|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | |  | | МИНОБРНАУКИ РОССИИ | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА**  **Филиал РТУ МИРЭА в г. Ставрополе** | | |

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

**к выполнению курсовой работы**

**по дисциплине «Проектирование информационных систем»**

для студентов направления подготовки: 09.03.01 Информатика и вычислительная техника Направленность (профиль): Программное обеспечение вычислительной техники и автоматизированных систем

Квалификация: бакалавр

Ставрополь

Методические указания составлены в соответствии с требованиями государственного стандарта высшего образования и программой дисциплины «Проектирование информационных систем» для студентов направления подготовки 09.03.01 Информатика и вычислительная техника.

Составитель: Киселева Т.В. , к.ф.-м.н., доцент

Содержание

[Введение 4](#_Toc2772645)

[1. Цели и задачи курсового проекта 4](#_Toc2772646)

[2. Формулировка задания и его объем 5](#_Toc2772647)

[3. Основное содержание курсового проекта по разделам, последовательность и порядок их выполнения 6](#_Toc2772648)

[4. Общие требования к курсовому проекту, требования к оформлению 72](#_Toc2772649)

[5. Рекомендации по организации работы над проектом, примерный календарный план его выполнения 74](#_Toc2772650)

[6. Порядок защиты и ответственность студента за выполнение задания по курсовому проекту 76](#_Toc2772651)

[7. Список рекомендуемой литературы 77](#_Toc2772652)

[Приложение 1 78](#_Toc2772653)

[Приложение 2 79](#_Toc2772654)

[Приложение 3 80](#_Toc2772655)

[Приложение 4 81](#_Toc2772656)

## Введение

Курсовой проект – самостоятельная работа студента, основной целью и содержанием которого является развитие навыков теоретических и экспериментальных исследований, инженерных расчетов, составления технико-экономического обоснования различных решений или обобщений, оценка результатов исследований, способствующих успешной подготовке к выполнению выпускной квалификационной работы.

Курсовой проект занимает важное место в учебном процессе, поскольку он способствует получению навыков и знаний, необходимых студенту при изучении других дисциплин. Курсовой проект является одним из этапов изучения дисциплины «Проектирование информационных систем», в котором проверяется уровень подготовки студентов по профилю изучаемой дисциплины, умение самостоятельно работать с литературой, принимать обоснованные решения при проектировании автоматизированных информационных систем (АИС).

Данный курсовой проект позволяет научиться студенту решать актуальные на сегодняшний день задачи связанные с анализом и проектированием разработки автоматизированных информационных систем.

## 1. Цели и задачи курсового проекта

Основной целью освоения дисциплины «Проектирование информационных систем» является формирование представлений о технологиях проектирования информационных систем, набора профессиональных компетенций будущего бакалавра по направлению подготовки 09.03.01 Информатика и вычислительная техника.

Задачи дисциплины:

* изучение основных этапов жизненного цикла информационной системы;
* знакомство с различными технологиями проектирования ИС;
* знакомство с принципами управления программными проектами;
* изучение принципов структурного анализа и проектирования;
* изучение технологий моделирования IDEF0, DFD;
* изучение технологии объектно-ориентированного проектирования и моделирования (RUP, UML);
* формирование навыков использования структурного и объектно-ориентированного подходов при моделировании ИС.

## 2. Формулировка задания и его объем

Курсовой проект выполняется студентами дневной и заочной форм обучения в установленные учебным планом сроки. Курсовой проект выполняется в соответствии с заданием, определяющим:

- тему курсового проекта;

- исходные данные к проекту;

- содержание пояснительной записки (перечень подлежащих разработке вопросов).

Тема курсового проекта выбирается в соответствии с тематикой (примерная тематика курсовых проектов приведена в Приложении 4). Тема и задание на курсовой проект должны быть согласованы с руководителем проекта и утверждены заведующим кафедрой.

Для разработки темы курсового проекта необходимо подобрать учебную, нормативную, специальную научную литературу и познакомиться с ней. Отбирать следует литературу и источники информации в Internet, которые освещают как общетеоретическую сторону проблемы, так и действующую практику разработки приложений.

Рекомендуемые инструментальные средства: Rational Rose, Oracle Designer, AllFusion Process Modeler (BPWin) и AllFusion ERwin Data Modeler (ERWin), ARIS, Power Designer, Microsoft Visio, Microsoft Access, Microsoft Visual Studio.

## 3. Основное содержание курсового проекта по разделам, последовательность и порядок их выполнения

Пояснительная записка должна содержать следующие элементы, расположенные в указанном порядке:

* титульный лист;
* задание на выполнение курсовой работы;
* аннотацию;
* содержание;
* введение;
* основную часть (по разделам);
* заключение;
* список использованных источников;
* приложения (при необходимости).

Титульный лист и задание на курсовую работу оформляется по установленному образцу (Приложение 1 и Приложение 2).

**Аннотация** представляет собой данные о теме, основной цели курсовой работы, ее объеме, количестве рисунков, таблиц, приложений, использованных источников.

По своей структуре курсовой проект состоит из введения, аналитической и практической частей, заключения, списка использованных источников и приложений.

**Примерный план курсового проекта:**

Введение

1. Аналитическая часть

1.1. Характеристика и анализ объекта исследования.

1.2. Разработка функциональной модели предметной области

1.3. Стоимостной анализ

1.4. Описание и сравнение программ-аналогов

1.5. Разработка технического задания.

2. Практическая часть

2.1. Проектирование диаграммы вариантов использования (диаграммы прецедентов)

2.2. Разработка диаграммы последовательностей

2.3. Проектирование диаграммы взаимодействия для объектов предметной области

2.4. Разработка логической модели предметной области с помощью диаграмм классов

2.5. Проектирование диаграммы состояний для объектов предметной области

2.6. Проектирование диаграмм компонентов для объектов, входящих в предметную область

2.7. Разработка диаграмм размещения для объектов информационной системы

2.8. Создание прототипа информационной системы

2.8.1. Требования к информационной системе

2.8.2. Технология создания прототипа информационной системы

2.8.3. Инструкция пользователю

Заключение

Список использованных источников

Во **введении** обосновывается актуальность выбранной темы для эффективного управления в исследуемой предметной области. Характеризуется объект (область бизнеса) и предмет (конкретный административный бизнес-процесс) курсового проектирования в соответствие с выбранной темой. Определяются цели и формулируются задачи, которые ставятся и решаются студентом в ходе курсового проектирования. Приводится краткое содержание курсового проекта – что рассматривается в первой главе, во второй... и т.п. Во введении следует также указать методы и инструментальные (программные) средства, которые автор использовал в процессе выполнения КП.

В конце введения следует отметить элементы научной новизны и практической значимости полученных результатов для конкретной предметной области.

Объем**Введения**должен быть не более 2-х страниц.

Структура основной частикурсового проекта.

**1. Аналитическая часть**

**1.1. Характеристика и анализ объекта исследования**

В первом подразделе этой части курсового проекта дается краткая характеристика предметной области, как объекта, для которого разрабатывается информационная подсистема, то есть проводится анализ предметной области с целью выделения объектов и определения и связей между ними.

**Общая структура этого подраздела имеет следующий вид**:

1. Область деятельности, партнёры и перечень основных бизнес процессов.
2. Адреса и телефоны.
3. Контактные лица.
4. Сотрудники.
5. Организационная диаграмма.
6. Физическая диаграмма.
7. Подробное описание основных бизнес-процессов.

**Пример характеристики и анализа объекта исследования**

Гипотетическая книготорговая фирма «Либерия» занимается продажей книг и дисков в магазинах, так же осуществляется продажа через Интернет и возможна доставка заказанной литературы из одного магазина в другой.

Основные бизнес-процессы фирмы: доставка закупленной продукции в магазин, её последующее оформление, предоставление на продажу.

Адреса и телефоны:

Адрес главного магазина: г. Москва, ул. Ивана Грозного, д. 11;

тел./факс (844) 245-11-22 (многоканальный); график работы: с 10.00 до 20.45 без перерыва и выходных

г. Тула, ул. Ленина, д.66;

тел./факс (654) 968-77-88 (многоканальный); график работы: с 10.00 до 20.00 без перерыва и выходных

г. Самара, ул. Кирова, д. 56;

тел./факс (846) 586-99-00 (многоканальный); график работы: с 10.00 до 20.00 без перерыва и выходных

Контактные лица:

Генеральный директор: Иванов Егор Васильевич

Исполнительный директор: Дубровский Олег Иванович

Директор по маркетингу: Антонова Варвара Степановна

Сотрудники: на настоящий момент штат компании составляет 510 сотрудников.

Оргструктура книготорговой фирмы «Либерия» имеет следующий вид:

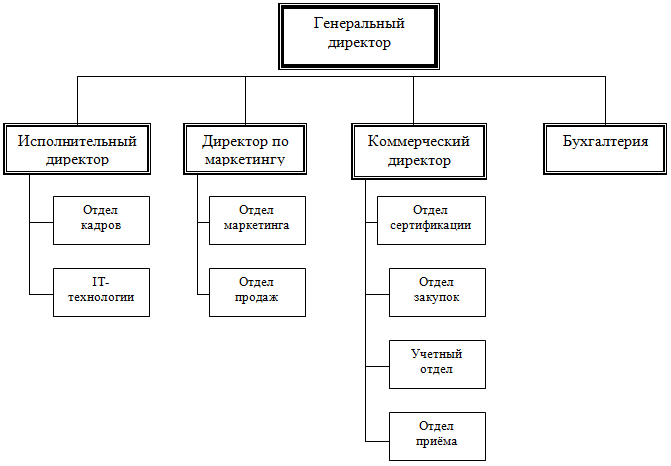


Рисунок 1.1 – Организационная диаграмма

Фирма «Либерия» занимается продажей дисков и книг. Закупка книг осуществляется у крупных издательств, например ИД «Нева», ИД «Росман», ИД «Армада» ИД «Рипол»; диски закупаются оптом, например, у компании «Бука», «Руссобит-М», «Акелла» и т. п. Продажа осуществляется как через Интернет, так и в магазинах, так же возможна доставка необходимой книги/диска в один из магазинов города

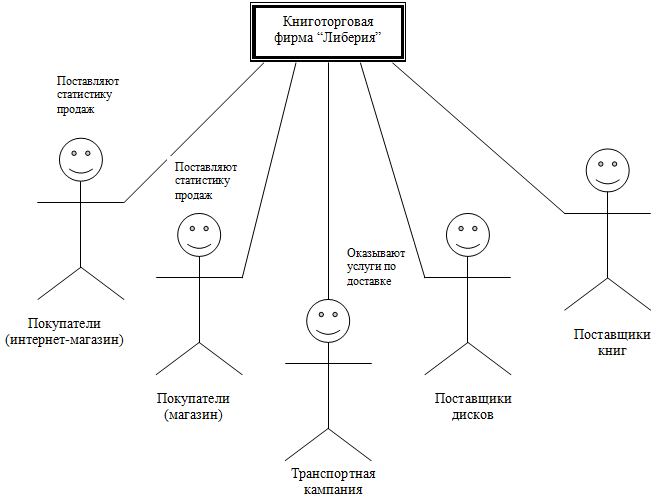


Рисунок 1.2 – Физическая диаграмма

Описание бизнес-процессов компании.

1.Бизнес-процесс «Доставка в магазин»

1.1.Поставщики связываются с отделом закупок, в свою очередь отдел закупок ориентируется на учетный отдел, в котором статистически подсчитывается количество необходимых книг и дисков (расчет берется на основании данных по продажам в магазине и интернет-магазине), а также, учитывается количество новинок.

1.2.Получив заказ, поставщики его выполняют, а данные о заключенной сделке попадают к менеджеру отдела маркетинга, оценивающему затраты.

1.3.Заказ развозят по магазинам, учетный отдел фиксирует это, переводя товар в отдел приема.

2.Бизнес-процесс «Оформление документации»

2.1.После транспортировки, как сказано выше, отдел приема вместе с отделом IT-технологий оформляют новые поступления.

2.2.Происходит подсчет книг и дисков, поступивших в наличие.

2.3.IT-инженеры вносят книги и диски в базы данных, учитывая, для книг: присвоенный код, жанр, автора, издательство; для дисков: также код, тип (музыка, игрушка, фильм), отсюда жанр для каждого типа, записывающую кампанию, исполнителя (для музыки), название (игры и фильмы) и цену.

2.4. Сертификацией занимается соответствующий отдел, в неё входит:

* Сертификация дисков (подтверждение лицензионного качества);
* Проверка диска (наличие царапин, пригодность для прослушивания);
* Проверка книг (механические повреждения) и т.п..

3.Бизнес-процесс «Продажа»

Данный бизнес-процесс делится на две части: Интернет-продажи и Магазин.

3.3.1 Продажа в магазине

1. После того как покупатель из всего ассортимента выбирает то, что надо, он идет на кассу, там его обслуживает кассир, происходит оплата и выдается чек.
2. При покупке диска, консультант продавец оформляет гарантию сроком на две недели, на кассе оплачивается диск и выдается чек плюс гарантия к нему.

3.3.2 Интернет-продажа

* 1. В Инетернет-магазине, который контролируют IT-инженеры, пользователь заходит на сайт.
  2. Обязательно регистрируется!
  3. Выбирает из списков то, что понравилось.
  4. Подтверждает покупку, транспортная кампания привозит данный заказ на дом, происходит обмен деньги-товар, но так же выдается и чек и гарантия, что бы обезопасить и покупателя и магазин.

**1.2. Разработка функциональной модели предметной области**

BPwin – мощный инструмент моделирования, который используется для анализа, документирования и реорганизации сложных бизнес-процессов. Модель, созданная средствами BPwin, позволяет четко документировать различные аспекты деятельности – действия, которые необходимо предпринять, способы их осуществления, требующиеся для этого ресурсы и др. Таким образом, формируется целостная картина деятельности предприятия – от моделей организации работы в маленьких отделах до сложных иерархических структур.

Построение модели системы средствами BPWIN.

Тип методологии моделирования –IDEF0. Если необходимо, то можно использовать и DFDи IDEF3. Все надписи должны быть выполнены на русском языке.

Примерная структура (сложность) модели в BPWIN:

1. Контекстная диаграмма
2. Диаграмма второго уровня (результат декомпозиции контекстной диаграммы); она должна содержать не менее четырёх работ.
3. Не менее трёх диаграмм третьего уровня (результат декомпозиции диаграммы второго уровня). На каждой из диаграмм третьего уровня должно быть не менее двух-трёх работ.

**Пример разработки функциональной модели предметной области**

*Построение диаграммы бизнес-процессов при помощи BPWin*

Модель работы представляет собой систему наборов работ, где каждая из работ – это совокупность нескольких объектов или наборов объектов. Работы (Activity) обозначают поименованные процессы, функции или задачи, которые происходят в течение определенного времени и имеют распознаваемые результаты. Работы изображаются в виде прямоугольников. Все работы должны быть названы и определены. При создании новой модели (меню File/New) автоматически создается контекстная диаграмма с единственной работой, изображающей систему в целом (рисунок 1.3).

В диаграмме описания бизнес-процесса можно подключать различные типы стрелок (Arrow) к работам. Стрелки описывают взаимодействие работ и представляют собой некую информацию. На данной диаграмме:

1. Вход (Input) материал или информация, которые используются или преобразуются работой для получения результата (выхода).

* книгопечатная продукция – книги, поступающие из издательства;
* диски.

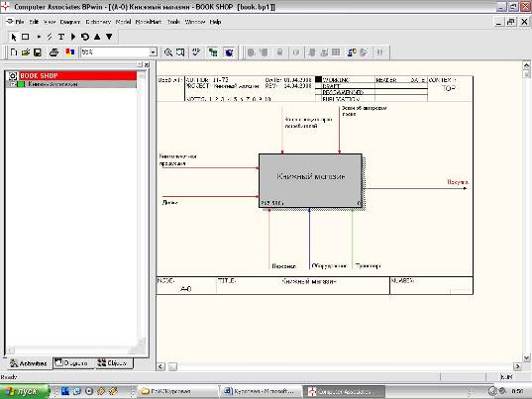


Рисунок 1.3 – Система «Книжный магазин» в общем виде

1. Управление (Control) — правила, стратегии, процедуры или стандарты, которыми руководствуется работа. В данной диаграмме это законы

* Закон о защите прав потребителей;
* Закон об авторском праве;

1. Выход (Output) — материалы или информация, которые производятся работой.

* Конечная цель – покупка;

1. Механизм (Mechanism) — ресурсы, которые выполняют работу.

* Персонал – сотрудники, выполняющие работу.
* Оборудование необходимое в работе (компьютеры, кассовые аппараты)
* Транспорт

Систему разбиваем на 4 части.

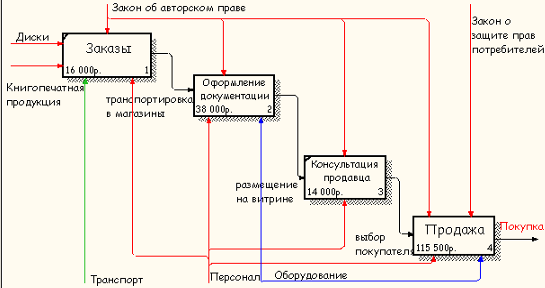


Рисунок 1.4 – Основные бизнес-процессы

Стрелки между работами обозначают состояние системы после выполнения каждого из этапов. Для того чтобы детально показать ИС можно несколько блоков разбить на составные части. Разобьем блок «Оформление документации» на 3 части:

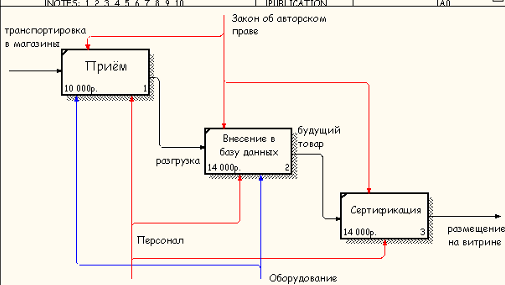


Рисунок 1.5 – Оформление документации

При оформлении документов учитывается закон об авторском праве, интересы автора либо исполнителя. Персонал принимает непосредственное участие.

«Продажа» состоит из 2 процессов (рисунок 1.6): Продажа, которая осуществляется через Интернет и продажи непосредственно в магазине (рисунок 1.7). При этом продажа через Интернет включает доставку при помощи транспортной кампании. Это отражается в стоимостном анализе (пункт 1.3.).

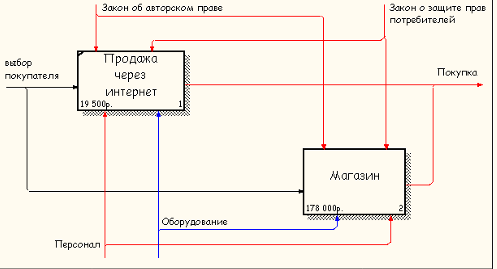


Рисунок 1.6 – Блок «Продажа»

На блок «Продажа» влияет и закона об авторском праве и закон защиты прав потребителей, основную работу выполняет персонал при помощи необходимого оборудования.

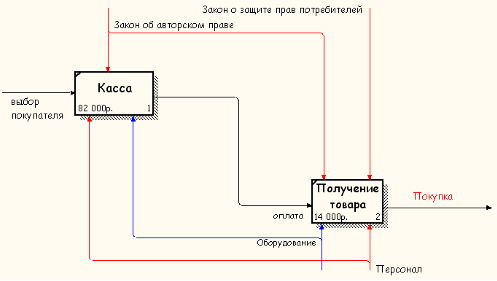


Рисунок 1.7 – Блок «Магазин»

Что бы рассмотреть полностью модель, воспользуемся NodeTree (Диаграмма деревьев узлов), которая показывает иерархию работ в модели и позволяет рассмотреть всю модель целиком, хотя не показывает взаимосвязи между работами. Для этого из меню диаграмм (Diagram) выбираем «AddNodeTree» (Добавить дерево узлов), следуя инструкциям, получаем:

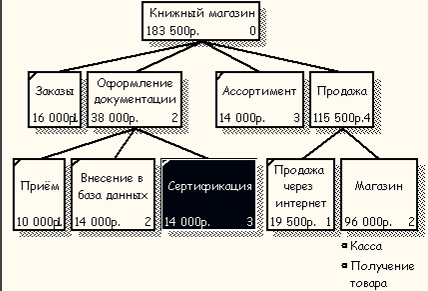


Рисунок 1.8 – Диаграмма деревьев узлов

Эта диаграмма наглядно отражает все процессы информационной системы, включая и затраты по каждому из них; связи между блоками.

1.3. Стоимостный анализ

После того как модель построена, необходимо просчитать затраты начального уровня, и затраты на содержание модели в будущем. Для этого система будет оцениваться через количество. BPwin предоставляет аналитику два инструмента для оценки модели — стоимостный анализ, основанный на работах (ActivityBasedCosting, ABC), и свойства, определяемые пользователем (UserDefinedProperties, UDP). В данной работе используется только функциональное оценивание – ABC – это технология выявления и исследования стоимости выполнения той или иной функции (действия).

ABC включает следующие основные понятия:

* Объект затрат — причина, по которой работа выполняется, обычно основной выход работы. В данном случае это покупка книги либо диска.
* Двигатель затрат — характеристики входов и управлений работы;
* Центры затрат, которые можно трактовать как статьи расхода.

**Примерная структура:**

Необходимо создать отчёт по модели и выполнить стоимостной анализ.

Сумма издержек должна быть отображена на каждой из работ.

**Пример стоимостного анализа**

Для модели «Книжный магазин» можно выявить 5 центров затрат:

1. Исходные расходы это первоначальные затраты при создании компании: на закупку оборудования, наем персонала и т.п.

2. Заработная плата – ежемесячное отчисление для всех сотрудников.

3. Накладные расходы, к ним относятся расходы на оборудование (компьютеры, кассовые аппараты).

4. Постоянные расходы – это расходы на отопление, освещение, налоги и т.д.

5. Транспорт, необходимый для доставки продукции.

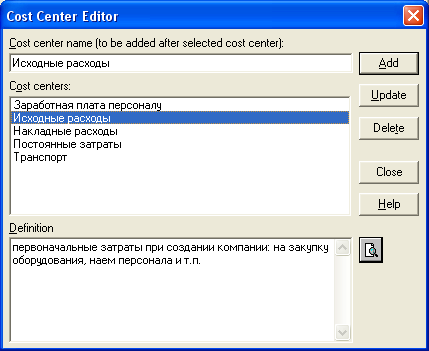


Рисунок 1.9 – Редактор центра затрат

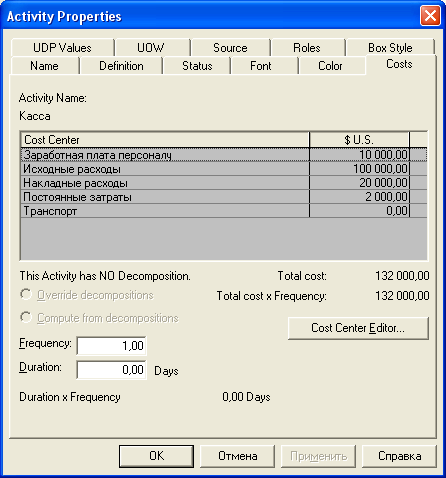


Рисунок 1.10 – Затраты для процесса «Касса»

Таким образом, прописываются затраты по всем блокам, начиная с нижнего узла.

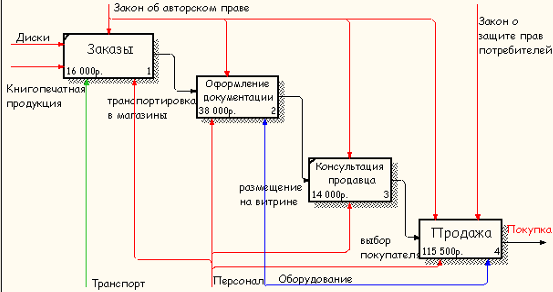


Рисунок 1.11 – Полученные затраты

Просчитав все блоки в начальном будет итоговая цена:

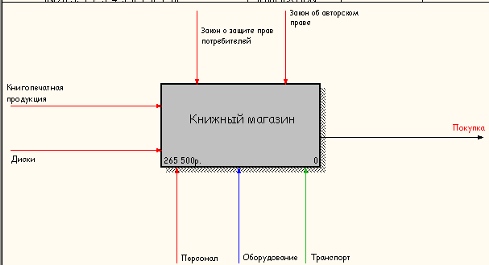


Рисунок 1.12 – Итоговая цена

Формирование отчетов

Для создания отчета необходимо Tools – ReportBuilder – ReportBuilder.

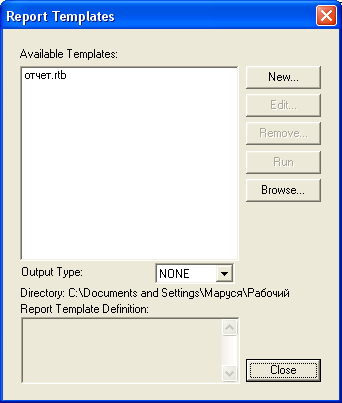


Рисунок 1.13 – Окно отчета

Нажимаем New (рисунок 1.14), в этом окне:

1. Задаем свойства шаблона и формат.
2. Настраиваемсекции: Model, Activity, Cost Centre, Arrow и Picture
3. Задаем свойства секций (имя, шрифт, цвет и т.д.)
4. Задаем таблицы для пояснения содержания секций
5. Задаем свойства таблиц

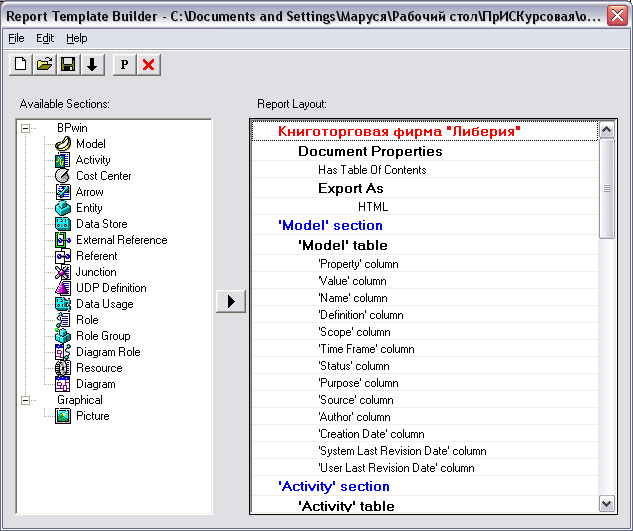


Рисунок 1.14 – Параметры отчета

Далее нажимаем Run и получаем отчет в формате HTML.

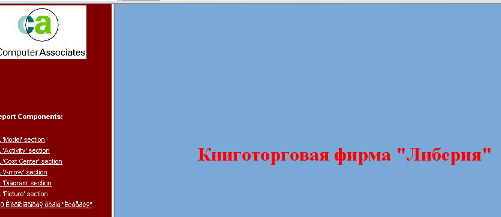


Рисунок 1.15 – Готовый отчет

**1.4. Описание и сравнение программ-аналогов (не менее 3 программ)**

Выполнить в следующей последовательности:

* указать веб ресурс, представляющий данную программу;
* кратко описать функции программ-аналогов;
* выделить достоинства и недостатки каждой, указать, почему их нельзя использовать для решения обозначенной проблемы;
* выделить критерии, значимые для решаемой задачи;
* представить результаты сравнения в наглядном виде (например, в виде таблицы).

**1.5.Техническое задание**

Этот параграф должен содержать спецификацию требований.

**Данный раздел оформляется в виде Приложения 1** к курсовому проекту.

В соответствии с ГОСТ 34.602-89 Техническое задание включает в себя следующие разделы:

1. Общие сведения.
2. Назначение и цели создания системы.
3. Характеристика объектов автоматизации.
4. Требования к системе.
5. Состав и содержание работ по созданию системы.
6. Порядок контроля и приёмки системы.
7. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие.
8. Требования к документированию.
9. Источники разработки.

Для заполнения разделов технического задания проектировщикам необходимо выполнить целый ряд работ на предыдущих этапах проектирования таких, как «Формирование требований к АС» и «Разработка концепции АС». Должны быть уже на этих ранних стадиях построены модели будущей системы, правда, пока укрупненные, которые позволят определить требования к будущей системе и ее подсистемам.

**Выводы по аналитической части**

Выводы должны быть конкретными, то есть вытекать из содержания раздела. Выводы описательного характера недопустимы, например, « ….В этом разделе мы рассмотрели….».

**2. рактическая часть**

**2.1. Проектирование диаграммы вариантов использования (диаграммы прецедентов)**

В этом пункте пояснительной записки приводится и описывается последовательность действий по созданию диаграммы вариантов использования информационной подсистемы и сама диаграмма (актеры, прецеденты и т. д.). Выбирается один, наиболее важный для реализации информационной подсистемы, вариант использования. Для выбранного варианта использования производится спецификация функциональных требований с помощью текстовых сценариев.

**Вариант использования** (прецедент)представляет собой последовательность действий (транзакций), выполняемых системой в ответ на собы­тие, инициируемое некоторым внешним объектом (действующим лицом). Вариант использования описывает типичное взаимодей­ствие между пользователем и системой. В простейшем случае ва­риант использования определяется в процессе обсуждения с пользо­вателем тех функций, которые он хотел бы реализовать.

**Действующее лицо** (actor)–это роль, которую пользователь играет по отношению к системе. Действующие лица представля­ют собой роли, а не конкретных людей или наименования работ. Несмотря на то, что на диаграммах вариантов использования они изображаются в виде стилизованных человеческих фигурок, дей­ствующее лицо может также быть внешней системой, которой необходима информация от данной системы. Показывать на ди­аграмме действующих лиц следует только в том случае, когда им действительно необходимы некоторые варианты использования.

Действующие лица делятся на три основных типа – пользова­тели системы, другие системы, взаимодействующие с данной, и время. Время становится действующим лицом, если от него зави­сит запуск каких-либо событий в системе.

Для наглядного представления вариантов использования в ка­честве основных элементов процесса разработки программного обеспечения (ПО) информационных систем применяют­ся диаграммы вариантов использования. На рисунке 2.1 показан при­мер такой диаграммы для банковской системы с банкоматами.

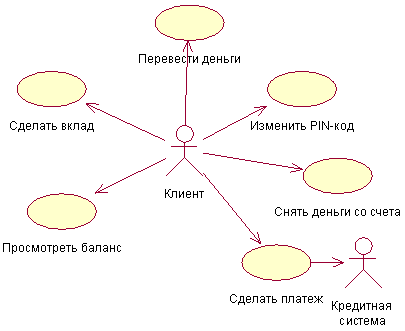


Рисунок 2.1 – Пример диаграммы вариантов использования

для банковской системы с банкоматами

На данной диаграмме человеческие фигурки обозначают дей­ствующих лиц, овалы – варианты использования, а линии и стрел­ки – различные связи между действующими лицами и варианта­ми использования.

На этой диаграмме показаны два действующих лица: клиент и кредитная система. Существуют также шесть основных дей­ствий, выполняемых моделируемой системой: перевести деньги, сделать вклад, снять деньги со счета, просмотреть баланс, изме­нить PIN-код и сделать платеж.

На диаграмме вариантов использования показано взаимодей­ствие между вариантами использования и действующими лица­ми. Она отражает требования к системе с точки зрения пользова­теля. Таким образом, варианты использования – это функции, выполняемые системой, а действующие лица – это заинтересо­ванные лица (stakeholders) по отношению к создаваемой систе­ме. Такие диаграммы показывают, какие действующие лица ини­циируют варианты использования. Из них также видно, когда действующее лицо получает информацию от варианта использо­вания. Данная диаграмма, например, отражает взаимодействие между вариантами использования и действующими лицами бан­ковской системы. В сущности, диаграмма вариантов использо­вания иллюстрирует требования к системе. В нашем примере кли­ент банка инициирует большое количество различных вариан­тов использования: «Снять деньги со счета», «Перевести деньги», «Сделать вклад», «Просмотреть баланс» и «Изменить PIN-код». От варианта использования «Сделать платеж» стрелка указывает на «Кредитную систему». Действующими лицами могут быть внеш­ние системы, и потому в данном случае «Кредитная система» показана именно как действующее лицо − она внешняя для бан­ковской системы. Направленная от варианта использования к дей­ствующему лицу стрелка показывает, что вариант использования предоставляет некоторую информацию, используемую действу­ющим лицом. В данном случае вариант использования «Сделать платеж» предоставляет «Кредитной системе» информацию об оп­лате по кредитной карточке.

Все варианты использования, так или иначе, связаны с внешни­ми требованиями к функциональности системы. Варианты исполь­зования всегда следует анализировать вместе с действующими ли­цами системы, определяя при этом реальные задачи пользовате­лей и рассматривая альтернативные способы решения этих задач.

Действующие лица могут играть различные роли по отноше­нию к варианту использования. Они могут пользоваться его ре­зультатами или могут сами непосредственно в нем участвовать. Значимость различных ролей действующего лица зависит от того, каким образом используются его связи.

Конкретная цель диаграмм вариантов использования – это документирование вариантов использования (все, входящее в сферу применения системы), действующих лиц (все вне этой сфе­ры) и связей между ними. Разрабатывая диаграммы вариантов использования, старайтесь придерживаться следующих правил:

* не моделируйте связи между действующими лицами. По оп­ределению действующие лица находятся вне сферы действия сис­темы. Это означает, что связи между ними также не относятся к ее компетенции;
* не соединяйте стрелкой два варианта использования непос­редственно. Диаграммы данного типа описывают только, какие варианты использования доступны системе, а не порядок их выполнения. Для отображения порядка выполнения вариантов ис­пользования применяют диаграммы деятельностей;
* каждый вариант использования должен быть инициирован действующим лицом. Это означает, что всегда имеется стрелка, начинающаяся на действующем лице и заканчивающаяся на ва­рианте использования.

Хорошим источником для идентификации вариантов исполь­зования служат внешние события. Следует начать с перечисления всех событий, происходящих во внешнем мире, на которые систе­ма должна каким-то образом реагировать. Какое-либо конкрет­ное событие может повлечь за собой реакцию системы, не требую­щую вмешательства пользователей, или, наоборот, вызвать чисто пользовательскую реакцию. Идентификация событий, на которые необходимо реагировать, помогает идентифицировать варианты использования.

Варианты использования начинают описывать, что должна будет делать система. Для того чтобы фактически разработать систему, однако потребуются более конкретные детали. Эти де­тали описываются в документе, называемом «поток событий» (flowofevents). Целью потока событий является документирова­ние процесса обработки данных, реализуемого в рамках вариан­та использования. Этот документ подробно описывает, что бу­дут делать пользователи системы и что – сама система.

Хотя поток событий и описывается подробно, он также не должен зависеть от реализации. Цель – описать то, что будет де­лать система, а не как она будет делать это. Обычно поток собы­тий включает:

* краткое описание;
* предусловия (pre-conditions);
* основной поток событий;
* альтернативный поток событий;
* постусловия (post-conditions). Последовательно рассмотрим эти составные части.

*Краткое описание*. Каждый вариант использования должен иметь краткое опи­сание того, что он будет делать. Например, вариант использова­ния «Перевести деньги» может содержать следующее описание:

Вариант использования «Перевести деньги» позволяет клиен­ту или служащему банка переводить деньги с одного счета до вос­требования или сберегательного счета на другой.

*Предусловия*. Предусловия варианта использования – это такие условия, которые должны быть выполнены, прежде чем вариант исполь­зования начнет выполняться сам. Например, таким условием может быть выполнение другого варианта использования или наличие у пользователя прав доступа, требуемых для запуска это­го. Не у всех вариантов использования бывают предусловия.

Ранее упоминалось, что диаграммы вариантов использования не должны отражать порядок их выполнения. С помощью пре­дусловий, однако, можно документировать и такую информацию. Так, предусловием одного варианта использования может быть то, что в это время должен выполняться другой.

*Основной и альтернативный потоки событий*. Конкретные детали вариантов использования описываются в основном и альтернативных потоках событий. Поток событий поэтапно описывает, что должно происходить во время выпол­нения заложенной в варианты использования функциональнос­ти. Поток событий уделяет внимание тому, что будет делать сис­тема, а не как она будет делать это, причем описывает все это с точки зрения пользователя. Основной и альтернативный потоки событий включают следующее описание:

* каким образом запускается вариант использования;
* различные пути выполнения варианта использования;
* нормальный, или основной, поток событий варианта ис­пользования;
* отклонения от основного потока событий (так называемые альтернативные потоки);
* потоки ошибок;
* каким образом завершается вариант использования.

Например, поток событий варианта использования «Снять деньги со счета» может выглядеть следующим образом:

*Основной поток*. Вариант использования начинается, когда клиент вставляет свою карточку в банкомат.

1. банкомат выводит приветствие и предлагает клиенту ввести свой персональный PIN-код;
2. клиент вводит PIN-код;
3. банкомат подтверждает введенный код. Если код не подтверж­ден, выполняется альтернативный поток событий А1;
4. банкомат выводит список доступных действий:
   1. сделать вклад;
   2. снять деньги со счета;
   3. перевести деньги.
5. клиент выбирает пункт «Снять деньги со счета»;
6. банкомат запрашивает, сколько денег надо снять;
7. клиент вводит требуемую сумму;
8. банкомат определяет, имеется ли на счете достаточно денег. Если денег недостаточно, выполняется альтернативный поток А2. Если во время подтверждения суммы возникают ошибки, выполняется поток ошибок Е1;
9. банкомат вычитает требуемую сумму из счета клиента;
10. банкомат выдает клиенту требуемую сумму наличными;
11. банкомат возвращает клиенту его карточку;
12. банкомат печатает чек для клиента;
13. вариант использования завершается.

*Альтернативный поток А1*. Ввод неправильногоPIN-кода:

1. банкомат информирует клиента, что код введен неправильно;
2. банкомат возвращает клиенту его карточку;
3. вариант использования завершается.

*Альтернативный вариант использования А2*. Недостаточно денег

на счете:

1. банкомат информирует клиента, что денег на его счете недоста­точно;
2. банкомат возвращает клиенту его карточку;
3. вариант использования завершается.

*Поток ошибок El*. Ошибка в подтверждении запраши­ваемой суммы:

1. банкомат сообщает пользователю, что при подтверждении зап­рашиваемой суммы произошла ошибка, и дает ему номер телефона служ­бы поддержки клиентов банка;
2. банкомат заносит сведения об ошибке в журнал ошибок. Каждая запись содержит дату и время ошибки, имя клиента, номер его счета и код ошибки;
3. банкомат возвращает клиенту его карточку;
4. вариант использования завершается.

*Постусловия*. Постусловиями называются такие условия, которые всегда должны быть выполнены после завершения варианта использо­вания. Например, в конце варианта использования можно поме­тить флажком какой-нибудь переключатель. Информация тако­го типа входит в состав постусловий. Как и для предусловий, с помощью постусловий можно вводить информацию о порядке выполнения вариантов использования системы. Если, например, после одного из вариантов использования должен всегда выполняться другой, это можно описать как постусловие. Такие усло­вия имеются не у каждого варианта использования.

*Спецификация функциональных требований с помощью текстовых сценариев*. Содержание варианта использования может быть дополнительно специфици­ровано в форме пояснительного текста, который раскрывает смысл или се­мантику выполняемых действий при выполнении варианта использования.

Такой пояснительный текст применительно к диаграммам вариантов исполь­зования получил специальное название текста-сценария или просто − сценария.

Сценарий (scenario)– специально написанный текст, описывающий поведение моделируемой системы в форме последова­тельности выполняемых действий актеров и самой системы.

В настоящее время предлагаются различные способы (шаблоны) для написания подобных сценариев. Один из таких шаблонов (таблица 2.1) рекомендуется разработчикам для применения на начальных этапах концептуального моделирования.

Таблица 2.1 – Шаблон для написания сценария отдельного прецедента

| Главный раздел | Раздел «Типичный ход событий» | Раздел  «Исключения» | Раздел  «Примечания» |
| --- | --- | --- | --- |
| Имя варианта использования  Актеры  Цель  Краткое описание Тип  Ссылки на другие варианты использования | Типичный ход собы­тий, приводящий к успешному выполнению данного варианта использования | Исключение № 1 | Примечание № 1 |
| Исключение № 2 | Примечание № 2 |
| ... | ... |
| Исключение № n | Примечание № n |

Для удобства записи и чтения сценария часто используют модификацию дан­ного шаблона, располагая разделы последовательно сверху вниз. При этом написание сценариев модели начинают с базовых или основных вариантов использования, после чего рассматриваются сценарии второстепенных или включаемых вариантов использования.

Сценарии расширяющих вариантов использования обычно помещаются в разделе исключений базового варианта использования. Желательно, чтобы каждому исключению соответствовал от­дельный вариант использования, который соединяется с базовым отношением расширения. В противном случае будет нарушено логическое соответствие между текстами сценариев и диаграммой вариантов использования.

Если поведение некоторого варианта использования представляется разра­ботчику тривиальным, то и написание для него сценария не является необходимым. В этом случае можно ограничиться простым примечанием к соот­ветствующему исключению. При написании сценариев вариантов исполь­зования важно понимать, что сценарий должен дополнять или уточнять диаграмму вариантов использования, но не заменять ее полностью. В про­тивном случае будут потеряны достоинства визуального представления мо­делей.

*Примечание. При разработке диаграмм вариантов использования следует вовремя остановиться и не пытаться на этапе концептуального моделирования изображать логику или последова­тельность выполнения отдельных действий. Для последней цели в языке UML специально предназначены диаграммы последовательности и деятельности, которые рассматриваются далее.*

*Пример сценария для системы продажи товаров в интернет-магазине*. Для иллюстрации рассмотренного материала дополним диаграмму вариантов использования для системы продажи товаров в интер­нет-магазине (рисунок 2.2) текстовым сценарием, написанным на основе предложенного шаблона (таблица 2.2).

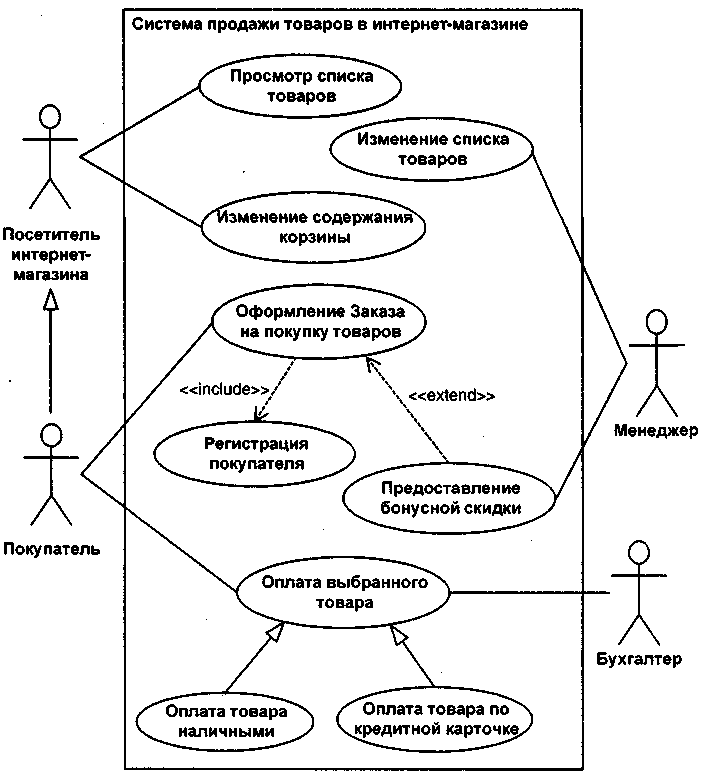


Рисунок 2.2 – Диаграмма вариантов использования

для системы продажи товаров в интернет-магазине

Сценарий будет дополнять построен­ную диаграмму, раскрывая содержание отдельных действий и их логическую последовательность, что характеризует внешнее представление о процессе функционирования системы.

В качестве примера выберем достаточно простой вариант использования «Просмотр списка товаров», для которого соответствующий сценарий удобно представить в виде двух отдельных таблиц, каждая из которых опи­сывает отдельный раздел шаблона. В главном разделе сценария (таблица 2.2) указывается имя рассматриваемого варианта использования, имена взаимо­связанных с ним актеров, цель выполнения, тип и ссылки на другие вариан­ты использования.

Таблица 2.2 – Главный раздел сценария выполнения варианта использования «Просмотр списка товаров»

|  |  |
| --- | --- |
| Вариант использования | Просмотр списка товаров |
| Актеры | Посетитель интернет-магазина |
| Краткое описание | Получение требуемой информации о товарах, пред­ставленных в интернет-магазине |
| Цель | Посетитель интернет-магазина просматривает ин­формацию о товарах. Система обеспечивает доступ к любому товару и удобную навигацию по различным категориям товаров |
| Тип | Базовый |
| Ссылки на другие вариан­ты использования | Отсутствуют |

Поскольку для варианта использования «Просмотр списка товаров» отсутст­вуют включаемые варианты использования, ссылки на другие варианты ис­пользования также будут отсутствовать. В следующем разделе сценария (таблица 2.3) описывается последовательность действий, приводящая к успеш­ному выполнению рассматриваемого варианта использования. В качестве инициатора действий выступает актер «Посетитель интернет-магазина». Для удобства каждое действие помечается порядковым номером в последова­тельности.

Таблица 2.3 – Раздел типичного хода событий сценария выполнения вариантаиспользования «Просмотр списка товаров»

| Действия актеров | Отклик системы |
| --- | --- |
| 1. Посетитель загружает исходную стра­ницу интернет-магазина в браузер | 2. Система отображает исходную стра­ницу интернет-магазина |
| 3. Посетитель интернет-магазина вы­бирает категорию то­варов | 4. Система отображает информацию о выбранной категории товаров |
| 5. Посетитель интернет-магазина вы­бирает интересующий товар | 6. Система отображает общую инфор­мацию о выбранном товаре |
| 7. Посетитель интернет-магазина вы­бирает просмотр детальной инфор­мации об интересующем товаре | 8. Система отображает детальную ин­формацию о выбранном товаре |
| 9. Посетитель интернет-магазина мо­жет пожелать вернуться на исходную страницу интернет-магазина | 10. Система отображает исходную стра­ницу интернет-магазина |

Поскольку для рассматриваемого варианта использования отсутствуют рас­ширяющие варианты использования, секции исключений останутся пустыми.

*Связи между вариантами использования и действующими ли­цами*. В языке UML для вариантов использования и действующих лиц поддерживается несколько типов связей. Это связи комму­никации (communication), включения (include), расширения (extend) и обобщения (generalization).

Связь коммуникации–это связь между вариантом использо­вания и действующим лицом. На языке UML связи коммуника­ции показывают с помощью однонаправленной ассоциации (ли­нии со стрелкой). Направление стрелки позволяет понять, кто инициирует коммуникацию.

Связь включенияприменяется в тех ситуациях, когда имеется какой-либо фрагмент поведения системы, который повторяется более чем в одном варианте использования. С помощью таких связей обычно моделируют многократно используемую функци­ональность. В примере банковской системы варианты использо­вания «Снять деньги со счета» и «Сделать вклад» должны опоз­нать (аутентифицировать) клиента и его PIN-код перед осуще­ствлением самой транзакции. Вместо того чтобы подробно описывать процесс аутентификации для каждого из них, можно поместить эту функциональность в свой собственный вариант использования под названием «Аутентифицировать клиента».

Связь расширенияприменяется при описании изменений в нор­мальном поведении системы. Она позволяет варианту использо­вания только при необходимости применять функциональные воз­можности другого.

На языке UML связи включения и расширения показывают в виде зависимостей с соответствующими стереотипами (рисунке 2.3).

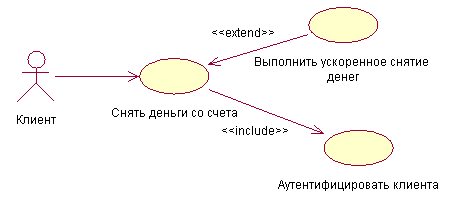


Рисунок 2**.**3 – Связи использования и расширения

Сформулируем следующие определения.

**Отношение включения** (include) – специфицирует тот факт, что некоторый вариант использования содержит поведение, определенное в другом варианте использования.

**Отношение расширения** (extend) – определяет взаимосвязь одного варианта использования с некоторым другим вариантом использования, функциональность или поведение которого задействуется первым не всегда, а только при выполнении некоторых дополнительных условий.

**Стереотип (stereotype)**− элемент модели, который расширяет се­мантику базового элемента метамодели языка UML.

В моделях языка UML стереотипы могут быть представлены в форме текста, заключенного в двойные угловые кавычки и размещенно­го выше или перед именем элемента модели (рисунок 1.3). Первая буква текста имени стереотипа не должна быть прописной буквой. Если применяются несколько стереотипов, то их имена изображаются в форме разделенного запятыми списка с парой кавычек.

С помощью **связи обобщения** показывают, что у нескольких действующих лиц имеются общие черты. Например, клиенты могут быть двух типов: корпоративные и индивидуальные. Эту связь можно моделировать с помощью нотации, показанной на рисунке 2.4.

Нет необходимости всегда создавать связи этого типа. В об­щем случае они нужны, если поведение действующего лица од­ного типа отличается от поведения другого постольку, посколь­ку это затрагивает систему. Если оба подтипа применяют одни и те же варианты использования, показывать обобщение действу­ющего лица не требуется.

Варианты использования являются необходимым средством на стадии формирования требований к программному обеспечению АИС. Каждый вариант ис­пользования – это потенциальное требование к системе, и пока оно не выявлено, невозможно запланировать его реализацию.

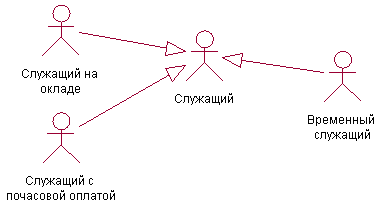


Рисунок 2**.**4 – Обобщение действующего лица

Примечание – Все последующие диаграммы UML будут строиться для выбранного вами одного наиболее важного для реализации информационной подсистемы варианта использования.

**Внимание! Ни в коем случае не следует включать в текст первого и последующих пунктов пояснительной записки определения (что такое актер, что такое диаграмма вариантов использования и пр.).**

**2.2. Разработка диаграммы последовательностей**

В этом пункте пояснительной записки приводится и описывается последовательность действий по созданию диаграммы последовательности для одного из прецедентов информационной подсистемы (см. лабораторное занятие «Построение диаграмм последовательности (Sequencediagram)»). Выбор рассматриваемого прецедента должен быть обоснован ранее во втором разделе пояснительной записки.

Диаграмма последовательности – одна из видов диаграмм взаимодействия.

Диаграммы взаимодействия (interactiondiagrams) описывают поведение взаимодействующих групп объектов.

Как правило, диаграмма взаимодействия охватывает поведе­ние объектов в рамках только одного варианта использования. На такой диаграмме отображается ряд объектов и те сообщения, которыми они обмениваются между собой.

**Сообщение** (message)*-* средство, с помощью которого объект-отправитель запрашивает у объекта-получателя выполнение од­ной из его операций.

**Информационное**(informative) **сообщение**–сообщение, снаб­жающее объект-получатель информацией для обновления его состояния.

**Сообщение-запрос** (interrogative)*-* сообщение, запрашивающее выдачу информации об объекте-получателе.

**Императивное** (imperative) **сообщение**–сообщение, запраши­вающее у объекта-получателя выполнение действий.

Существуют два вида диаграмм взаимодействия: диаграммы последовательности (sequencediagrams) и кооперативные диаг­раммы (collaborationdiagrams).

***Диаграммы последовательности***. Диаграммы последовательности отражают поток событий, происходящих в рамках варианта использования. Например, ва­риант использования «Снять деньги со счета» предусматривает несколько возможных последовательностей, таких как снятие де­нег, попытка снять деньги, не имея их достаточного количества на счете, попытка снять деньги по неправильномуPIN-коду и неко­торых других. Нормальный сценарий снятия некоторой суммы де­нег со счета показан на рисунке 2.5. Под сценарием понимается конк­ретный экземпляр потока событий.

Эта диаграмма последовательности отображает поток собы­тий в рамках варианта использования «Снять деньги со счета». Все действующие лица показаны в верхней части диаграммы; в приве­денном выше примере изображено действующее лицо Клиент (Customer). Объекты, требуемые системе для выполнения вариан­та использования «Снять деньги со счета», также представлены в верхней части диаграммы. Стрелки соответствуют сообщениям, передаваемым между действующим лицом и объектом или между объектами для выполнения требуемых функций.

На диаграмме последовательности объект изображается в виде прямоугольника на вершине пунктирной вертикальной ли­нии. Эта вертикальная линия называется линией жизни (lifeline)объекта. Она представляет собой фрагмент жизненного цикла объекта в процессе взаимодействия.

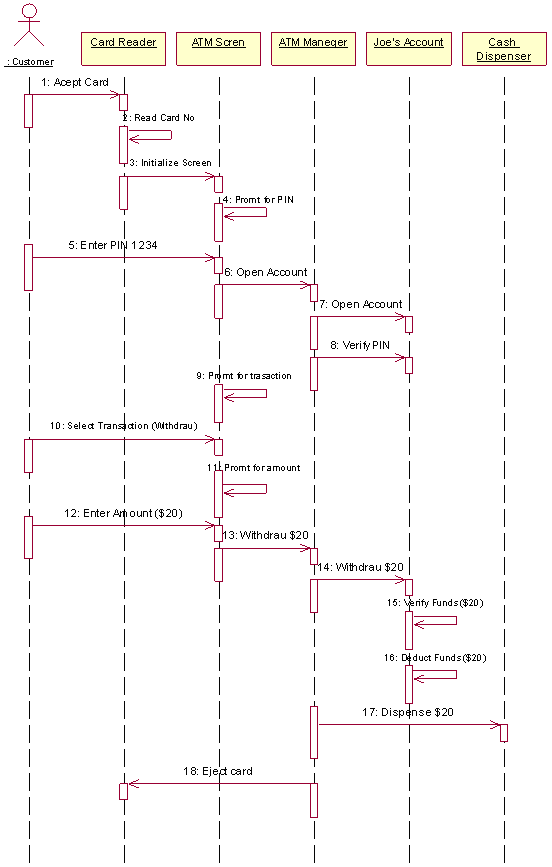


Рисунок 2.5 – Диаграмма последовательности

Каждое сообщение изображается в виде стрелки между линия­ми жизни двух объектов. Сообщения появляются в том порядке, как они показаны на странице, сверху вниз. Каждое сообщение помечается как минимум именем сообщения; при желании можно добавить также аргументы и некоторую управляющую информацию и, кроме того, показать самоделегирование (self-delegation) – сообщение, которое объект посылает самому себе, при этом стрелка сообщения указывает на ту же самую линию жизни.

Хороший способ первоначального обнаружения некоторых объектов – это изучение имен существительных в потоке собы­тий. Можно также прочитать документы, описывающие конкретный сценарий. Поток событий для варианта использования «Снять деньги со счета» говорит о человеке, снимающем некото­рую сумму денег со счета с помощью банкомата.

Не все объекты, показанные на диаграмме, явно присутству­ют в потоке событий. Там, например, может не быть форм для заполнения, но их необходимо показать на диаграмме, чтобы позволить действующему лицу ввести новую информацию в си­стему или просмотреть ее. В потоке событий, скорее всего, также не будет и управляющих объектов (controlobjects). Эти объекты управляют последовательностью событий в варианте использо­вания.

**2.3. Проектирование диаграммы взаимодействия для объектов предметной области**

В этом пункте пояснительной записки приводится и описывается последовательность действий по созданию диаграммы сотрудничества для выбранного прецедента информационной подсистемы.

***Кооперативные диаграммы*** являются вторым видом диаграмм взаимодействия.

Подобно диаграммам последовательности, кооперативные ди­аграммы отображают поток событий через конкретный сценарий вариантаиспользования. Диаграммы последовательности упоря­дочены по времени, а кооперативные диаграммы заостряют вни­мание на связях между объектами. На рисунке 2.6 приведена коопера­тивная диаграмма, описывающая, как клиент снимает деньги со счета.

Как видно из рисунка 2.6, здесь представлена вся та информа­ция, которая была и на диаграмме последовательности, но коо­перативная диаграмма по-другому описывает поток событий. Из нее легче понять связи между объектами, однако труднее уяснить последовательность событий.

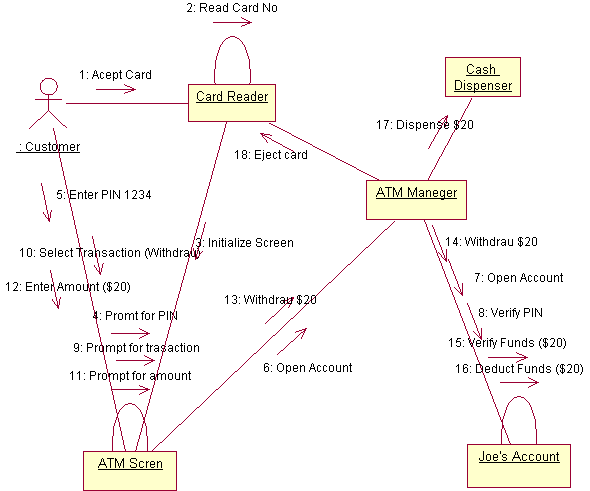


Рисунок 2.6 – Кооперативная диаграмма, описывающая,

как клиент снимает деньги со счета

По этой причине часто для какого-либо сценария создают диаграммы обоих типов. Хотя они служат одной и той же цели и содержат одну и ту же информацию, но представляют ее с раз­ных точек зрения.

На кооперативной диаграмме, так же как и на диаграмме последовательности, стрелки обозначают сообщения, обмен кото­рыми осуществляется в рамках данного варианта использования. Их временная последовательность, однако, указывается путем нумерации сообщений.

**2.4. Разработка логической модели предметной области с помощью диаграмм классов**

В этом пункте пояснительной записки приводится и описывается диаграмма классов для выбранного прецедента информационной подсистемы.

Диаграмма классов определяет типы классов 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. Диаграмма классов для варианта использования «Снять деньги со счета» представлена на рисунке 2.7.

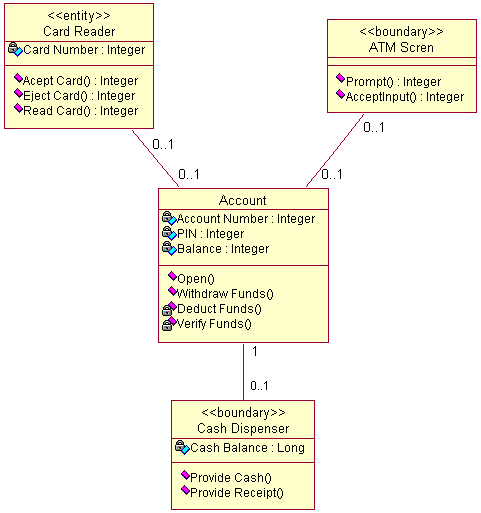


Рисунок 2.7 –Диаграмма классов для варианта использования

«Снять деньги со счета»

На этой диаграмме классов показаны связи между классами, реализующими вариант использования «Снять деньги со счета». В этом процессе задействованы четыре класса: CardReader (уст­ройство для чтения карточек), Account (счет), ATMScreen (экран ATM) и CashDispenser (кассовый аппарат). Каждый класс на диаграмме выглядит в виде прямоугольника, разделенного на три части. В первой содержится имя класса, во второй − его атрибу­ты. В последней части содержатся операции класса, отражающие его поведение (действия, выполняемые классом).

Связывающие классы линии отражают взаимодействие между классами. Так, класс Account связан с классом ATMScreen, пото­му что они непосредственно сообщаются и взаимодействуют друг с другом. Класс CardReader не связан с классом CashDispenser, поскольку они не сообщаются друг с другом непосредственно.

***Стереотипы классов***. Стереотипы – это механизм, позволяющий разделять классы на категории. В языке UML основными стереотипами являются: Boundary (граница), Entity (сущность) и Control (управление).

Граничные классы(boundaryclasses) – это классы, которые расположены на границе системы и окружающей среды. Они включают все формы, отчеты, интерфейсы с аппаратурой (такой, как принтеры или сканеры) и интерфейсы с другими системами.

Для того чтобы найти граничные классы, надо исследовать ди­аграммы вариантов использования. Каждому взаимодействию между действующим лицом и вариантом использования должен соответствовать, по крайней мере, один граничный класс. Именно такой класс позволяет действующему лицу взаимодействовать с системой.

Классы-сущности(entityclasses) отражают основные понятия (абстракции) предметной области и, как правило, содержат хра­нимую информацию. Обычно для каждого класса-сущности со­здают таблицу в базе данных.

Управляющие классы(controlclasses) отвечают за координа­цию действий других классов. Обычно у каждого варианта ис­пользования имеется один управляющий класс, контролирующий последовательность событий этого варианта использования. Уп­равляющий класс отвечает за координацию, но сам не несет в себе никакой функциональности – остальные классы не посылают ему большого количества сообщений. Вместо этого он сам посылает множество сообщений. Управляющий класс просто делегирует ответственность другим классам, по этой причине его часто на­зывают классом-менеджером.

В системе могут быть и другие управляющие классы, общие для нескольких вариантов использования. Например, класс SecurityManager (менеджер безопасности), отвечающий за контроль событий, связанных с безопасностью. Класс TransactionManager (менеджер транзакций) занимается координацией сооб­щений, относящихся к транзакциям с базой данных. Могут быть и другие менеджеры для работы с другими элементами функцио­нирования системы, такими, как разделение ресурсов, распреде­ленная обработка данных или обработка ошибок.

Помимо упомянутых выше стереотипов можно создавать и свои собственные.

***Механизм пакетов***. Пакеты применяют, чтобы сгруппировать классы, обладаю­щие некоторой общностью. Существует несколько наиболее распространенных подходов к группировке. Во-первых, можно груп­пировать их по стереотипу. В таком случае получается один па­кет с классами-сущностями, один с граничными классами, один с управляющими классами и т. д. Этот подход может быть полезен с точки зрения размещения готовой системы, поскольку все на­ходящиеся на клиентских машинах компоненты с граничными классами уже оказываются в одном пакете.

Другой подход заключается в объединении классов по их фун­кциональности. Например, в пакете Security (безопасность) содер­жатся все классы, отвечающие за безопасность приложения. В та­ком случае другие пакеты могут называться EmployeeMaintenance (Работа с сотрудниками), Reporting (Подготовка отчетов) и ErrorHandling (Обработка ошибок). Преимущество этого подхода зак­лючается в возможности повторного использования.

Механизм пакетов применим к любым элементам модели, а не только к классам. Если для группировки классов не приме­нять некоторые эвристики, то она становится весьма произволь­ной. Одна из них, которая в основном используется в UML, – это зависимость. Зависимость между двумя пакетами существует в том случае, если между любыми двумя классами в пакетах суще­ствует любая зависимость. Таким образом, диаграмма пакетов(рисунок 2.8) представляет собой диаграмму, содержащую пакеты классов и зависимости между ними. Строго говоря, пакеты и за­висимости являются элементами диаграммы классов, т. е. диаг­рамма пакетов – это форма диаграммы классов.

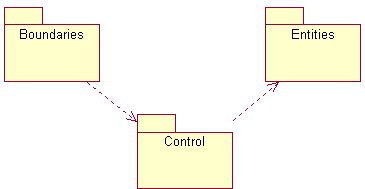


Рисунок 2.8 – Диаграмма пакетов

Зависимость между двумя элементами имеет место в том слу­чае, если изменения в определении одного элемента могут повлечь за собой изменения в другом. Что касается классов, то причины для зависимостей могут быть самыми разными: один класс посы­лает сообщение другому; один класс включает часть данных дру­гого класса; один класс использует другой в качестве параметра операции.

Если класс меняет свой интерфейс, то любое сообще­ние, которое он посылает, может утратить свою силу. Пакеты не дают ответа на вопрос, каким образом можно умень­шить количество зависимостей в вашей системе, однако они по­могают выделить эти зависимости, а после того поработать над снижением их количества. Диаграммы пакетов можно считать ос­новным средством управления общей структурой системы.

Пакеты являются жизненно необходимым средством для боль­ших проектов. Их следует использовать в тех случаях, когда ди­аграмма классов, охватывающая всю систему в целом и разме­щенная на единственном листе бумаги формата А4, становится нечитаемой.

***Атрибуты***. **Атрибут** – это элемент информации, связанный с классом. Например, у класса Company (Компания) могут быть атрибуты Name (Название), Address (Адрес) и NumberOfEmployees (Число служащих).

Атрибуты содержатся внутри класса, поэтому они скрыты от других классов. В связи с этим иногда требуется указать, какие классы имеют право читать и изменять атрибуты. Это свойство называется *видимостью* атрибута (attributevisibility).

У атрибута можно определить четыре возможных значения этого параметра. Рассмотрим каждый из них в контексте приме­ра (рисунок 1.9). Пусть у нас имеется класс Employee с атрибутом Address и класс Company. Возможны следующие спецификаторы доступа к атрибутам и операциям класса:

* **public** (общий, открытый).Это значение видимости пред­полагает, что атрибут будет виден всеми остальными классами. Любой класс может просмотреть или изменить значение атрибу­та. В таком случае класс Company может изменить значение ат­рибута Address класса Employee. В соответствии с нотацией UML общему атрибуту предшествует знак «».
* **private** (закрытый, секретный).Соответствующий атрибут не виден никаким другим классам. Класс Employee будет знать значение атрибута Address и сможет изменять его, но класс Company не сможет его ни увидеть, ни редактировать. Если это понадобится, он должен попросить класс Employee просмотреть или изменить значение этого атрибута, что обычно делается с помощью общих операций. Закрытый атрибут обозначается зна­ком «» в соответствии с нотацией UML.
* **protected** (защищенный).Такой атрибут доступен только самому классу и его потомкам. Допустим, что у нас имеются два типа сотрудников - с почасовой оплатой и на окладе. Таким образом, мы получаем два других класса HourlyEmp и SalariedEmp, являющихся потомками класса Employee. Защищенный атрибут Address можно просмотреть или изменить из классов Employee, HourlyEmp и SalariedEmp, но не из класса Company. Нотация UML для защищенного атрибута - это знак «».
* **packageorImplementation** (пакетный).Предполагает, что данный атрибут является общим, но только в пределах его паке­та. Допустим, что атрибут Address имеет пакетную видимость. В таком случае он может быть изменен из класса Company, только если этот класс находится в том же пакете. Этот тип видимости не обозначается никаким специальным значком.

В общем случае атрибуты рекомендуется делать закрытыми или защищенными. Это позволяет лучше контролировать сам атрибут и код, а также избежать ситуации, когда значение атрибута изменя­ется всеми классами системы. Вместо этого логика изменения атри­бута будет заключена в том же классе, что и сам этот атрибут. Зада­ваемые параметры видимости повлияют на генерируемый код.

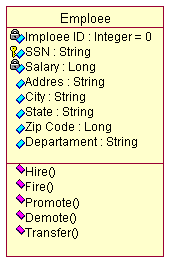


Рисунок 2.9 – Спецификаторы доступа к атрибутам и операциям класса

***Операции***. Операции реализуют связанное с классом поведение. Опера­ция включает три части – имя, параметры и тип возвращаемого значения. Параметры – это аргументы, получаемые операцией «на входе». Тип возвращаемого значения относится к результату дей­ствия операции.

На диаграмме классов можно показывать как имена опера­ций, так и имена операций вместе с их параметрами и типом воз­вращаемого значения. Для того чтобы уменьшить загруженность диаграммы, полезно бывает на некоторых из них показывать толь­ко имена операций, а на других их полную сигнатуру.

В языке UML операции имеют следующую нотацию:

*Имя Операции (аргумент1 : тип данных аргумента1, аргумент2: тип данных аргумента2,...) : тип возвращаемого значения;*

Существует четыре различных типа операций:

* *операции реализации*(implementeroperations) реализуют не­которые бизнес-функции. Такие операции можно найти, иссле­дуя диаграммы взаимодействия. Диаграммы этого типа фокусируются на бизнес-функциях, и каждое сообщение диаграммы, ско­рее всего, можно соотнести соперацией реализации. Каждая операция реализации должна быть легко прослежи­ваема до соответствующего требования. Это достигается на раз­личных этапах моделирования. Операция выводится из сообще­ния на диаграмме взаимодействия, сообщения исходят из под­робного описания потока событий, который создается на основе варианта использования, а последний - на основе требований. Возможность проследить всю эту цепочку позволяет гарантиро­вать, что каждое требование будет реализовано в коде, а каждый фрагмент кода реализует какое-то требование;
* *операции управления*(manageroperations) управляют созданием и уничтожением объектов. В эту категорию попадают конст­рукторы и деструкторы классов;
* *операции доступа*(accessoperations) существуют для того, чтобы просматривать или изменять значения атрибутов других классов. Пусть, например, у нас имеется атрибут Salary класса Employee. Мы не хотим, чтобы все остальные классы могли изменять этот ат­рибут. Вместо этого к классу Employee мы добавляем две операции доступа – GetSalary и SetSalary. К первой из них, являющейся об­щей, могут обращаться и другие классы. Она просто получает зна­чение атрибута Salary и возвращает его вызвавшему ее классу. Опе­рация SetSalary также является общей, она помогает вызвавшему ее классу установить новое значение атрибута Salary. Эта операция может содержать любые правила и условия проверки, которые не­обходимо выполнить, чтобы зарплата могла быть изменена. Такой подход дает возможность безопасно инкапсулировать атрибуты внутри класса, защитив их от других классов, но все же позволяет осуществить к ним контролируемый доступ. Создание операций Get и Set (получения и изменения значения) для каждо­го атрибута класса является стандартом;
* *вспомогательные операции*(helperoperations) – это такие опе­рации класса, которые необходимы ему для выполнения его от­ветственностей, но о которых другие классы не должны ничего знать. Это закрытые и защищенные операции класса.

Чтобы идентифицировать операции, выполните следующие действия:

* изучите диаграммы последовательности и кооперативные диаграммы. Большая часть сообщений на этих диаграммах является операциями реализации. Рефлексивные сообщения будут вспомогательными операциями.
* рассмотрите управляющие операции. Может потребовать­ся добавить конструкторы и деструкторы.
* рассмотрите операции доступа. Для каждого атрибута клас­са, с которым должны будут работать другие классы, надо со­здать операции Get и Set.

**Связь** представляет собой семантическую взаимосвязь между классами. Она дает классу возможность узнавать об атрибутах, операциях и связях другого класса. Иными словами, чтобы один класс мог послать сообщение другому на диаграмме последова­тельности или кооперативной диаграмме, между ними должна существовать связь.

Имеются четыре типа связей, которые могут быть установле­ны между классами: ассоциации, зависимости, агрегации и обоб­щения.

*Ассоциации*(association) – это семантическая связь между клас­сами. Ее рисуют на диаграмме классов в виде обыкновенной ли­нии (рисунке 2.10).



Рисунок 2.10 – Ассоциация

Ассоциации могут быть двунаправленными, как в примере, или однонаправленными. На языке UML двунаправленные ассо­циации рисуют в виде простой линии без стрелок (или со стрелка­ми с обеих ее сторон). На однонаправленной ассоциации изобра­жают только одну стрелку, показывающую ее направление.

Направление ассоциации можно определить, изучая диаграм­мы последовательности и кооперативные диаграммы. Если все сообщения на них отправляются только одним классом и прини­маются только другим классом, а не наоборот, между этими клас­сами имеет место однонаправленная связь. Если хотя бы одно сообщение отправляется в обратную сторону, ассоциация долж­на быть двунаправленной.

Ассоциации могут быть рефлексивными. Рефлексивная ассо­циация предполагает, что один экземпляр класса взаимодейству­ет с другими экземплярами этого же класса.

*Зависимости*(dependency) также отражают связь между клас­сами, но они всегда однонаправлены и показывают, что один класс зависит от определений, сделанных в другом. Зависимости изображают в виде стрелки, проведенной пунктирной линией (рисунке 2.11).



Рисунок 2.11 –Зависимость

При генерации кода для этих классов к ним не будут добав­ляться новые атрибуты. Однако будут созданы специфические для языка операторы, необходимые для поддержки связи. Например, на языке C++ в код войдут необходимые операторы #include.

*Агрегации*(aggregations) представляют собой более тесную форму ассоциации. Агрегация – это связь между целым и его ча­стью. Например, у вас может быть класс Автомобиль, а также классы Двигатель, Покрышки и классы для других частей авто­мобиля. В результате объект класса Автомобиль будет состоять из объекта класса Двигатель, четырех объектов Покрышки и т.д. Агрегации визуализируют в виде линии с ромбиком у класса, яв­ляющегося целым (рисунок 2.12).



Рисунок 2.12 –Агрегация

В дополнение к простой агрегации UML вводит более силь­ную разновидность агрегации, называемую *композицией*. Соглас­но композиции объект-часть может принадлежать только един­ственному целому, и, кроме того, зачастую жизненный цикл час­тей совпадает с циклом целого: они живут и умирают вместе с ним. Любое удаление целого распространяется на его части.

Такое каскадное удаление нередко рассматривается как часть определения агрегации, однако оно всегда подразумевается в том случае, когда множественность роли составляет 1.. 1; например, если необходимо удалить Клиента, то это удаление должно распрост­раниться и на Заказы (и, в свою очередь, на Строки\_заказа).

*Обобщения*(generalization) показывают связи наследования между двумя классами. Большинство объектно-ориентированных языков непосредственно поддерживают концепцию наследова­ния. Она позволяет одному классу наследовать все атрибуты, операции и связи другого. На языке UML связи наследования называют обобщениями и изображают в виде стрелок от класса-потомка к классу-предку (рисунок 2.13).

Помимо наследуемых атрибутов, операций и связей каждый подкласс, как правило, имеет свои собствен­ные уникальные атрибуты, операции и связи.

*Множественность*(multiplicity) показывает, сколько экземп­ляров одного класса взаимодействуют с помощью этой связи с одним экземпляром другого класса в данный момент времени.

Например, при разработке системы регистрации курсов в уни­верситете можно определить классы Course (Курс) и Student (Сту­дент). Между ними установлена связь: у курсов могут быть сту­денты, а у студентов – курсы. Вопрос, на который должен отве­тить параметр множественности: «Сколько курсов студент может посещать в данный момент?» и «Сколько студентов может за раз посещать один курс?».

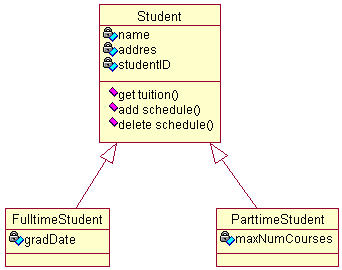


Рисунок 2.13 – Обобщение

Так как множественность дает ответ на оба эти вопроса, ее индикаторы устанавливаются на обоих концах линии связи. Если в примере регистрации курсов один студент мо­жет посещать от 0 до 4 курсов, а один курс могут слушать от 10 до 20 студентов, то на диаграмме классов это можно изобразить следующим образом (рисунок 2.14).

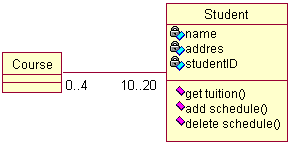


Рисунок 2.14 –Множественность

Для обозначения множественности приняты следующие нотации (таблица 2.4).

Таблица 2.4 – Значения множественности

|  |  |
| --- | --- |
| Множественность | Значение |
| \* | Много |
| 0 | Нуль |
| 1 | Один |
| 0..\* | Нуль или больше |
| 1..\* | Один или больше |
| 0..1 | Нуль или один |
| 1..1 | Ровно один |

Связи можно уточнить с помощью имен связей или ролевых имен. Имя связи – это обычно глагол или глагольная фраза, описывающая, зачем она нужна. Например, между классом Person (человек) и классом Company (компания) может существо­вать ассоциация. В связи с этим возникает вопрос, является ли объект класса Person клиентом компании, ее сотрудником или вла­дельцем? Для того чтобы определить это, ассоциацию можно на­звать «employs» (нанимает). На рисунке 2.15 приведено имя связи.



Рисунок 2.15 –Имя связи

Имена у связей определять необязательно. Обычно это дела­ют, если причина создания связи не очевидна. Имя показывают около линии соответствующей связи.

Ролевые имена применяют в связях ассоциации или агре­гации вместо имен для описания того, зачем эти связи нужны. Воз­вращаясь к примеру с классами Person и Company, можно сказать, что класс Person играет роль сотрудника класса Company. Роле­вые имена - это обычно имена существительные или включающие их фразы, их показывают на диаграмме рядом с классом, играю­щим соответствующую роль. Как правило, пользуются или роле­вым именем, или именем связи, но не обоими сразу. Как и имена связей, ролевые имена не обязательны, их дают, только если цель связи не очевидна. Пример ролевых имен приводится на рисунке 2.16.



Рисунок 2.16 – Ролевые имена

Так же в этом пункте пояснительной записки приводится и описывается последовательность действий по созданию диаграммы классов для выбранного прецедента информационной подсистемы.

**2.5. Проектирование диаграммы состояний для объектов предметной области**

В этом пункте приводится и описывается последовательность действий по созданию диаграммы состояний для классов и диаграммы компонентов для выбранного прецедента информационной подсистемы.

***Диаграммы состояний*** определяют все возможные состояния, в которых может находиться конкретный объект, а также про­цесс смены состояний объекта в результате наступления некото­рых событий. Существует много форм диаграмм состояний, не­значительно отличающихся друг от друга семантикой.

На рисунке 2.17 приводится пример диаграммы состояний для банковского счета (account). Можно также наблюдать процесс перехода счета из одного состояния в другое. Например, если клиент требует закрыть счет, он переходит в состояние «закрыт». Требование клиента называется событием (event), именно такие события и вызывают переход из одного состояния в другое.

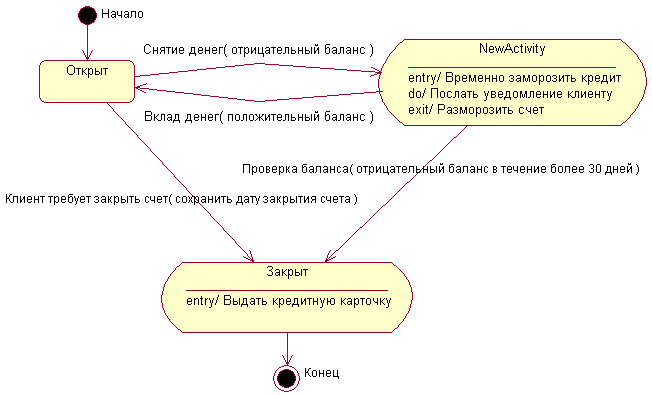


Рисунок 2.17 – Диаграмма состояний для класса Account

Если клиент снимает деньги со счета, он может перейти в со­стояние «Превышение кредита». Это происходит только в том слу­чае, если баланс по счету меньше нуля, что отражено условием [от­рицательный баланс] на нашей диаграмме. Заключенное в квад­ратных скобках условие (guardcondition) определяет, когда может произойти переход из одного состояния в другое.

На диаграмме имеются два специальных состояния - началь­ное (start) и конечное (stop). Начальное состояние выделено чер­ной точкой, оно соответствует состоянию объекта, когда он толь­ко что был создан. Конечное состояние обозначается черной точ­кой в белом кружке, оно соответствует состоянию объекта непосредственно перед его уничтожением.

На диаграмме состоя­ний может быть одно и только одно начальное состояние. В то же время может быть столько конечных состояний, сколько вам нужно, или их может не быть вообще. Когда, объект находится в каком-то конкретном состоянии, могут выполняться различные процессы. В нашем примере при превышении кредита клиенту посылается соответствующее сообщение. Процессы, происходя­щие в тот момент, когда объект находится в определенном со­стоянии, называются действиями (actions).

С состоянием можно связывать следующие данные: деятель­ность, входное действие, выходное действие и событие. Рассмот­рим каждый из них в контексте диаграммы состояний для класса Account банковской системы.

Деятельность(activity) – это поведение, реализуемое объек­том, пока он находится в данном состоянии. Например, когда счет находится в состоянии «Закрыт», происходит возврат кре­дитной карточки пользователю. 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; ее обозначению должно предшество­вать слово do (делать) и слэш.

Входное действие(entryaction) – это поведение, которое вы­полняется, когда объект переходит в данное состояние. Как только счет в банке переходит в состояние «Превышен счет» (см.рисунок 2.17), выполняется действие «Временно заморозить счет» независимо от того, откуда объект перешел в это состояние. Таким образом, данное действие осуществляется не после того, как объект пере­шел в это состояние, а скорее как часть этого перехода. В отли­чие от деятельности входное действие рассматривается как не­прерываемое.

Входное действие также показывают внутри состояния, его обозначению предшествуют слово entry (вход) и косая черта.

Выходное действие(exitaction) подобно входному действию. Однако оно осуществляется как составная часть процесса выхода из данного состояния. В нашем примере при выходе объекта account из со­стояния «Превышен счет» выполняется действие «Разморозить счет». Оно является частью процесса такого перехода. Как и вход­ное, выходное действие является непрерываемым.

Выходное действие изображают внутри состояния, его опи­санию предшествуют слово exit (выход) и косая черта.

Поведение объекта во время деятельности, при входных и выходных действиях, может включать отправку события друго­му объекту. Например, объект account (счет) может посылать со­бытие объекту cardreader (устройство чтения карты). В этом слу­чае описанию деятельности, входного или выходного действия предшествует знак «^». Соответствующая строка на диаграмме выглядит как

**Do: ^Цель. Событие (Аргументы)**

Здесь Цель – это объект, получающий событие, Событие – это посылаемое сообщение, а Аргументы являются параметрами по­сылаемого сообщения.

Деятельность может также выполняться в результате получе­ния объектом некоторого события. Например, объект Account может быть в состоянии «Открыто». При получении некоторого события выполняется определенная деятельность.

Переходом(transition) называется перемещение объекта из одного состояния в другое. На диаграмме все переходы изобра­жают в виде стрелки, начинающейся на первоначальном состоя­нии и заканчивающейся последующим.

Переходы могут быть рефлексивными. Объект может перей­ти в то же состояние, в котором он внастоящий момент находит­ся. Рефлексивные переходы изображают в виде стрелки, начина­ющейся и завершающейся на одном и том же состоянии.

У перехода существует несколько спецификаций. Они вклю­чают события, аргументы, ограждающие условия, действия и посылаемые события. Рассмотрим каждое из них в контексте при­мера банковской системы.

*Событие*(event) - это то, что вызывает переход из одного со­стояния в другое. В нашем примере событие «Клиент требует зак­рыть» вызывает переход счета из открытого в закрытое состоя­ние. Событие размещают на диаграмме вдоль линии перехода.

На диаграмме для отображения события можно использовать как имя операции, так и обычную фразу. В нашем примере собы­тия описаны обычными фразами. Если нужно использовать опе­рации, то событие «Клиент требует закрыть» можно было бы назвать RequestClosure ( ).

У событий могут быть *аргументы*. Так, событие «Сделать вклад», вызывающее переход счета из состояния «Превышен счет» в состояние «Открыто», может иметь аргумент Amount (Количе­ство), описывающий сумму депозита.

Большинство переходов должны иметь события, так как имен­но они, прежде всего, заставляют переход осуществиться. Тем не менее, бывают и автоматические переходы, не имеющие событий. При этом объект сам перемещается из одного состояния в другое со скоростью, позволяющей осуществиться входным действиям, деятельности и выходным действиям.

*Ограждающие условия* (guardconditions) определяют, когда переход может или не может осуществиться. В нашем примере событие «Сделать вклад» переведет счет из состояния «Превы­шение счета» в состояние «Открыто», но только при условии, если баланс будет больше нуля. В противном случае переход не осу­ществится.

Ограждающие условия изображают на диаграмме вдоль ли­нии перехода после имени события, заключая их в квадратные скобки.

Ограждающие условия задавать необязательно. Однако если существует несколько автоматических переходов из состояния, необходимо определить для них взаимно исключающие ограж­дающие условия. Это поможет читателю диаграммы понять, ка­кой путь перехода будет автоматически выбран.

*Действие* (action), как уже говорилось, является непрерывае­мым поведением, осуществляющимся как часть перехода. Входные и выходные действия показывают внутри состояний, посколь­ку они определяют, что происходит, когда объект входит или выходит из состояния. Большую часть действий, однако, изобра­жают вдоль линии перехода, так как они не должны осуществ­ляться при входе или выходе из состояния.

Например, при переходе счета из открытого в закрытое со­стояние выполняется действие «Сохранить дату закрытия счета». Это непрерываемое поведение осуществляется только во время перехода из состояния «Открыто» в состояние «Закрыто».

Действие рисуют вдоль линии перехода после имени собы­тия, его изображению предшествует косая черта.

Событие или действие может быть поведением внутри объек­та, а может представлять собой сообщение, посылаемое другому объекту. Если событие или действие посылается другому объек­ту, перед ним на диаграмме помещают знак «^».

Диаграммы состояний не надо создавать для каждого класса, они применяются только в сложных случаях. Если объект класса может существовать в нескольких состояниях и в каждом из них ведет себя по-разному, для него может потребоваться диаграмма состояний.

**2.6. Проектирование диаграмм компонентов для объектов, входящих в предметную область**

***Диаграммы компонентов*** показывают, как выглядит модель на физическом уровне. На них изображены компоненты программ­ного обеспечения и связи между ними. При этом на такой диаг­рамме выделяют два типа компонентов: исполняемые компонен­ты и библиотеки кода.

Каждый класс модели (или подсистема) преобразуется в ком­понент исходного кода. После создания они сразу добавляются к диаграмме компонентов. Между отдельными компонентами изоб­ражают зависимости, соответствующие зависимостям на этапе компиляции или выполнения программы.

На рисунке 2.18 изображена одна из диаграмм компонентов для банковской системы.

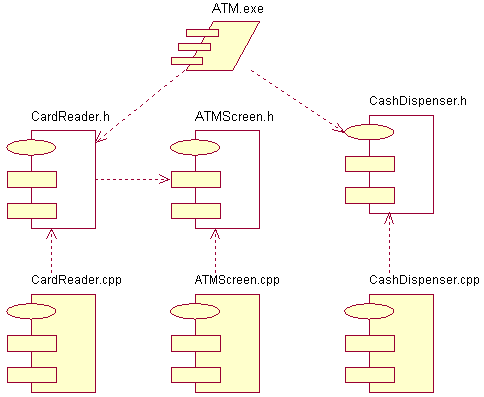


Рисунок 2.18 –Диаграмма компонентов для клиентской части системы

На этой диаграмме показаны компоненты для клиентской час­ти системы. В данном случае система разрабатывается на языке C++. У каждого класса имеется свой собственный заголовочный файл и файл с расширением .СРР, так что каждый класс преобразуется в свои собственные компоненты на диаграмме. Например, класс ATMScreen преобразуется в компонент ATMScreen диаграммы. Он пре­образуется также и во второй компонент ATMScreen. Вместе эти два компонента представляют тело и заголовок класса ATMScreen. Выделенный (залитый) компонент называется спецификацией пакета (packagespecification) и соответствует файлу тела класса ATMScreen на языке C++ (файл с расширением \*.cpp). Невыделенный компо­нент (не залитый) также называется спецификацией пакета, но соответствует за­головочному файлу класса языка C++ (файл с расширением \*.h).

Компонент АТМ.ехе является спецификацией задачи и представля­ет поток обработки информации (threadofprocessing). В данном случае поток обработки является исполняемой программой.

Компоненты соединены штриховой линией, что соответству­ет зависимостям между ними. Например, класс CardReader зави­сит от класса ATMScreen. Это означает, что для того, чтобы класс CardReader мог быть скомпилирован, класс ATMScreen должен уже существовать. После компиляции всех классов может быть создан исполняемый файл ATMClient.exe.

Пример банковской системы содержит два потока обработ­ки, и таким образом получаются два исполняемых файла. Один из них − это клиентская часть системы, она содержит компонен­ты CashDispenser, CardReader и ATMScreen. Второй файл - это сервер, включающий в себя компонент Account. Диаграмма ком­понентов для сервера показана на рисунке 2.19.

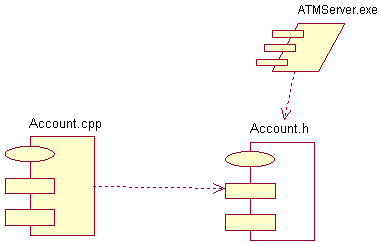


Рисунок 2.19 – Диаграмма компонентов для сервера

Как видно из примера, у системы может быть несколько ди­аграмм компонентов в зависимости от числа подсистем или ис­полняемых файлов. Каждая подсистема является пакетом ком­понентов. В общем случае пакеты − это совокупности компонен­тов. Пример банковской системы содержит два пакета: клиентская часть и сервер.

Диаграммы компонентов применяются теми участниками проекта, кто отвечает за компиляцию системы. Из нее видно, в каком порядке надо компилировать компоненты, а также какие исполняемые компоненты будут созданы системой. На такой ди­аграмме показано соответствие классов реализованным компо­нентам. Она нужна там, где начинается генерация кода.

**2.7. Разработка диаграмм размещения для объектов информационной системы**

В этом пункте приводится и описывается последовательность действий по созданию диаграммы размещения информационной.

Диаграмма размещенияотражает физические взаимосвязи меж­ду программными и аппаратными компонентами системы. Она по­казывает размещение объектов и компонентов в распределенной системе.

Каждый узел на диаграмме размещения представляет собой некоторый тип вычислительного устройства - в большинстве слу­чаев часть аппаратуры. Эта аппаратура может быть простым ус­тройством или датчиком, а может быть и мэйнфреймом.

Диаграмма размещения показывает физическое расположе­ние сети и местонахождение в ней различных компонентов. В нашем примере банковская система состоит из большого коли­чества подсистем, выполняемых на отдельных физических уст­ройствах, или узлах (node). Диаграмма размещения для банковс­кой системы показана на рисунке 2.20.Из данной диаграммы можно узнать о физическом размеще­нии системы. Клиентские программы будут работать в несколь­ких местах на различных сайтах. Через закрытые сети будет осу­ществляться их сообщение с региональным сервером системы, с работающим программным обеспечением. В свою очередь, ре­гиональный сервер посредством локальной сети будет сообщать­ся с сервером банковской базы данных, работающим под управ­лением Oracle. Наконец, с региональным сервером соединен принтер.

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

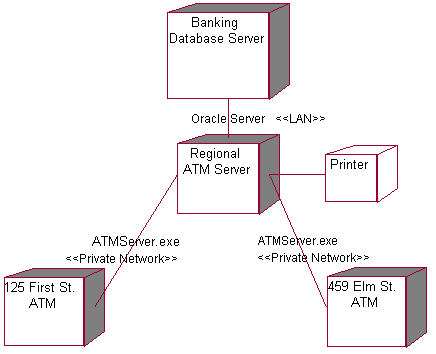


Рисунок 2.20 –Диаграмма размещения для банковской системы

**2.8. Создание прототипа информационной системы**

Данный пункт должен описывать последовательность разработки интерфейса ИС, содержать подробное описание наиболее существенных его особенностей.

Необходимо описать технологию работы пользователя – последовательность заполнения справочников, документов, вид получаемых отчетов. Представить созданные объекты, заполненные данными, близкими реальным, в виде скриншотов.

**2.8.1. Требования к информационной системе**

В данном пункте должны быть перечислены функциональные требования к информационной системе из технического задания и дана ссылка **на приложение (например, Приложение А)**, в котором приводится полный текст Технического задания. ТЗ должно быть подготовлено в соответствии с [ГОСТ 34.602-89](http://www.nist.ru/hr/doc/gost/34-602-89.htm).

**2.8.2. Технология создания прототипа информационной системы**

Данный раздел должен описывать последовательность разработки интерфейса **в конкретной программной среде**, содержать подробное описание наиболее существенных его особенностей.

Для каждого объекта хранимых данных (справочников, документов, перечислений, регистров, констант) представить, фрагменты конфигурации (скриншоты), модули.

**2.8.3. Инструкция пользователю**

Описать технологию работы пользователя – последовательность заполнения справочников, документов, вид получаемых отчетов. Представить созданные объекты, заполненные данными, близкими реальным, в виде скриншотов.

**Выводы**

Выводы должны быть конкретными, то есть вытекать из содержания раздела. Выводы описательного характера недопустимы, например, « ….В этом разделе мы рассмотрели….».

***Пример создания прототипа информационной системы***

Действующее лицо Менеджер по работе с клиентами подключается к системе и вводит свое имя и пароль (рис. 2.21).

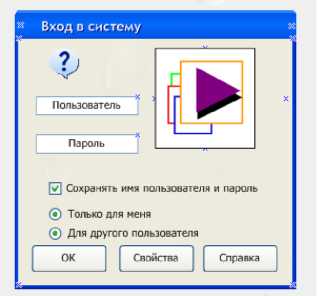


Рисунок 2.21 – Главное окно информационной системы

После ввода имени пользователя и пароля система проверяет правильность пароля и выводит возможные варианты действий: добавить (Add), изменить (Change), удалить (Delete), просмотреть (View), выйти *(Exit)*(рис. 2.22)*.*



Рисунок 2.22 – Окно «Выбор варианта действий»

Если выбрана операция добавить (Add), выполняется поток добавить ново­го клиента (Add a NewClient). Система отображает диалоговое окно, содержа­щее поля для ввода данных о новом клиенте, пользователь заполняет поля вво­да, система запоминает введенные данные. Затем прецедент начинается сначала (рис. 2.23).



Рисунок 2.23 – Диалоговые окна пользовательского интерфейса

Если выбрана операция изменить (Change) выполняется поток*Изменить данные о клиенте* (ChangeClientData). Система отображает диалоговое окно, содержащее список клиентов и поле для ввода номера клиента (или поля для данных клиента), *Менеджер по работе с клиентами* выбирает необходимого клиента из списка или вводит его номер в поле (или его данные), система ото­бражает информацию о данном клиенте, после изменений система запоминает введенные данные. Затем прецедент начинается сначала (рис. 2.24).

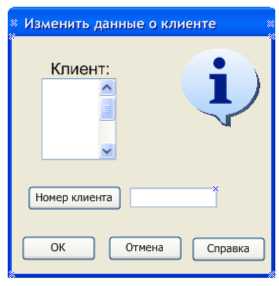


Рисунок 2.24 – Диалоговое окно «Изменить данные о клиенте»

Если выбрана операция удалить (Delete), выполняется поток удалить клиента (DeleteClient). Система отображает диалоговое окно, содержащее список кли­ентов и поле для ввода номера клиента. Менеджер выбирает необходимого клиента из списка или вводит его номер в поле. Система удаляет выбранного клиента. Затем прецедент начинается сначала (рис. 2.25)

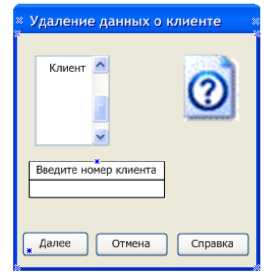


Рисунок 2.25 – Диалоговое окно «Удаление данных о клиенте»

Если выбрана операция просмотреть (View), выполняется поток просмотреть данные о клиенте (ViewClientData). Система отображает диалоговое окно, со­держащее список клиентов и поле для ввода номера клиента (или поля ввода). Менеджер выбирает необходимого клиента из списка или вводит его номер в поле ввода. Система отображает информацию о выбранном клиенте. Когда ме­неджер просмотрит информацию, прецедент начнется сначала (рис. 2.26).

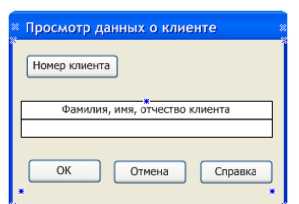


Рисунок 2.26 – Диалоговое окно «Просмотр данных о клиенте»

Если выбрана операция выйти (Exit) прецедент завершается.

**Выводы по практической части**

Выводы должны быть конкретными, то есть вытекать из содержания раздела. Выводы описательного характера недопустимы, например, « ….В этом разделе мы рассмотрели….».

**Заключение**

В заключении к курсовому проекту необходимо сформулировать выводы о проделанной работе.

Заключение должно содержать общие выводы, сделанные по результатам проведенного исследования. В заключение необходимо проанализировать проделанную работу, изложить в порядке проведения исследования промежуточные практические и теоретические результаты и выводы, обобщить их и сформулировать общий вывод по всей работе, оценив ее успешность, показать общий вывод в контексте складывающихся перспектив дальнейшего изучения, охарактеризовать его научную значимость и возможность практического применения. Желательно оценить не только главные итоги работы, но и побочные, второстепенные результаты, которые могут также обладать самостоятельным научным значением. Следует указать пути и перспективы дальнейшего исследования темы, обрисовать задачи, которые еще предстоит решить. Текст заключения должен быть написан так, чтобы **выводы соотносились с поставленными во введении целью и задачами исследования.** Основные выводы лучше изложить в форме пронумерованных тезисов, в каждом из которых выделить и обосновать один конкретный вывод. Формулировки всех выводов должны быть предельно четкими, ясными, краткими и логически безупречными; давать полное представление о содержании, значимости, обоснованности и эффективности разработок.

Таким образом, заключение представляет собой окончательный, итоговый синтез всего ценного и значимого, существенного и нового, что содержится в КП.

Примерный объем заключения 1 – 2 страницы.

**Список использованных источников**

**Список** *(не менее 20 источников*) демонстрирует степень осведомленности студента в литературе по теме КП. В список источников включают все использованные при выполнении КП источники: правительственные постановления, отраслевые приказы и инструкции, учебники и учебные пособия, методические указания, справочники, статьи в периодических изданиях, статьи из сборников трудов, отчеты по научно-исследовательской работе, инструктивную документацию по использованию программных средств, электронные адреса использованных Internet-ресурсов. Ссылки на источники даются в квадратных скобках в самом тексте КП. При цитировании или пересказе авторского текста с указанием номера или диапазона страниц ([12, с. 29]), когда не применяется цитирование – без указания страниц ([12]). Нумерация ссылок в статье сквозная.

Список дают в алфавитном порядке в соответствие с ГОСТ 7.1–2003 [11]. Общая схема библиографической записи выглядит следующим образом:

1. Авторы (фамилия, имя, отчество индивидуального автора (авторов); наименование коллективного автора).

2. Название (Название и сведения, относящиеся к заглавию (раскрывают тематику, вид, назначение документа и т.д.)).

3. Сведения об ответственности (содержат информацию о составителях, редакторах, переводчиках и т.п., об организациях, от имени которых опубликован документ).

4. Сведения об издании (содержат данные о повторности издания, его переработках и т.п.).

5. Место издания Издательство или издающая организация, дата издания.

6. Объем (сведения о количестве страниц, листов).

**Примеры библиографического описания**

**I. Описание книг**

**1.** **Книги одного, двух или трех авторов** описываются под фамилией первого автора:

* **книга одного автора**:

       Чалдаева**,** Л. А. Экономика предприятия : учебник для бакалавров / Л. А. Чалдаева.— 3-е изд., перераб. и доп.— М.: Юрайт, 2013.— 411 с.

* **книга двух авторов**:

       Нехаев, Г. А. Металлические конструкции в примерах и задачах: учеб. пособие  / Г. А. Нехаев, И. А. Захарова.— М.: Изд-во Ассоциации строительных вузов, 2010.— 144 с.

* **книга трех авторов**:

    Акимов, А. П. Работа колес: монография / А. П. Акимов, В. И. Медведев, В. В. Чегулов.— Чебоксары: ЧПИ (ф) МГОУ, 2011.— 168 с.

**2.** **Книги четырех и более авторов** указываются под заглавием (названием) книги. После названия книги, за косой чертой пишется фамилия одного автора и вместо следующих фамилий слово — [и др.].

      Информационно-измерительная техника и электроника  : учебник / Г. Г. Раннев [и др.];под  ред. Г. Г. Раннева.— 3-е изд., стереотип.— М.: Академия, 2009.— 512 с.

**3. Книги с коллективом авторов, или в которых не указан автор**, указываются под заглавием (названием) книги. За косой чертой пишется фамилия редактора, составителя или другого ответственного лица.

Анализ и диагностика финансово-хозяйственной деятельности предприятия: учебник / под ред. В. Я. Позднякова.— М.: Инфра-М, 2010.— 617 с.

**II. Описание статьи из журнала**

 При описании статей из журналов указываются автор статьи, ее название, затем, за двумя косыми чертами указывают название журнала, в котором она опубликована, год, номер, страницы, на которых помещена статья.

* **статья одного автора**:

Леденева, Г. Л. К вопросу об эволюции в архитектурном творчестве / Г. Л. Леденева  // Промышленное и гражданское строительство.— 2009.— № 3.— С. 31–33.

* **статья двух авторов**:

Шитов, В. Н. Комплексный подход к анализу конкурентоспособности предприятия [Текст] / В. Н. Шитов, О. Ф. Цымбалист // Экономический анализ: теория и практика.— 2014.— № 13. -  С. 59–63.

* **статья трех авторов**:

Зацепин, П. М. Комплексная безопасность потребителей эксплуатационных характеристик строений / П. М. Зацепин, Н. Н. Теодорович, А. И. Мохов // Промышленное и гражданское строительство. – 2009.— № 3.— С. 42.

* **статья четырех и более авторов**:

Опыт применения специальных технологий производства работ по устройству ограждающих конструкций котлованов / С. С. Зуев [и др.] // Промышленное и гражданское строителсьво.— 2009.— № 3.— С. 49-50.

**III. Описание статьи из книг и сборников**

* **статья из книги одного автора**:

Каратуев, А. Г. Цели финансового менеджмента / А. Г. Каратуев // Финансовый менеджмент: учебно-справочное пособие / А. Г. Каратуев.— М., 2001.— С. 207–451.

* **статья из книги двух авторов**:

Безуглов,  А.  А.  Президент Российской Федерации   /   А. А. Безуглов // Безуглов А. А. Конституционное право России: учебник для юридических вузов (полный курс): в 3-х т. / А. А. Безуглов, С. А. Солдатов.— М., 2001.— Т. 1.— С. 137–370.

* **статья из книги трех и более авторов**:

Григорьев В. В. Торги: разработка документации: методы проведения / В. В. Григорьев // Григорьев В. В. Управление муниципальной недвижимостью: учебно-практическое пособие / В. А. Григорьев, М. А. Батурин, Л. И. Мишарин.— М., 2001.— С. 399–404.

Маркетинговая программа в автомобилестроении (ОАО “АвтоВАЗ”) // Российский маркетинг на пороге третьего тысячелетия: практика крупнейших компаний / А. А. Браверман [и др.]; под ред. А. А. Бравермана.— М., 2001.— Гл. 4.— С. 195–272: табл.

* **статья из сборника научных трудов**:

 Данилова, Н. Е. Моделирование процессов в следящем приводе с исполнительным двигателем постоянного тока при независимом возбуждении  / Н. Е. Данилова, С. Н. Ниссенбаум // Инновации в образовательном процессе: сб. тр. науч.-практич. конф.— Чебоксары: ЧПИ (ф) МГОУ, 2013.— Вып. 11. -  С. 158–160.

**IV. Описание официальных изданий**

Конституция Российской Федерации: принята всенародным голосованием 12 декабря 1993 года.— М.: Эксмо, 2013.— 63 с.

Уголовный кодекс Российской Федерации. Официальный текст: текст Кодекса приводится по состоянию на 23 сентября 2013 г.— М.: Омега-Л, 2013.— 193 с.

          О проведении в Российской Федерации года молодежи  : указ Президента Российской Федерации от 18.09.2008 г. № 1383 // Вестник образования России.— 2008.— № 20 (окт.). -  С. 13–14.

**V. Описание нормативно-технических и технических документов**

ГОСТ Р 517721–2001. Аппаратура радиоэлектронная бытовая. Входные и выходные параметры и типы соединений. Технические требования.  — Введ. 2002-01-01.— М.: Изд-во стандартов, 2001.— 27 с.

или

Аппаратура радиоэлектронная бытовая. Входные и выходные параметры и типы  соединения. Технические требования: ГОСТ Р 517721–2001. -Введ. 2002-01-01.— М.: Изд-во стандартов, 2001.— 27 с.

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

**VI. Описание электронных ресурсов**

* **диск**

Даль, В. И. Толковый словарь живого великого языка Владимира Даля [Электронный ресурс] / В. И. Даль; подгот. по 2-му  печ. изд. 1880–1882 гг. – Электрон. дан. – М.: АСТ, 1998. – 1 электрон. опт. диск (CD-ROM).

* **электронный журнал**

        Краснов, И. С. Методологические аспекты здорового образа жизни россиян [Электронный ресурс] / И. С. Краснов // Физическая культура: науч.-метод. журн. – 2013.— № 2. – Режим доступа: http://sportedu.ru. – (Дата обращения: 05.02.2018).

* **сайт**

Защита персональных данных пользователей и сотрудников библиотеки [Электронный ресурс]. – Режим доступа: http://www.nbrkomi.ru. – Заглавие с экрана. – (Дата обращения: 14.04.2018).

**Приложения**

В приложения выносятся техническое задание и диаграммы, спроектированные в ходе выполнения КП.

В приложения следует включать вспомогательный материал, который загромождает текст основной части и затрудняет его восприятие. К вспомогательному материалу относятся промежуточные расчеты, таблицы вспомогательных цифровых данных, инструкции, методики, компьютерные распечатки, иллюстрации вспомогательного характера, заполненные формы отчетности и другие документы. Приложения должны быть расположены в порядке появления ссылок в тексте.

Каждое приложение следует начинать с новой страницы с указанием наверху справа страницы слова "ПРИЛОЖЕНИЕ" и его обозначения (номера) шрифтом Times New Roman, 16пт, обычный, жирный. Приложение может иметь содержательный заголовок, который записывается симметрично относительно текста прописными буквами шрифтом Times New Roman 14 пт, обычный, жирный. Приложения оформляют как продолжение объема курсового проекта на последующих страницах и помещают после списка использованных источников. Нумерация приложений. Если в работе более одного приложения, их необходимо пронумеровать. Приложения обозначают арабскими цифрами. Все приложения должны быть перечислены в содержании. Текст каждого приложения при необходимости может быть разделен на пункты. Рисунки, таблицы и формулы, помещаемые в приложении, нумеруются арабскими цифрами в пределах каждого приложения: «Табл. П. 1.1» (первая таблица первого приложения). Страницы, на которых размещены приложения, включают в общую нумерацию страниц КП.

## 4. Общие требования к курсовому проекту, требования к оформлению

Для выполнения курсовой работы студент должен руководствоваться следующими документами, представленными на сайте:

1. Инструкция по организации курсового проектирования (СМКО МИРЭА 7.5.1/04.И.05-18).

2. Рекомендации по оформлению письменных работ обучающихся по образовательным программам бакалавриата, специалитета и магистратуры (СМКО МИРЭА 7.5.1/03.П.69-16).

Данные документы представлены на сайте университета.

Кроме того, при выполнении работы необходимо выполнять следующие требования:

1. Поля: левое – 30 мм, правое – 10 мм, верхнее и нижнее – 20 мм;

абзацный отступ – 1,25 мм; межстрочный интервал – полуторный.

2. Автоматическая расстановка переносов.

3.Названияненумерованных разделов:Содержание, Введение, Заключение, Список использованных источников – выполнять с выравниванием по центру.

4. Оформление разделов, подразделов– выравнивание по ширине с отступом 1,25 без дополнительных строк**:**

1. Название раздела

1.1. Название подраздела

Текст первого раздела.

5. Оформление таблиц:

Таблица 2.1 – Название таблицы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |

Продолжение текста работы…

Продолжение таблицы 2.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |

6. Оформление рисунков:

Нарисовали рисунок

Рисунок 1.1 – Название рисунка по центру

После рисунка до текста – 1 интервал.

7. Оформление перечислений:

Абзацный отступ 1,25 см, «номер - буква – дефис».

8. На все источники, приведенные в списке литературы, должны быть ссылки в тексте.

Также должны быть все подписи (руководителя, зав. кафедрой, студента), проставлены все даты, отзыв руководителя (Приложение 3), заполненное задание.

Все листы курсовой работы должны быть пронумерованы, а сама работа сшита. Нумерация страниц должна быть сквозной: первой страницей является титульный лист. Однако номер страниц проставляется со второго листа *Введения* (в общую нумерацию включаются титульный лист, лист задания, аннотация, содержание, первый лист введения).

Каждый раздел и подпункт в тексте должны иметь заголовок в точном соответствии с наименованием в содержании. Точка в конце заголовка не ставится. Новый раздел следует начинать с новой страницы. Новый подраздел (подпункт) можно начинать на той же странице, на которой закончился предыдущий, если на этой странице кроме заголовка может поместиться несколько строчек текста. В противном случае новый подраздел следует начинать со следующей страницы.

При использовании в тексте таблиц их следует помещать по ходу изложения материала после ссылки на них. Таблицы нумеруются в сквозном порядке в пределах раздела (двойным номером, например, 2.2 – номер второй таблицы во втором разделе). Слово «Таблица 2.2 – …» и ее номер указываются по одной строке с заголовком таблицы. Заголовок таблицы пишется с заглавной буквы, точка в конце не ставится. Таблицы с большим количеством строк допускается переносить на другую страницу. При переносе части таблицы на следующую страницу над ней помещают слова «Продолжение таблицы…» с указанием номера и полностью переносится шапка таблицы. Заголовок таблицы помещают только над ее первой частью.

## 5. Рекомендации по организации работы над проектом, примерный календарный план его выполнения

Выполнение основных разделов проекта рекомендуется выполнять по следующему календарному плану (таблица 1).

Таблица 1 – Календарный график выполнения курсового проекта

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Этапы работы | | Недели | | | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 1 | Формулировка темы и получение задания |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | Работа с литературой |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | Разработка моделей бизнес-процессов |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 | Написание текста работы |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 | Оформление работы |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6 | Проверка. Сдача на кафедру |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7 | Нормоконтроль |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8 | Защита |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

На этапе формулировки темы и получения задания на курсовой проект: тема и задание курсового проекта выбираются студентом из списка, утвержденного на заседании кафедры, согласуются с руководителем и утверждаются заведующим кафедрой.

На этапе работы с литературой производится изучение теоретического материала. Руководитель рекомендует список литературы, с которым студенту необходимо ознакомиться перед выполнением основной части работы. Студент имеет право пополнить предложенный список по своему усмотрению.

На этапе разработкимоделей бизнес-процессовв соответствии с заданием к курсового проекта студент разрабатывает функционал ььные и объектно-ориентированные модели предметной области.

На этапе оформления работынужно предусмотреть, что общий объем пояснительной записки не должен превышать 50 страниц, в том числе введение – не более 3 страниц. Записка иллюстрируется схемами, скриншотами экранных форм, диалоговых окон, выходными документами, текстами программ (можно выносить в приложение).

На этапе проверки и сдачи работы на кафедрупояснительная записка сдается на проверку руководителю работы в срок не менее чем за 10 дней до дня защиты. По итогам проверки руководитель либо допускает студента к защите, либо возвращает пояснительную записку на доработку.

На этапе защиты курсового проекта **с**тудент делает доклад (около 5 минут), в котором кратко излагает результаты работы.

## 6. Порядок защиты и ответственность студента за выполнение задания по курсовому проекту

Законченный и надлежащим образом оформленный курсовой проект представляется **не позднее установленного срока на кафедру.** Дата представ­ления курсового проекта регистрируется лаборантом кафедры в специальном журнале и фиксируется на титульном листе работы.

Руководитель проекта подготавливает отзыв (Приложение 3).

Проверенный научным руководителем и допущенный к защите КП возвращается студенту для подготовки ее к защите. Автор имеет право доработать, исправить представленный материал, добавить новые страницы к проверенному тексту. Этот материал добавляется к КП; пометки, вопросы и замечания руководителя, как и весь старый текст не могут быть изъяты из работы.

Защита – является важным и завершающим этапом выполнения курсового проекта.

Порядок защиты курсового проекта и состав комиссии утверждается на заседании кафедры. Студент делает доклад (5-7 минут), в котором кратко излагает результаты проектирования, демонстрирует графическую часть проекта и разработанную программную реализацию. Оценка КП выставляется комиссией и является дифференцированной.

## 7. Список рекомендуемой литературы

***Основная***

1. Коваленко В.В. Проектирование информационных систем : учеб. пособие / В.В. Коваленко. – М.: ФОРУМ: ИНФРА-М, 2018. – 320 с. Режим доступа: http://znanium.com/catalog/product/980117
2. Проектирование информационных систем: Учебное пособие / Заботина Н.Н. - М.: НИЦ ИНФРА-М, 2016. - 331 с. Режим доступа: http://znanium.com/catalog/product/542810

***Дополнительная***

1. Хетагуров Я.А. Проектирование автоматизированных систем обработки информации и управления (АСОИУ) [Электронный ресурс]: учебник / Я.А. Хетагуров. – Эл. изд. – Электрон. текстовые дан. (1 файл pdf : 243 с.). – М.: БИНОМ. Лаборатория знаний, 2015. – (Учебник для высшей школы). Режим доступа: http://znanium.com/catalog.php?bookinfo=539500
2. Разработка и эксплуатация автоматизированных информационных систем: учеб. пособие/ Л.Г. Гагарина. – М.: ИД «ФОРУМ»: ИНФРА-М, 2018. – 384 с. Режим доступа: http://znanium.com/catalog/product/942717

## Приложение 1

|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА**  **филиал РТУ МИРЭА в г. Ставрополе** | | |

Кафедра \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование кафедры)

**КУРСОВАЯ РАБОТА**

по дисциплине \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование дисциплины)

**Тема курсовой работы \_\_\_\_\_**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Студент группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(учебная группа, фамилия, имя, отчество студента) (подпись студента)

**Руководитель курсовой работы \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Ф.И.О., должность, звание, ученая степень (подпись руководителя)

**Рецензент** (при наличии) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

должность, звание, ученая степень подпись рецензента

Работа представлена к защите «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_ г.

Допущен к защите «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_\_ г.

Ставрополь 20\_\_\_

|  |  |  |
| --- | --- | --- |
|  |  | Приложение 2 |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА**  **филиал РТУ МИРЭА в г. Ставрополе** | | |

Кафедра \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование кафедры)

Утверждаю

Заведующий кафедрой \_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

подпись ФИО

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_\_\_ г.

**ЗАДАНИЕ**

**на выполнение курсовой работы** по дисциплине

«\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Группа \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Тема \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Исходные данные:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Перечень вопросов, подлежащих разработке, и обязательного графического материала:\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Срок представления к защите курсовой работы до «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_\_ г.**

**Задание на курсовую работу выдал \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

подпись руководителя (Ф.И.О. руководителя)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_\_ г.

**Задание на курсовую работу получил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

подпись обучающегося (Ф.И.О. обучающегося)

Ставрополь 20\_\_\_

## Приложение 3

**ОТЗЫВ РУКОВОДИТЕЛЯ КУРСОВОЙ РАБОТЫ**

Кафедра \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Автор работы (студент)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Шифр, направление подготовки\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Форма обучения\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Курс, группа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дисциплина \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Тема работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель работы \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ФИО, должность, звание, степень

**Оценка соответствия проекта требованиям**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Показатели /**  **требования** | **Соответствие проекта требованиям** | | | | | |
| Соответствие теме и заданию | Не соответствует |  | Соответствует | | |  |
| Соответствие рекомендациям по оформлению текстов, таблиц, рисунков, ГОСТов и т.д. | Не соответствует |  | Частично соответствует |  | Соответствует |  |
| Полнота разработки поставленных вопросов | Вопросы не разработаны |  | Вопросы разработаны частично |  | Вопросы разработаны полностью |  |
| Качество разработки поставленных вопросов | Вопросы не разработаны |  | Вопросы разработаны частично |  | Вопросы разработаны полностью |  |
| Теоретическая значимость | Не несет значимости |  | Частичная значимость |  | Полноценная |  |
| Практическая значимость | Не несет значимости |  | Частичная значимость |  | Полноценная |  |
| Характеристика работы студента | Не самостоятельная работа |  | Самостоятельная работа | | |  |

|  |  |
| --- | --- |
| Оценка | Критерии выставляемой оценки |
| Отлично, повышенный | Работа соответствует теме и заданию, выполнены рекомендации по оформлению текстов, таблиц, рисунков, ГОСТов и т.д., полнота разработки поставленных вопросов полноценная, качество разработки поставленных вопросов полноценное, теоретическая значимость полноценная, практическая значимость полноценная, выполнена самостоятельно. |
| Хорошо, базовый | Работа соответствует теме и заданию, выполнены рекомендации по оформлению текстов, таблиц, рисунков, ГОСТов и т.д., полнота разработки поставленных вопросов частичная, качество разработки поставленных вопросов полноценное, частичная теоретическая значимость, практическая значимость полноценная, выполнена самостоятельно. |
| Удовлетворительно, базовый | Работа соответствует теме и заданию, частично выполнены рекомендации по оформлению текстов, таблиц, рисунков, ГОСТов и т.д., полнота разработки поставленных вопросов частичная, качество разработки поставленных вопросов частичное, частичная теоретическая значимость, частичная практическая значимость, выполнена самостоятельно. |
| Неудовлетворительно | Работа не соответствует теме и заданию, не выполнены рекомендации по оформлению текстов, таблиц, рисунков, ГОСТов и т.д., не полная и не качественная разработка поставленных вопросов, не несет теоретической и практической значимости, выполнена не самостоятельно. |

Рекомендуемая оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

Подпись ФИО

## Приложение 4

**Варианты тем предметных областей**

1. Проектирование информационной подсистемы автоматизации документооборота предприятия
2. Проектирование информационной подсистемы института
3. Проектирование информационной подсистемы факультета института
4. Проектирование информационной подсистемы кафедры института
5. Проектирование информационной подсистемы торгового предприятия
6. Проектирование информационной подсистемы автотранспортного предприятия
7. Проектирование информационной подсистемы лечебного учреждения (больницы)
8. Проектирование информационной подсистемы лечебного учреждения (поликлиники)
9. Проектирование информационной подсистемы банка.
10. Проектирование информационной подсистемы культурно-спортивного центра
11. Проектирование информационной подсистемы издательства.
12. Проектирование информационной подсистемы предприятия связи
13. Проектирование информационной подсистемы школы (колледжа, гимназии)
14. Проектирование информационной подсистемы выставочного центра
15. Проектирование информационной подсистемы службы занятости
16. Проектирование информационной подсистемы акционерного общества
17. Проектирование информационной подсистемы инвестиционной фирмы
18. Проектирование информационной подсистемы малого предприятия
19. Проектирование информационной подсистемы бухгалтерии малого предприятия
20. Проектирование информационной подсистемы страховой фирмы