 Отчет о НИР должен быть выполнен любым печатным способом на пишущей машинке или с использованием компьютера и принтера на одной стороне листа белой бумаги формата А4 через полтора интервала. Цвет шрифта должен быть черным, высота букв, цифр и других знаков – не менее 1,8 мм (кегль не менее 12). Полужирный шрифт не применяется.

Текст отчета следует печатать, соблюдая следующие размеры полей: правое – не менее 10 мм, верхнее и нижнее – не менее 20 мм, левое – не менее 30 мм.

Разрешается использовать компьютерные возможности акцентирования внимания на определенных терминах, формулах, теоремах, применяя шрифты разной гарнитуры.

5.1.3 Вне зависимости от способа выполнения отчета качество напечатанного текста и оформления иллюстраций, таблиц, распечаток с ПЭВМ должно удовлетворять требованию их четкого воспроизведения.

5.1.4 При выполнении отчета необходимо соблюдать равномерную плотность, контрастность и четкость изображения по всему отчету. В отчете должны быть четкие, не расплывшиеся линии, буквы, цифры и знаки.

5.1.5 Опечатки, описки и графические неточности, обнаруженные в процессе подготовки отчета, допускается исправлять подчисткой или закрашиванием белой краской и нанесением на том же месте исправленного текста (графики) машинописным способом или черными чернилами, пастой или тушью – рукописным способом.

Повреждения листов отчета, помарки и следы не полностью удаленного прежнего текста (графики) не допускаются.

После внесения исправлений документ должен удовлетворять требованиям микрофильмирования, установленным ГОСТ 13.1.002.

5.1.6 Фамилии, названия учреждений, организаций, фирм, название изделий и другие имена собственные в отчете приводят на языке оригинала. Допускается транслитерировать имена собственные и приводить названия организаций в переводе на язык отчета с добавлением (при первом упоминании) оригинального названия.

5.1.7 Сокращение русских слов и словосочетаний в отчете – по ГОСТ 7.12.

5.2 Построение отчета

5.2.1 Наименования структурных элементов отчета «Список исполнителей», «Реферат», «Содержание», «Определения», «Обозначения и сокращения», «Введение», «Заключение», «Список использованных источников», «Приложение» служат заголовками структурных элементов отчета. Заголовки структурных элементов следует располагать в середине строки без точки в конце и печатать прописными буквами, не подчеркивая.

5.2.2 Основную часть отчета следует делить на разделы, подразделы и пункты. Пункты, при необходимости, могут делиться на подпункты. При делении текста отчета на пункты и подпункты необходимо, чтобы каждый пункт содержал законченную информацию.

5.2.3 Разделы, подразделы, пункты и подпункты следует нумеровать арабскими цифрами и записывать с абзацного отступа.

Разделы должны иметь порядковую нумерацию в пределах всего текста, за исключением приложений. Пример – 1, 2, 3 и т. д.

Номер подраздела или пункта включает номер раздела и порядковый номер подраздела или пункта, разделенные точкой. Пример – 1.1, 1.2, 1.3 и т. д.

Номер подпункта включает номер раздела, подраздела, пункта и порядковый номер подпункта, разделенные точкой. Пример – 1.1.1.1, 1.1.1.2, 1.1.1.3 и т. д.

После номера раздела, подраздела, пункта и подпункта в тексте точку не ставят. Если текст отчета подразделяют только на пункты, их следует нумеровать, за исключением приложений, порядковыми номерами в пределах всего отчета. Если раздел или подраздел имеет только один пункт или пункт имеет один подпункт, то нумеровать его не следует.

5.2.4 Разделы, подразделы должны иметь заголовки. Пункты, как правило, заголовков не имеют. Заголовки должны четко и кратко отражать содержание разделов, подразделов.

5.2.5 Заголовки разделов, подразделов и пунктов следует печатать с абзацного отступа с прописной буквы без точки в конце, не подчеркивая.

Если заголовок состоит из двух предложений, их разделяют точкой.

5.3 Нумерация страниц отчета

5.3.1 Страницы отчета следует нумеровать арабскими цифрами, соблюдая сквозную нумерацию по всему тексту отчета. Номер страницы проставляют в центре нижней части листа без точки,

5.3.2 Титульный лист включают в общую нумерацию страниц отчета. Номер страницы на титульном листе не проставляют.

5.3.3 Иллюстрации и таблицы, расположенные на отдельных листах, включают в общую нумерацию страниц отчета. Иллюстрации и таблицы на листе формата А3 учитывают, как одну страницу.

5.4 Нумерация разделов, подразделов, пунктов, подпунктов отчета

5.4.1 Разделы отчета должны иметь порядковые номера в пределах всего отчета, обозначенные арабскими цифрами без точки и записанные с абзацного отступа. Подразделы должны иметь нумерацию в пределах каждого раздела. Номер подраздела состоит из номеров раздела и подраздела, разделенных точкой. В конце номера подраздела точка не ставится. Разделы, как и подразделы, могут состоять из одного или нескольких пунктов.

5.4.2 Если отчет не имеет подразделов, то нумерация пунктов в нем должна быть в пределах каждого раздела, и номер пункта должен состоять из номеров раздела и пункта, разделенных точкой. В конце номера пункта точка не ставится.

Пример

1 Типы и основные размеры

1.1

1.2 } Нумерация пунктов первого раздела отчета

1.3

2 Технические требования

2.1

2.2 } Нумерация пунктов второго раздела отчета

Если отчет имеет подразделы, то нумерация пунктов должна быть в пределах подраздела и номер пункта должен состоять из номеров раздела, подраздела и пункта, разделенных точками, например:

3 Методы испытаний

3.1 Аппараты, материалы и реактивы

3.1.1

3.1.2 } Нумерация пунктов первого подраздела третьего раздела отчета

3.1.3

3.2 Подготовка к испытанию

3.2.1

3.2.2 } Нумерация пунктов второго подраздела третьего раздела отчета

3.2.3

5.4.3 Если раздел состоит из одного подраздела, то подраздел не нумеруется. Если подраздел состоит из одного пункта, то пункт не нумеруется.

5.4.4 Если текст отчета подразделяется только на пункты, то они нумеруются порядковыми номерами в пределах всего отчета.

5.4.5 Пункты, при необходимости, могут быть разбиты на подпункты, которые должны иметь порядковую нумерацию в пределах каждого пункта, например 4.2.1.1, 4.2.1.2, 4.2.1.3 и т. д.

5.4.6 Внутри пунктов или подпунктов могут быть приведены перечисления.

Перед каждым элементом перечисления следует ставить дефис. При необходимости ссылки в тексте отчета на один из элементов перечисления вместо дефиса ставятся строчные буквы в порядке русского алфавита, начиная с буквы а (за исключением букв е, з, й, о, ч, ъ, ы, ь).

Для дальнейшей детализации перечислений необходимо использовать арабские цифры, после которых ставится скобка, а запись производится с абзацного отступа, как показано в примере.

Пример

а) _____

б) _____

1) _____

2) _____

в) _____

5.4.7 Если отчет состоит из двух и более частей, каждая часть должна иметь свой порядковый номер. Номер каждой части следует проставлять

арабскими цифрами на титульном листе под указанием вида отчета, например, «Часть 2».

5.4.8 Каждый структурный элемент отчета следует начинать с нового листа (страницы).

5.4.9 Нумерация страниц отчета и приложений, входящих в состав отчета, должна быть сквозная.

5.5 Иллюстрации

5.5.1 Иллюстрации (чертежи, графики, схемы, компьютерные распечатки, диаграммы, фотоснимки) следует располагать в отчете непосредственно после текста, в котором они упоминаются впервые, или на следующей странице.

Иллюстрации могут быть в компьютерном исполнении, в том числе и цветные.

На все иллюстрации должны быть даны ссылки в отчете.

5.5.2 Чертежи, графики, диаграммы, схемы, иллюстрации, помещаемые в отчете, должны соответствовать требованиям государственных стандартов Единой системы конструкторской документации (ЕСКД).

Допускается выполнение чертежей, графиков, диаграмм, схем посредством использования компьютерной печати.

5.5.3 Фотоснимки размером меньше формата А4 должны быть наклеены на стандартные листы белой бумаги.

5.5.4 Иллюстрации, за исключением иллюстрации приложений, следует нумеровать арабскими цифрами сквозной нумерацией.

Если рисунок один, то он обозначается «Рисунок 1». Слово «рисунок» и его наименование располагают посередине строки.

5.5.5 Допускается нумеровать иллюстрации в пределах раздела. В этом случае номер иллюстрации состоит из номера раздела и порядкового номера иллюстрации, разделенных точкой. Например, Рисунок 1.1.

5.5.6 Иллюстрации, при необходимости, могут иметь наименование и пояснительные данные (подрисуночный текст). Слово «Рисунок» и наименование помещают после пояснительных данных и располагают следующим образом: Рисунок 1 – Детали прибора.

5.5.7 Иллюстрации каждого приложения обозначают отдельной нумерацией арабскими цифрами с добавлением перед цифрой обозначения приложения. Например, Рисунок А.3.

5.5.8 При ссылках на иллюстрации следует писать «... в соответствии с рисунком 2» при сквозной нумерации и «... в соответствии с рисунком 1.2» при нумерации в пределах раздела.

5.6 Таблицы

5.6.1 Таблицы применяют для лучшей наглядности и удобства сравнения показателей. Наименование таблицы, при его наличии, должно отражать ее содержание, быть точным, кратким. Наименование таблицы следует помещать над таблицей слева, без абзацного отступа в одну строку с ее номером через тире.

5.6.2 Таблицу следует располагать в отчете непосредственно после текста, в котором она упоминается впервые, или на следующей странице.

5.6.3 На все таблицы должны быть ссылки в отчете. При ссылке следует писать слово «таблица» с указанием ее номера.

5.6.4 Таблицу с большим числом строк допускается переносить на другой лист (страницу). При переносе части таблицы на другой лист (страницу) слово «Таблица», ее номер и наименование указывают один раз слева над первой частью таблицы, а над другими частями также слева пишут слова «Продолжение таблицы» и указывают номер таблицы. Таблицу с большим количеством граф допускается делить на части и помещать одну часть под другой в пределах одной страницы. Если строки и графы таблицы выходят за формат страницы, то в первом случае в каждой части таблицы повторяется головка, во втором случае – боковик. При делении таблицы на части допускается ее головку или боковик заменять соответственно номером граф и строк. При этом нумеруют арабскими цифрами графы и (или) строки первой части таблицы. Если повторяющийся в разных строках графы таблицы текст состоит из одного слова, то его после первого написания допускается заменять кавычками; если из двух и более слов, то при первом повторении его заменяют словами «То же», а далее – кавычками. Ставить кавычки вместо повторяющихся цифр, марок, знаков, математических и химических символов не допускается. Если цифровые или иные данные в какой-либо строке таблицы не приводят, то в ней ставят прочерк.

5.6.5 Цифровой материал, как правило, оформляют в виде таблиц. Пример оформления таблицы приведен на рис. Г.1

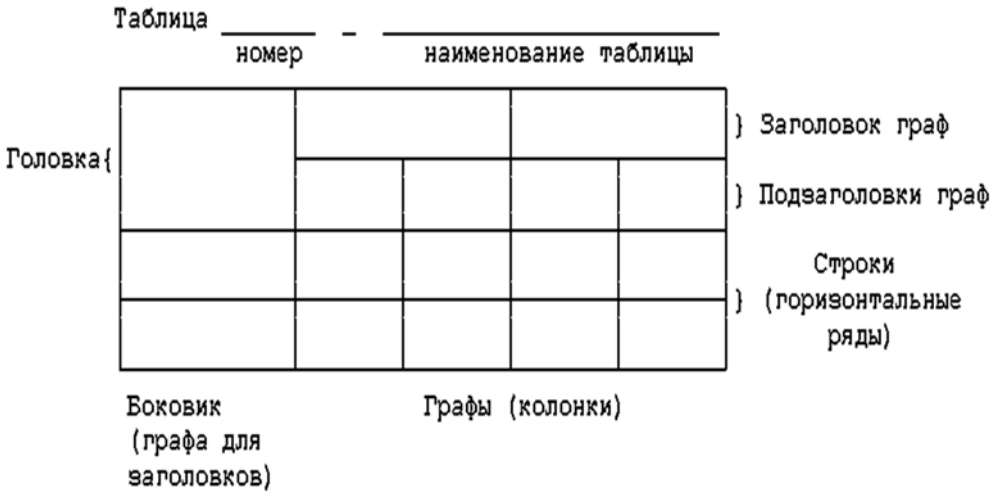


Рис. Г.1 – Пример оформления таблицы

5.6.6 Таблицы, за исключением таблиц приложений, следует нумеровать арабскими цифрами сквозной нумерацией. Допускается нумеровать таблицы в пределах раздела. В этом случае номер таблицы состоит из номера раздела и порядкового номера таблицы, разделенных точкой. Таблицы каждого приложения обозначают отдельной нумерацией арабскими цифрами с добавлением перед цифрой обозначения приложения. Если в отчете одна таблица, то она должна быть обозначена «Таблица 1» или «Таблица В.1», если она приведена в приложении В.

5.6.7 Заголовки граф и строк таблицы следует писать с прописной буквы в единственном числе, а подзаголовки граф – со строчной буквы, если они составляют одно предложение с заголовком, или с прописной буквы, если они имеют самостоятельное значение. В конце заголовков и подзаголовков таблиц точки не ставят.

5.6.8 Таблицы слева, справа и снизу, как правило, ограничивают линиями. Допускается применять размер шрифта в таблице меньший, чем в тексте. Разделять заголовки и подзаголовки боковика и граф диагональными линиями не допускается. Горизонтальные и вертикальные линии, разграничивающие строки таблицы, допускается не проводить, если их отсутствие не затрудняет пользование таблицей. Заголовки граф, как правило, записывают параллельно строкам таблицы. При необходимости допускается перпендикулярное расположение заголовков граф. Головка таблицы должна быть отделена линией от остальной части таблицы.

5.7 Примечания и сноски

5.7.1 Слово «Примечание» следует печатать с прописной буквы с абзаца и не подчеркивать.

5.7.2 Примечания приводят в отчетах, если необходимы пояснения или справочные данные к содержанию текста, таблиц или графического материала.

5.7.3 Примечания следует помещать непосредственно после текстового, графического материала или в таблице, к которым относятся эти примечания. Слово «Примечание» следует печатать с прописной буквы с абзацного отступа и не подчеркивать. Если примечание одно, то после слова «Примечание» ставится тире и примечание печатается с прописной буквы. Одно примечание не нумеруют. Несколько примечаний нумеруют по порядку арабскими цифрами без проставления точки. Примечание к таблице помещают в конце таблицы над линией, обозначающей окончание таблицы.

Пример

Примечание –_____
_____

Несколько примечаний нумеруются по порядку арабскими цифрами.

Пример

Примечания

1 _____

2 _____

3 _____

5.7.4 При необходимости дополнительного пояснения в отчете его допускается оформлять в виде сноски. Знак сноски ставят непосредственно после того слова, числа, символа, предложения, к которому дается пояснение. Знак сноски выполняют надстрочно арабскими цифрами со скобкой. Допускается вместо цифр выполнять сноски звездочками «*». Применять более трех звездочек на странице не допускается. Сноску располагают в конце страницы с абзацного отступа, отделяя от текста короткой горизонтальной линией слева. Сноску к таблице располагают в конце таблицы над линией, обозначающей окончание таблицы.

5.8 Формулы и уравнения

5.8.1 Уравнения и формулы следует выделять из текста в отдельную строку. Выше и ниже каждой формулы или уравнения должно быть оставлено не менее одной свободной строки. Если уравнение не умещается в одну строку, то оно должно быть перенесено после знака равенства (=) или после знаков плюс (+), минус (–), умножения (х), деления (:), или других математических знаков, причем знак в начале следующей строки повторяют. При переносе формулы на знаке, символизирующем операцию умножения, применяют знак «Х».

5.8.2 Пояснение значений символов и числовых коэффициентов следует приводить непосредственно под формулой в той же последовательности, в которой они даны в формуле.

5.8.3 Формулы в отчете следует нумеровать порядковой нумерацией в пределах всего отчета арабскими цифрами в круглых скобках в крайнем правом положении на строке.

Пример:

$$A = a:b, \quad (1)$$

$$B = c:e. \quad (2)$$

Одну формулу обозначают – (1).

5.8.4 Формулы, помещаемые в приложениях, должны нумероваться отдельной нумерацией арабскими цифрами в пределах каждого приложения с добавлением перед каждой цифрой обозначения приложения, например формула (В.1).

5.8.5 Ссылки в тексте на порядковые номера формул дают в скобках. Пример – в формуле (1).

5.8.6 Допускается нумерация формул в пределах раздела. В этом случае номер формулы состоит из номера раздела и порядкового номера формулы, разделенных точкой, например (3.1).

5.8.7 Порядок изложения в отчете математических уравнений такой же, как и формул.

5.8.8 В отчете допускается выполнение формул и уравнений рукописным способом черными чернилами.

5.9 Ссылки

5.9.1 Ссылки на использованные источники следует указывать порядковым номером библиографического описания источника в списке использованных источников. Порядковый номер ссылки заключают в квадратные скобки. Нумерация ссылок ведется арабскими цифрами в порядке приведения ссылок в тексте отчета независимо от деления отчета на разделы.

5.9.2 При ссылках на стандарты и технические условия указывают только их обозначение, при этом допускается не указывать год их утверждения при условии полного описания стандарта и технических условий в списке использованных источников в соответствии с ГОСТ 7.1.

5.10 Титульный лист

5.10.1 Титульный лист содержит реквизиты:

– наименование вышестоящей организации или другого структурного образования, в систему которого входит организация-исполнитель, наименование организации (в том числе и сокращенное);

– индекс УДК, код ВКГ Общероссийского классификатора продукции (ОКП) (для отчетов о НИР, предшествующих разработке и модернизации продукции) и номер государственной регистрации НИР, проставляемые организацией-исполнителем, а также надпись «Инв. N» – эти данные размещаются одно под другим;

– специальные отметки (при наличии в отчете численных данных о свойствах веществ и материалов в этой части проставляется аббревиатура ГСССД – государственная служба стандартных справочных данных);

– гриф согласования, гриф утверждения.

Гриф утверждения состоит из слова «Утверждаю», должности с указанием наименования организации, ученой степени, ученого звания лица, утвердившего отчет, личной подписи, ее расшифровки и даты утверждения отчета. Здесь же проставляется печать организации, утвердившей отчет. Гриф согласования состоит из слова «Согласовано», должности с указанием наименования организации, ученой степени, ученого звания лица, согласовавшего отчет, его личной подписи, ее расшифровки, даты согласования, печати согласующей организации.

Если согласование проводилось письмом, следует указывать сокращенное наименование согласующей организации, исходящий номер и дату

письма. В реквизитах «гриф согласования» и «гриф утверждения» составные части, состоящие из нескольких строк, печатают через 1 межстрочный интервал, а сами составные части отделяют друг от друга 1,5 межстрочным интервалом. В этих реквизитах слова «Утверждаю» и «Согласовано» допускается центрировать в пределах поля реквизита.

Подписи и даты подписания должны быть выполнены только черными чернилами или тушью. Дата оформляется арабскими цифрами в следующей последовательности: день месяца, месяц, год. День месяца и месяц оформляются двумя парами арабских цифр, разделенными точкой; год – четырьмя арабскими цифрами. Например, дату 10 апреля 2000 г. следует оформлять: 10.04.2000.

Допускается словесно-цифровой способ оформления даты, например: 10 апреля 2000 г. Допускается также оформление даты в следующей последовательности: год, месяц, день месяца, например 2000.04.10;

– вид документа приводят прописными буквами, наименование программы (НИР) – строчными буквами с первой прописной, наименование отчета – прописными буквами, вид отчета (промежуточный или заключительный) – строчными буквами в круглых скобках.

Если наименование отчета совпадает с наименованием темы программы (НИР), то его печатают прописными буквами;

– шифр государственной научно-технической программы, шифр работы, присвоенный организацией-исполнителем;

– должности, ученые степени, ученые звания руководителей организации-исполнителя НИР, руководителей НИР (если печатается в несколько строк, то печатать через 1 межстрочный интервал), затем оставляют свободное поле для личных подписей и помещают инициалы и фамилии лиц, подписавших отчет, в одной строке с подписями проставляют даты подписания (если на титульном листе не размещаются все необходимые подписи, то их переносят на дополнительную страницу титульного листа). В правом верхнем углу дополнительной страницы указывают «Продолжение титульного листа», а в конце первой страницы справа указывают «Продолжение на следующем листе»;

– город и год выпуска отчета. (Поправка).

5.10.2 Примеры оформления титульных листов приведены в приложении Б.

5.11 Список исполнителей

5.11.1 Фамилии и инициалы, должности, ученые степени, ученые звания в списке следует располагать столбцом. Слева указывают должности, ученые степени, ученые звания руководителя НИР, ответственных исполнителей, исполнителей и соисполнителей (если печатается в несколько строк, то печатать через 1 межстрочный интервал), затем оставляют свободное

поле для подлинных подписей, справа указывают инициалы и фамилии исполнителей и соисполнителей. Возле каждой фамилии в скобках следует указывать номер раздела (подраздела) и фактическую часть работы, подготовленную конкретным исполнителем. Для соисполнителей следует также указывать наименование организации-соисполнителя.

5.12 Определения, обозначения и сокращения

Перечень должен располагаться столбцом. Слева в алфавитном порядке приводят сокращения, условные обозначения, символы, единицы физических величин и термины, справа – их детальную расшифровку.

5.13 Список использованных источников

Сведения об источниках следует располагать в порядке появления ссылок на источники в тексте отчета и нумеровать арабскими цифрами без точки и печатать с абзацного отступа.

5.14 Приложения

5.14.1 Приложение оформляют как продолжение данного документа на последующих его листах или выпускают в виде самостоятельного документа.

5.14.2 В тексте отчета на все приложения должны быть даны ссылки. Приложения располагают в порядке ссылок на них в тексте отчета.

5.14.3 Каждое приложение следует начинать с новой страницы с указанием наверху посередине страницы слова «Приложение», его обозначения. Приложение должно иметь заголовок, который записывают симметрично относительно текста с прописной буквы отдельной строкой.

5.14.4 Приложения обозначают заглавными буквами русского алфавита, начиная с А, за исключением букв Е, З, Й, О, Ч, Ъ, Ы, Ь. После слова «Приложение» следует буква, обозначающая его последовательность. Допускается обозначение приложений буквами латинского алфавита, за исключением букв I и O. В случае полного использования букв русского и латинского алфавитов допускается обозначать приложения арабскими цифрами. Если в отчете одно приложение, оно обозначается «Приложение А».

5.14.5 Текст каждого приложения, при необходимости, может быть разделен на разделы, подразделы, пункты, подпункты, которые нумеруют в пределах каждого приложения. Перед номером ставится обозначение этого приложения.

5.14.6 Приложения должны иметь общую с остальной частью документа сквозную нумерацию страниц.

5.14.7 Приложение или несколько приложений могут быть оформлены в виде отдельной книги отчета, при этом на титульном листе под номером книги следует писать слово «Приложение». При необходимости такое приложение может иметь раздел «Содержание».

ПРИЛОЖЕНИЕ Д

ГОСТ Р (ИСО/МЭК 12119–2000)

Информационная технология «Пакеты программ».

Требования к качеству и тестирование

Information technology. Software packages.
Quality requirements and testing

1 Область применения

Настоящий стандарт применяется для пакетов программ. Например, для текстовых процессоров, электронных таблиц, программ баз данных, графических пакетов, программ, реализующих технические и научные функции, и для сервисных программ (утилит).

Стандарт устанавливает:

– требования к пакетам программ (требования к их качеству);

– инструкции по испытанию пакета программ на соответствие его установленным требованиям (инструкции по тестированию, в частности по тестированию третьей стороной).

Стандарт предназначен только для пакетов программ, являющихся объектами продажи и поставки. Стандарт не связан с процессом их производства (включая соответствующие работы и промежуточные продукты, например, технические задания). Область применения настоящего стандарта не охватывает систему качества поставщика*.

Примечание – Для некоторых программных средств необходимы дополнительные требования, например, для программных средств, критичных по безопасности.

Пользователями настоящего стандарта являются:

a) поставщики, когда они:

1) определяют требования к пакету программ;

2) проектируют формат для описания продуктов;

3) оценивают собственные продукты;

4) выпускают декларации о соответствии (Руководство ИСО/МЭК 22 [1]);

5) обращаются за сертификатами или знаками соответствия (Руководство ИСО/МЭК 23 [2]);

b) органы по сертификации, которые хотят применять схему сертификации третьей стороной (международные, региональные или национальные) (Руководство ИСО/МЭК 16 [3], Руководство ИСО/МЭК 28 [4] и Руководство ИСО/МЭК 44 [5]);

c) испытательные лаборатории, которые желают соблюдать инструкции по тестированию при проведении тестирования для выдачи сертификата или знака соответствия (Руководство ИСО/МЭК 25 [6]);

d) аккредитующие органы, проводящие аккредитацию органов по сертификации и испытательных лабораторий (Руководство ИСО/МЭК 40 [7] и Руководство ИСО/МЭК 58 [8]);

e) аудиторы испытательных лабораторий, проводящие оценку их компетентности (Руководство ИСО/МЭК 58 [8]);

f) покупатели, которые могут:

1) сравнить свои требования с требованиями, описанными в настоящем стандарте;

2) сравнить требования к конкретной рабочей задаче с информацией, приведенной в описаниях продуктов для существующих продуктов;

3) искать сертифицированные продукты;

4) проверить удовлетворение требований по другим аспектам;

g) пользователи, которые выбирают лучшие продукты[1]

2 Определения

В настоящем стандарте используют следующие термины с соответствующими определениями:

2.1 **функция** (function): Реализация в программе алгоритма, по которому пользователь или программа могут частично или полностью выполнять решаемую задачу.

Примечания

1 Пользователю нет необходимости вызывать функцию (например, автоматическое резервирование или сохранение данных).

2 Определение функции в настоящем стандарте уже, чем в ИСО/МЭК 2382-14 [9] (в части определений отказа, сбоя, эксплуатации и надежности), но шире аналогичных определений в ИСО 2382-2 [10] и ИСО 2382-15 [11].

2.2 **нормативный документ** (requirements document): Документ, содержащий любую комбинацию рекомендаций, требований или правил, которым должен удовлетворять пакет программ.

Примечание – Например, технический или эргономический стандарт, перечень требований (или типовое техническое задание) от какой-либо группировки (например, рыночного сектора, технической ассоциации или ассоциации пользователей), закон или указ.

2.3 **описание** продукта (product description): Документ, определяющий свойства пакета программ, основным назначением которого является оказание помощи потенциальным покупателям в оценке пригодности для них данного продукта до его приобретения.

Примечание – Данный термин является более конкретным, чем термин «описание системы» по ИСО/МЭК 2382-20 [12]. Назначением описания продукта является включение в него «информации на упаковке» по ГОСТ Р

[1] Стандарт дополнен приложением.

ИСО 9127 [13]. Описание продукта не является техническим заданием (или техническими условиями), оно имеет другое назначение.

2.4 **документация пользователя** (user documentation): Полный комплект документов, поставляемых в печатном или другом виде, который обеспечивает применение продукта, а также является его неотъемлемой частью продукта.

2.5 **документация пакета** (package documentation): Описание продукта и документация пользователя.

2.6 **контрольный пример** (test case): Документально оформленное руководство для испытателя, которое определяет, как должна или может быть протестирована функция или комбинация функций. Контрольный пример должен содержать информацию, охватывающую следующие вопросы:

– цель тестирования;

– тестируемые функции;

– среда тестирования и другие условия (уточнение конфигурации средств испытаний и подготовительная работа);

– тестовые данные;

– процедура;

– ожидаемое поведение системы.

2.7 **сопровождение** (maintenance): Составная часть сопровождения системы (см. А.5.2), которая связана с модификацией (изменением) пакета программ.

3 Требования к качеству

В подразделах 3.1–3.3 установлены:

– требования, согласно которым каждый пакет программ должен содержать описание продукта и документацию пользователя;

– требования к описанию продукта. В частности, требование, согласно которому описание продукта должно содержать конкретную информацию, а все приводимые в нем формулировки должны быть проверяемыми (контролируемыми) и корректными;

– требования к документации пользователя;

– требования к любым программам и данным, входящим в состав пакета программ.

Примечания

1 Требования к документации пользователя, программам и данным содержат много общих требований (независимо от требований, которые могут быть заявлены в описании продукта), но они не охватывают всех свойств продукта, которые могут требоваться пользователям.

2 Некоторые свойства, например, «понятность» и «простота изучения» документации пользователя и программных сообщений, необходимы

с точки зрения пользователя. Однако из-за трудности их тестирования с получением точных и воспроизводимых результатов в настоящем стандарте данные требования формулируются только в виде рекомендаций.

3 Требования 3.1–3.3 расположены в том же порядке, что и характеристики в ГОСТ Р ИСО/МЭК 9126 [14].

Пакет программ соответствует настоящему стандарту, если он удовлетворяет всем требованиям, установленным в 3.1–3.3. Рекомендации (указанные с использованием глаголов «может» или «следует») являются необязательными (факультативными).

Примечание 4 – Соответствие продукта требованиям 3.1–3.3 может быть трудно или невозможно подтвердить. Однако испытание (включая проверку документов) согласно разделу 4 считают достаточным для подтверждения достоверности, требуемой для сертификата соответствия согласно Руководству ИСО/МЭК 2 [15].

3.1 Описание продукта

Каждый пакет программ должен содержать описание продукта. Описание продукта должно содержать характеристику продукта. Оно должно являться частью документации пакета для данного продукта. Оно должно содержать информацию по документации пользователя, программам и соответствующим данным.

Основными назначениями описания продукта являются:

– помощь пользователю или потенциальному покупателю при оценке ими пригодности продукта для их нужд. Для обеспечения этого описание продукта также должно содержать соответствующую торговую информацию;

– обеспечение основы для тестирования (см. раздел 4).

Описание продукта должно быть доступным для человека, заинтересованного в данном продукте.

3.1.1 Общие требования к содержанию

Описание продукта должно быть достаточно понятным, полным и простым при изучении, чтобы обеспечить помощь потенциальным покупателям при оценке ими пригодности данного продукта для их нужд до его покупки.

Данное описание должно быть внутренне непротиворечивым. Каждый термин должен иметь один и тот же смысл по всему документу.

Формулировки в описании продукта должны быть проверяемыми и корректными.

Примечание – Данное требование распространяется на формулировки в соответствующих внешних ссылочных нормативных документах (см. 3.1.2, е).

Пункты 3.1.2–3.1.8 определяют, что должно или может быть включено в описание продукта. В него могут быть включены дополнительные формулировки, относящиеся к продукту.

3.1.2 Обозначения и указания

a) *Обозначение описания продукта*

Описанию продукта должно быть присвоено индивидуальное обозначение как документу. Оно может иметь наименование, отличное от «описания продукта», например «Описание функциональных возможностей», «Информация о продукте», «Формуляр продукта».

b) *Обозначение продукта*

Описание продукта должно определять продукт. Обозначение продукта по крайней мере должно включать наименование продукта и обозначение его версии или даты выпуска. Если в описании продукта упомянуты два или несколько вариантов, то для каждого варианта по крайней мере должно быть установлено наименование продукта, наименование варианта и обозначение версии или даты его выпуска.

c) *Поставщик*

В описание продукта должны быть включены наименование и адрес по крайней мере одного поставщика.

Примечание – Наименование и адрес не обязательно включать в текст описания продукта, достаточно поставить на документ штамп (печать) соответствующего дилера.

d) *Рабочая задача*

В описании продукта должны быть определены целевые рабочие задачи, которые могут быть выполнены данным продуктом.

e) *Соответствие нормативным документам*

Из описания продукта могут быть даны ссылки на нормативные документы, которым удовлетворяет данный продукт, в этом случае должны быть указаны соответствующие редакции данных документов.

f) *Необходимая система*

Должна быть определена система (технические и программные средства и их конфигурация), необходимая для ввода продукта в эксплуатацию, включая наименования изготовителей и обозначения типов всех ее частей, например:

– процессоры, включая сопроцессоры;
– объем основной (оперативной) памяти;
– типы и объемы (памяти) периферийных запоминающих устройств;
– расширяющие платы;
– оборудование ввода и вывода;
– сетевое оборудование;
– системные и прочие программные средства.

Могут быть определены различные необходимые системы, например, для различных рабочих задач, различных граничных значений или различных требований к эффективности.

Формулировка «(или любое другое ..., если оно совместимо)» может быть приведена в описании продукта, если предварительно было определено конкретное техническое или программное средство. Формулировка «или модернизированная версия, если она совместима» может быть приведена, если предварительно в описании продукта была определена соответствующая версия продукта. Может быть приведена формулировка «от версии *X* до версии *Y*», а формулировка «от версии *X*» приводиться не должна.

Примечание – Формулировка «от версии *X*» позже может стать ошибочной, например, при появлении версии *X*+3, под которой пакет программ будет работать со сбоями.

g) *Интерфейсы с другими продуктами*

Должны быть определены соответствующие интерфейсы или продукты, если в описании продукта имеются ссылки на интерфейсы с другими продуктами.

h) *Объекты поставки*

Должен быть определен каждый физический компонент поставляемого продукта, в частности все печатные документы и все носители данных.

Должен быть установлен вид поставляемых программ, например, исходные программы, объектные (рабочие) модули или загрузочные модули.

Примечание – Указывать форматы носителей (например, форматы дискет) нет необходимости, потому что набор возможных форматов определяется необходимой системой (см. 3.1.2 f).

i) *Ввод в действие (инсталляция)*

Должно быть указано, будет ли инсталляция продукта проводиться пользователем или нет. *Поддержка*

Должно быть указано, будет или не будет предлагаться поддержка при эксплуатации продукта, к) *Сопровождение*

Должно быть указано, будет или не будет предлагаться сопровождение продукта. Если сопровождение предусматривается, то должно быть установлено, что оно подразумевает. 3.1.3 Формулировки функциональных возможностей

a) *Обзор функций*

В описании продукта должен быть приведен обзор функций продукта, вызываемых пользователем, необходимых для них данных и предоставляемых средств.

Для каждой функции (особенно для ее опции или варианта) должно быть четко установлено, является ли она частью:

– продукта;
– расширения продукта, полностью описанного в описании продукта;
– расширения продукта, на которое дана ссылка в описании продукта;
– не гарантируемого (не обязательного) приложения.

Примечание – Нет необходимости упоминать каждую функцию, вызываемую пользователем, и приводить все подробности вызова описываемой функции.

b) *Граничные значения*

Если использование продукта ограничено конкретными граничными значениями для продукта, они должны быть указаны в описании продукта.

Например:

– минимальные или максимальные значения;
– длины ключей;
– максимальное число записей в файле;
– максимальное число критериев поиска;
– минимальный объем выборки.

В случае, когда не возможно определить фиксированные граничные значения (например, когда они зависят от типа приложения или от исходных данных), тогда на них должны быть наложены соответствующие ограничения. Могут быть приведены допустимые комбинации значений и даны ссылки на более конкретную информацию из документации пользователя.

c) *Защита*

При необходимости в описание продукта должна быть включена информация по средствам предотвращения случайного или преднамеренного несанкционированного доступа к программам и данным.

3.1.4 Формулировки надежности

В описание продукта должна быть включена информация по процедурам сохранения данных.

Примечание – Данную информацию можно привести, указав, например, возможности резервирования данных с помощью функций операционной системы.

Могут быть описаны дополнительные характеристики продукта, которые обеспечивают его функциональные возможности. Например:

– проверки достоверности исходных данных;
– защита против серьезных последствий ошибки пользователя;
– восстановление при ошибках.

3.1.5 Формулировки практичности

a) *Интерфейс пользователя*

Должен быть назван тип интерфейса пользователя, например, строка команд, меню, окна, функциональная клавиша, функция подсказки.

b) *Требуемые знания*

Должны быть определены конкретные знания, которые необходимо усвоить пользователю для применения соответствующего продукта. Например:

– знание соответствующей технической области;
– знание операционной системы;
– знания, получаемые в результате специального обучения;
– знание языков, отличных от языка, на котором написано описание продукта.

Должны быть указаны все естественные языки, на которых написана документация пользователя и описан интерфейс пользователя (включая сообщения об ошибках и видимую информацию), как для самого пакета программ, так и для всех других продуктов, упомянутых в описании данного продукта.

Примечание – Данное требование строже, чем аналогичное требование 6.1.7 ГОСТ Р ИСО 9127 [13], согласно которому указание языка является необязательным.

c) *Адаптация к потребностям пользователя*

Если продукт может настраиваться (адаптироваться) пользователем, то должны быть указаны инструментальные средства для проведения такой настройки и условия их применения. Например:

– изменение параметров;
– изменение алгоритмов вычислений;
– назначение функциональных клавиш.

d) *Защита от нарушения авторских прав*

Если техническая защита от нарушения авторских прав может ухудшить практичность описываемого продукта, то в описании продукта должны быть указаны виды и средства такой защиты. Например:

– техническая защита от копирования;
– запрограммированные даты окончания использования продукта;
– интерактивные напоминания об оплате за копии.

e) *Эффективность применения и удовлетворение потребностей пользователя*

В описание продукта может быть внесена информация по эффективности применения продукта и удовлетворению им потребностей пользователя.

Примечание – Такая информация может быть описана в соответствии с рекомендациями, приведенными в ИСО 9241-11 [16].

3.1.6 Формулировки эффективности

В описание продукта может быть включена информация о характере поведения продукта во времени, например, времена ответа и оценки производительности для заданных функций при установленных условиях (например, для заданных конфигураций системы и профилей загрузки).

3.1.7 Формулировки сопровождаемости

В описание продукта могут быть внесены формулировки требований (правил) по сопровождению продукта.

3.1.8 Формулировки мобильности (переносимости)

В описание продукта могут быть внесены формулировки требований (правил) по мобильности продукта.

3.2 Документация пользователя

3.2.1 Полнота (completeness)

Документация пользователя должна содержать информацию, необходимую для использования продукта. В документации пользователя должны быть полностью описаны все функции, установленные в описании продукта, и все вызываемые пользователем функции из программы.

Все граничные значения, заданные в описании продукта, должны быть продублированы в документации пользователя.

Если установка (инсталляция) продукта может быть проведена пользователем, то в документацию пользователя должно быть включено руководство по установке продукта, содержащее всю необходимую информацию (см. 3.3.1 a). В руководстве могут быть указаны минимальные и максимальные объемы однократно устанавливаемых файлов.

Если сопровождение продукта может проводиться пользователем, то в документацию пользователя должно быть включено руководство по сопровождению программы, содержащее всю информацию, которая необходима для обеспечения данного вида сопровождения.

3.2.2 Правильность (correctness)

Вся информация в документации пользователя должна быть правильной. Кроме того, представление данной информации не должно содержать неоднозначных толкований и ошибок.

3.2.3 Непротиворечивость (consistency)

Документы, входящие в комплект документации пользователя, не должны противоречить сами себе, друг другу и описанию продукта. Каждый термин должен иметь один и тот же смысл во всех документах.

Примечание – Непротиворечивость программ и данных рассматривается в 3.3.1, d.

3.2.4 Понятность (understandability)

Документация пользователя должна быть понятной для сообщества пользователей, выполняющих указанную рабочую задачу, например посредством использования в ней соответствующим образом подобранных терминов, графических вставок, уточняющих пояснений и путем ссылок на полезные источники информации.

3.2.5 Простота обозрения (ease of overview)

Документация пользователя должна быть достаточно проста для изучения пользователем, чтобы он мог выявить все описываемые в ней взаимосвязи компонентов продукта.

В каждый документ может быть включено оглавление и предметный указатель.

Если документ не поставляется в напечатанном виде, то должна быть определена процедура его распечатки с соответствующего носителя данных.

3.3 Программы и данные

3.3.1 Функциональные возможности

a) *Установка (инсталляция)*

Если установка пакета может быть выполнена пользователем, то при ее проведении должна быть обеспечена возможность успешной установки программ в соответствии с информацией, содержащейся в руководстве по установке. Каждая из необходимых систем, указанных в описании продукта, должна быть пригодной для установки программ.

В процессе установки должно быть определено, могут ли установленные программы функционировать, например, путем использования поставленных с программами контрольных примеров или самотестирования с выдачей соответствующих сообщений.

b) *Реализация функций*

Все функции, указанные в документации пользователя, должны выполняться в виде, заданном Б документации пользователя, на соответствующих средствах, с соответствующими характеристиками и данными, в рамках граничных значений, заданных там же.

Примечание – Так как все функции, указанные в описании продукта, должны быть отражены в документации пользователя, из этого следует, что данные функции так же должны выполняться.

c) *Правильность*

Программы и данные должны соответствовать всем обязательным формулировкам, приведенным в описании продукта и документации пользователя. Функции должны выполняться методом, соответствующим рабочей задаче. В частности, программы и данные должны удовлетворять всем требованиям из любого нормативного документа, на который дана ссылка в описании продукта.

d) *Непротиворечивость*

Программы и данные не должны противоречить сами себе, а также описанию продукта и документации пользователя. Каждый термин везде должен иметь один и тот же смысл.

Управление работой программы со стороны пользователя и соответствующая реакция программы (например, сообщения, выходные экранные форматы и печатные отчеты) должны быть единообразно структурированы.

3.3.2 Надежность

Система, включая технические средства, необходимые программные средства и те программы, которые входят в продукт, не должны приходить в такое состояние, чтобы пользователь не мог их контролировать, а данные не должны ни повреждаться, ни теряться.

Это требование должно одинаково удовлетворяться в случаях, когда:

– возможность реализуется при конкретных ограничениях;

– имеют место попытки реализации возможности вне заданных ограничений;

– неправильные исходные данные вводятся пользователем или от других программ, перечисленных в описании продукта;

– нарушаются инструкции, заданные в документации пользователя.

Исключаются только те возможности прерывания технических средств и операционной системы, которые не могут быть распознаны любой программой (например, клавиша или комбинация клавиш для сброса системы).

Программы должны обнаруживать нарушения синтаксических правил для исходных данных. В случае, когда программа определяет исходные данные как ошибочные или неопределенные, она не должна их обрабатывать как допустимые исходные данные.

3.3.3 Практичность

В отношении практичности продукта сторонам, заключающим соглашение на основе настоящего стандарта, рекомендуется изучить возможности применения последних редакций стандартов серии ИСО 9241 [16]–[29].

Примечание – В частности, следует принимать во внимание ИСО 9241-10 [24] и ИСО 9241-13 [26].

a) *Понятность*

Запросы, сообщения и результаты выполнения программ должны быть понятными, например:

– путем соответствующего выбора терминов;
– путем графических представлений;
– путем представления исходной информации;
– путем пояснений из функции подсказки.

В сообщениях об ошибках должна содержаться подробная информация, разъясняющая причину или способ исправления соответствующих ошибок из-за неправильного применения продукта (например, путем ссылки на элемент документации пользователя).

b) *Простота обозрения*

На каждый носитель данных должно быть нанесено обозначение продукта, а если носителей несколько – различающий номер или текст.

Пользователю, работающему с программами, всегда должна быть предоставлена возможность выяснения, какая из функций выполняется.

Программы должны предоставлять пользователю информацию в таком виде, чтобы данная информация им легко воспринималась и читалась. Пользователь может руководствоваться соответствующими кодификаторами и классификаторами информации. При необходимости программы должны выдавать пользователю соответствующие уведомления.

Сообщения от программ следует проектировать таким образом, чтобы пользователь мог легко различать их типы, например:

– подтверждение приема;
– запросы от программ;
– предупреждения;
– сообщения об ошибках.

Форматы входных экранов, отчеты, а также другие исходные и выходные данные следует проектировать так, чтобы их можно было легко просматривать. Для этого могут быть использованы следующие возможности:

– алфавитно-цифровые поля и левостороннее выравнивание;

– числовые поля и правостороннее выравнивание;

– расположение в таблицах десятичных точек или запятых на одной вертикальной линии;

– распознаваемые ограничители полей;

– соответствующее распознавание полей, использование которых обязательно;

– указание ошибок ввода непосредственной засветкой на входном экране;

– привлечение внимания пользователя к изменению содержания экрана путем подачи визуального или звукового сигнала.

c) *Простота использования*

Исполнение функций, приводящих к серьезным последствиям при эксплуатации системы, должно быть обратимым, или программы должны выдавать четкие предупреждения о последствиях выполнения данных функций и запрашивать разрешающее подтверждение перед выполнением соответствующей команды. В частности, к серьезным последствиям могут привести стирание или перезапись данных, а также прерывания режима продолжительной обработки.

Если текст документа предоставляется в диалоговом режиме, то пользователю следует обеспечить возможность непосредственного доступа к отдельным структурным элементам текста (разделам, пунктам, абзацам и т. д.), например, путем выбора данных элементов из отображенного на экране содержания документа или с помощью функции поиска по ключевым словам.

3.3.4 Эффективность

Ничего не требуется. Однако в описании продукта должны присутствовать соответствующие формулировки эффективности.

3.3.5 Сопровождаемость

Ничего не требуется. Однако в описании продукта должны присутствовать соответствующие формулировки сопровождаемое™.

3.3.6 Мобильность (переносимость)

Ничего не требуется. Однако в описании продукта должны присутствовать соответствующие формулировки мобильности.

4 Указания по тестированию

Указания, приведенные в 4.1–4.5, определяют порядок тестирования продукта на соответствие его требованиям к качеству. Данные указания охватывают как тестирование для характеристик, присущих всем аналогичным продуктам, так и тестирование для характеристик, продекларированных

в описании продукта. Указания также охватывают как тестирование путем проверки документов, так и тестирование программ и данных по принципу «черного ящика».

Данные указания описывают функциональное тестирование (тестирование по принципу «черного ящика»). Структурное тестирование данными указаниями не охватывается, потому что для его проведения необходимо наличие исходного кода.

Рассматривают только тестирование продукта в необходимых для него системах. Эргономическую оценку на рабочем пространстве вычислительной системы в настоящем стандарте не рассматривают.

Примечания:

1 Данные указания предназначены в первую очередь для тестирования, проводимого третьей стороной по некоторой схеме сертификации (см. 1 c). При производстве дешевле и эффективнее использовать структурное тестирование.

2 В разделе 4 отсутствуют требования к пакетам программ (все данные требования приведены в разделе 3). Пакет программ может быть аттестован без проведения тестирования согласно разделу 4, а подобная процедура может оказаться неэффективной для выявления существующих в продукте несоответствий.

3 Так как необходимая система определяется в описании продукта, то любое несоответствие продукта в необходимой системе трактуется как несоответствие продукта.

4 Схема сертификации может предусматривать тестирование на соответствие необязательным рекомендациям.

5 Руководство по эргономической оценке приведено в ИСО 9241-11 [16].

4.1 Необходимые условия для тестирования

4.1.1 Наличие компонентов продукта

Для тестирования пакета программ должны иметься в наличии все его поставляемые компоненты (см. 3.1.2 h), а также нормативные документы, указанные в описании продукта (см. 3.1.2 e).

4.1.2 Наличие компонентов системы

Для тестирования пакета программ необходимо наличие составных частей всех вычислительных систем, указанных в описании продукта.

4.1.3 Обучение

Если в описании продукта указана необходимость соответствующего обучения, то испытатель (тестировщик) должен иметь доступ к учебным материалам и обучающим программам.

4.2 Работы по тестированию

Описание продукта, документация пользователя, программы и любые данные, поставляемые как части пакета программ:

– должны быть протестированы на соответствие требованиям раздела 3;

– могут быть протестированы на соответствие рекомендациям раздела 3.

Цели тестирования должны быть определены исходя из требований раздела 3 и должны охватывать все эти требования (полноту, непротиворечивость и т. д.).

Если в описании продукта упомянуты другие продукты, то их необходимо протестировать только по требованиям, предъявленным к ним в описании тестируемого продукта.

Подробные формулировки из описания продукта, документации пользователя, функций и данных для продукта тестировать нет необходимости, если по оценке тестировщика:

– они оказывают незначительное влияние на соответствие названной рабочей задаче;

– они могут быть протестированы в принципе, но с неоправданными затратами ресурсов.

Те подробные формулировки, которые не были протестированы, должны быть указаны в протоколах тестирования и в отчете о тестировании. Причины, по которым они не были протестированы, должны документироваться в протоколах тестирования.

4.2.1 Описание продукта

Должно быть протестировано выполнение соответствующих требований раздела 3, а выполнение рекомендаций раздела 3 может быть протестировано.

4.2.2 Документация пользователя

Должно быть протестировано выполнение соответствующих требований раздела 3, а выполнение рекомендаций раздела 3 может быть протестировано.

4.2.3 Программы и данные

Должно быть протестировано выполнение соответствующих требований раздела 3, а выполнение рекомендаций раздела 3 может быть протестировано.

Программы должны быть протестированы во всех вычислительных системах, указанных в описании продукта.

При наличии нескольких вариантов программы должен быть протестирован каждый из них. Каждая из функций, которые в соответствии с описанием продукта и документацией пользователя одинаковы в ряде вариантов, может быть протестирована в одном из вариантов.

Поставляемые программы и данные должны быть протестированы с использованием контрольных примеров, разработанных на основе описания продукта и документации пользователя. Другие материалы (например, исходные программы) не проверяют, за исключением случаев, когда это необходимо при тестировании формулировок из описания продукта или документации пользователя.

Контрольные примеры должны быть методологически и систематически проработаны.

Примечание – Допускается использование методологии тестирования с применением случайных кодов.

Если в документации пользователя приведены примеры, то они должны быть использованы в качестве контрольных, но проводимое тестирование не должно быть ограничено только этими примерами.

Могут быть использованы контрольные примеры, предоставляемые поставщиком программного пакета, но проводимое тестирование не должно быть ограничено только этими примерами.

а) *Установка (инсталляция)*

Если в соответствии с описанием продукта установка пакета может быть выполнена пользователем, должна быть проверена возможность инсталляции программ и протестирована возможность успешной установки пакета согласно описанию, приведенному в руководстве по установке.

Любым способом должно быть обеспечено, чтобы техническая и программная среда, в которой установлены программы, соответствовала формулировкам из описания продукта в части рассматриваемой вычислительной системы.

Ь) *Выполнение программы*

Контрольные примеры должны охватывать все функции, приведенные в описании продукта и документации пользователя, а также должны учитывать комбинации функций, характерные для рабочей задачи.

Программы должны быть протестированы по всем граничным значениям (в соответствии с описанием продукта и документацией пользователя) в необходимой системе, для которой заданы эти значения.

При тестировании должны быть использованы исходные данные и последовательности команд, которые в документации пользователя явно не рекомендуются или объявляются запрещенными (см. 3.3.2).

4.3 Протоколы тестирования

Протоколы по каждому тесту должны содержать информацию, достаточную для повторения теста (Руководство ИСО/МЭК 25 [6]). Данная информация должна включать:

– план тестирования или технические требования (спецификацию) к тестированию, содержащие контрольные примеры (для каждого контрольного примера указаны его цели, см. 2.6);

– все результаты, связанные с контрольными примерами, включая все ошибки, выявленные при выполнении теста;

– штат персонала, вовлеченного в тестирование.

4.4 Отчет о тестировании

В отчете о тестировании должны быть суммированы цели и результаты тестирования (описанные в протоколах тестирования для каждого теста). Отчет о тестировании должен иметь следующую структуру.

1 Обозначение продукта.

2 Вычислительные системы, использованные при тестировании (технические средства, программные средства и их конфигурация).

3 Использованные документы (включая их обозначения).

4 Результаты тестирования описания продукта, документации пользователя, программ и данных.

5 Перечень несоответствий требованиям.

6 Либо перечень несоответствий рекомендациям, либо перечень не учтенных в продукте

рекомендаций, либо формулировка того, что продукт не был протестирован на соответствие рекомендациям.

7 Дата окончания тестирования.

8 раздел 4 отчета о тестировании (Результаты тестирования) должны быть включены формулировки, соответствующие наименованию каждого пункта 3.1–4.2.

В разделе 6 отчета о тестировании дополнительно к формулировке, что продукт не был протестирован на соответствие рекомендациям, может быть приведен перечень замеченных несоответствий рекомендациям.

Обозначение отчета о тестировании (испытательная лаборатория, обозначение продукта, дата выпуска отчета о тестировании) и общее число его страниц должны проставляться на каждой странице отчета о тестировании. Отчет о тестировании должен включать:

– формулировку, что результаты тестирования относятся только к протестированным компонентам продукта;

– формулировку, что полная копия отчета о тестировании не может быть изготовлена без письменного разрешения соответствующей испытательной лаборатории (Руководство ИСО/МЭК 25 [6]).

Отчет о тестировании должен соответствовать положениям Руководства ИСО/МЭК 25 [6], относящимся к отчетам о тестировании.

4.5 Дополнительное тестирование

Когда продукт, который уже был протестирован, тестируется повторно (с учетом результатов предыдущего тестирования), тогда:

– все измененные части документов, функций и данных должны быть протестированы как новый продукт;

– все неизмененные части, на которые могут влиять измененные части или изменения в необходимой системе (в соответствии с опытной оценкой тестировщика), должны быть протестированы как новый продукт;

– все другие части должны быть по крайней мере выборочно протестированы.

ПРИЛОЖЕНИЕ Е

(справочное)

Определения из других нормативных документов

В настоящем приложении приведены некоторые термины и определения, используемые в настоящем стандарте, но описанные в других нормативных документах.

А.1 Общие термины

А.1.1 **программное средство** (software): Все или часть программ, процедур, правил и любой соответствующей документации системы обработки информации. (ИСО/МЭК 2382-1 [30], без примечания).

А.1.2 **пакет программ** (software package): Полный и документально оформленный набор программ, поставляемый отдельным пользователям для реализации общих приложений или функций (ИСО/МЭК 2382-20 [12], без примечания).

А.1.3 **системное программное средство** (system software): Независимое от приложения программное средство, которое обеспечивает выполнение прикладного программного средства (ИСО/МЭК 2382-20 [12]).

А.1.4 **сервисная программа** (utility routine, utility program): Программа (машинная программа), которая обеспечивает общие, часто используемые услуги для пользователей компьютера и обслуживающего персонала (ИСО/МЭК 2382-7 [31], без примеров).

А.1.5 **функциональный модуль** (functional unit): Объект технических или (и) программных средств, имеющий конкретное назначение (ИСО/МЭК 2382-1 [30]).

А.1.6 **(машинная) программа** [(computer) program]: Синтаксический блок, соответствующий правилам конкретного языка программирования и формирующий описания и операторы или команды, необходимые для решения (выполнения) некоторой функции, задачи или проблемы (ИСО/МЭК 2382-1 [30]).

А.1.7 **интерфейс** (interface): Общая граница между двумя функциональными блоками, определяемая различными характеристиками, относящимися к функциям, физическим соединениям, обменам сигналами, и другими соответствующими характеристиками блоков (ИСО/МЭК 2382-9 [32], без примечания).

А.1.8 **интерфейс пользователя** (user interface): Интерфейс, обеспечивающий возможность обмена информацией между пользователем (человеком) и компонентами технических или программных средств вычислительной системы (АНСИ/ИИИЭ Стд 610.12 [33]).

А.1.9 **конфигурация** (configuration): Способ организации и взаимосвязи технических и программных средств системы обработки информации (ИСО/МЭК 2382-1 [30]).

А.2 Характеристики продукта

А.2.1 **функциональные возможности** (functionality): Набор атрибутов, относящихся к сути набора функций и их конкретным свойствам. Функциями являются те, которые реализуют установленные или предполагаемые потребности (ГОСТ Р ИСО/МЭК 9126 [14], без примечаний).

А.2.2 **надежность** (reliability): Набор атрибутов, относящихся к способности программного средства сохранять свой уровень качества функционирования при установленных условиях за установленный период времени (ГОСТ Р ИСО/МЭК 9126 [14], без примечаний).

А.2.3 **практичность** (usability): Набор атрибутов, относящихся к объему работ, требуемых для использования и индивидуальной оценки такого использования определенным или предполагаемым кругом пользователей (ГОСТ Р ИСО/МЭК 9126 [14], без примечаний).

А.2.4 **эффективность** (efficiency): Набор атрибутов, относящихся к соотношению между уровнем качества функционирования программного средства и объемом используемых ресурсов при установленных условиях (ГОСТ Р ИСО/МЭК 9126 [14], без примечания).

А.2.5 **сопровождаемость** (maintainability): Набор атрибутов, относящихся к объему работ, требуемых для проведения конкретных изменений (модификаций) (ГОСТ Р ИСО/МЭК 9126 [14], без примечания).

А.2.6 **мобильность** (portability): Набор атрибутов, относящихся к способности программного средства быть перенесенным из одного окружения (среды) в другое (ГОСТ Р ИСО/МЭК 9126 [14], без примечания).

А.3 Данные

А.3.1 **данные** (data): Представление информации в формализованном виде, пригодном для передачи, интерпретации или обработки (ИСО/МЭК 2382-1 [30], без примечаний).

А.3.2 **носитель данных** (data medium): Материал, в котором или на котором данные могут быть записаны и с которого они могут быть восстановлены (ИСО/МЭК 2382-1 [30]).

А.4 Тестирование

А.4.1 **тестирование** (test): Техническая операция, которая заключается в определении одной или нескольких характеристик данного продукта, процесса или услуги по соответствующей процедуре (Руководство ИСО/МЭК 2 [15]).

А.4.2 **тестовые (контрольные) данные (test data):** Данные, используемые для контрольной задачи (ИСО/МЭК 2382-8 [34]).

А.4.3 **контрольная задача** (check problem): Задача с известным решением, используемая для того, чтобы определить, правильно ли функционирует функциональный модуль (ИСО/МЭК 2382-8 [34]).

А.4.4 **метод тестирования** (test method): Конкретная техническая процедура для проведения тестирования (Руководство ИСО/МЭК 2 [15]).

А.4.5 **план тестирования, план тестирования и оценки системы** (test plan, system test and evaluation plan): План, в котором устанавливаются уточненные требования, критерии, общая методология, обязанности и проводится общее планирование тестирования и оценки системы (ИСО/МЭК 2382-20 [12]).

А.4.6 **отчет о тестировании** (test report): Документ, в котором представляются результаты тестирования и другая информация, относящаяся к тестированию (Руководство ИСО/МЭК 2 [15]).

А.5 Прочие термины

А.5.1 **руководство по сопровождению программы** (program maintenance manual): Документ, в котором приводится вся информация, необходимая для сопровождения программы (ИСО/МЭК 2382-20 [12]).

А.5.2 **сопровождение системы** (system maintenance): Изменение (модификация) системы в целях корректировки ошибок, повышения производительности или адаптации системы к изменившимся условиям среды эксплуатации или к изменившимся требованиям (ИСО/МЭК 2382-20 [12]).

А.5.3 **рабочая задача** (work task): Предусмотренный результат функционирования рабочей системы (ИСО 6385 [35]).

А.5.4 **рабочая система** (work system): Рабочая система охватывает сочетание людей и оборудования, участвующих совместно в процессе выполнения рабочей задачи на рабочем месте в рабочей среде при условиях, налагаемых рабочей задачей (ИСО 6385 [35]).