МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторных работ

по дисциплине

«Разработка и стандартизация информационных систем»

для студентов направления

09.03.02 «Информационные системы и технологии»

Направленность (профиль) «Прикладное программирование и интернет-технологии»

Ставрополь

2026

Содержание

Введение

Целью дисциплины «Разработка и стандартизация информационных систем» является дать студентам знания об эволюции и современных тенденциях развития стандартизации программных средств. Ознакомить их с теоретическими основами и прикладными методами разработки, анализа, испытаний и внедрения программного обеспечения как необходимой составляющей современных информационных технологий.

Задачами дисциплины «Разработка и стандартизация информационных систем» является формирование набора профессиональных компетенций, а также ввести студентов в проблему стандартизации программного обеспечения, ознакомить с целями стандартизации и сертификации программного обеспечения и роли стандартизации и сертификации в обеспечении качества и конкурентоспособности программных и аппаратных средств.

Дисциплина «Разработка и стандартизация информационных систем» относится к части, формируемой участниками образовательных отношений. Ее освоение происходит в 7 семестре.

Перечень планируемых результатов обучения по дисциплине (модулю), соотнесённых с планируемыми результатами освоения образовательной программы

Код, формулировка компетенции	Код, формулировка индикатора	Планируемые результаты обучения по дисциплине (модулю), характеризующие этапы формирования компетенций, индикаторов
ПК-2 – способен использовать современные инструментальные средства и технологии программирования при разработке прикладного программного обеспечения вычислительных средств и систем различного функционального назначения	ПК-2 ИД-2-1 Проводит классификацию и осуществляет выбор современных инструментальных средств разработки прикладного программного обеспечения вычислительных средств и систем различного функционального назначения ПК-2 ИД-2-2 Демонстрирует знания технологий программирования, разработки прикладного программного обеспечения вычислительных средств и систем различного функционального назначения ПК-2 ИД-2-3 Использует современные инструментальные средства разработки прикладного программного обеспечения вычислительных средств и систем различного функционального назначения ПК-2 ИД-2-4 Применяет технологии	Грамотно проводит классификацию современных инструментальных средств разработки прикладного программного обеспечения вычислительных средств и систем различного функционального назначения Грамотно осуществляет выбор современных инструментальных средств разработки прикладного программного обеспечения вычислительных средств и систем различного функционального назначения Анализирует актуальные технологии программирования Выбирает адекватные методы анализа идеологических и ценностных аспектов профессиональной деятельности Грамотно разрабатывает прикладное программное обеспечение вычислительных средств и систем различного функционального назначения

	программирования при разработке прикладного программного обеспечения вычислительных средств и систем различного функционального назначения	Правильно использует современные инструментальные средства разработки прикладного программного обеспечения вычислительных средств и систем различного функционального назначения Правильно применяет технологии программирования при разработке прикладного программного обеспечения вычислительных средств и систем различного функционального назначения
ПК-4 – способен применять и осваивать новые средства автоматизированного проектирования, разработки, тестирования и сопровождения программного обеспечения	ПК–4$_{ИД-4.1}$ Демонстрирует знания новых средств автоматизированного проектирования ПК–4$_{ИД-4.2}$ Осуществляет выбор средств разработки, тестирования и сопровождения программного обеспечения ПК-4$_{ИД-4.3}$ Применяет новые средства автоматизированного проектирования и разработки программного обеспечения ПК-4$_{ИД-4.4}$ Осваивает новые средства автоматизированного проектирования и разработки программного обеспечения ПК-4$_{ИД-4.5}$ Осуществляет сопровождение программного обеспечения ПК-4$_{ИД-4.6}$ Разрабатывает и использует методики тестирования	Грамотно демонстрирует знания новых средств автоматизированного проектирования Правильно осуществляет выбор средств разработки, тестирования и сопровождения программного обеспечения Применяет новые средства автоматизированного проектирования и разработки программного обеспечения Грамотно осваивает новые средства автоматизированного проектирования и разработки программного обеспечения Анализирует и осуществляет сопровождение программного обеспечения Грамотно разрабатывает и использует методики тестирования
ПК-8 – способен осуществлять планирование и организацию собственной работы; планирование и координация работ по настройке и	ПК-8$_{ИД-8.1}$ Демонстрирует знания методов и технологий планирования и организации собственной работы ПК-8$_{ИД-8.2}$ Планирует работы по настройке и сопровождению программного продукта	Грамотно демонстрирует знания методов и технологий планирования и организации собственной работы Анализирует и планирует работы по настройке и сопровождению программного продукта

сопровождению программного продукта; составление частного технического задания на разработку программного продукта	ПК-8$_{ИД-8.3}$ Координирует работы по настройке и сопровождению программного продукта ПК-8$_{ИД-8.4}$ Имеет представление о методиках разработки частного технического задания ПК-8$_{ИД-8.5}$ Составляет частное техническое задание на разработку программного продукта ПК-8$_{ИД-8.5}$ Осуществляет и организует планирование собственной работы ПК-8$_{ИД-8.6}$ Организует собственную работу	Правильно координирует работы по настройке и сопровождению программного продукта Имеет представление о методиках разработки частного технического задания Грамотно составляет частное техническое задание на разработку программного продукта Осуществляет планирование и организацию собственной работы

Лабораторная работа 1

РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ

Цель и содержание работы: изучить правила написания технического задания (ТЗ) на разработку программного обеспечения; изучить ГОСТ 19.201–78; разработать документ "Техническое задание".

Формируемые компетенции: ПК-2, ПК-4, ПК-8

Теоретическое обоснование

На техническое задание существует стандарт ГОСТ 19.201-78 «Техническое задание. Требования к содержанию и оформлению». В соответствии с этим стандартом техническое задание должно содержать следующие разделы:

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

При необходимости допускается в техническое задание включать приложения. Рассмотрим пример разработки ТЗ для построения графиков элементарных функций и пример оформления титульного листа к ТЗ.

Пример технического задания 1

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

На рисунке 1.1 представлен пример титульного листа технического задания на учебный программный продукт. Текст технического задания приведен ниже.

1. ВВЕДЕНИЕ

Настоящее техническое задание распространяется на разработку программы построения графиков и таблиц значений функций одной переменной, предназначенной для использования школьниками старших классов.

В школьном курсе элементарной алгебры тема анализа функций является одной из самых сложных. При изучении данной темы школьники должны научиться исследовать и строить графики функций одной переменной, используя все известные характеристические точки функции, включая корни, точки разрыва первого и второго рода и т. д. Существующее программное обеспечение, которое может решать подобные задачи, является универсальным, например Eurica или MathCad. Оно имеет сравнительно сложный пользовательский интерфейс, ориентированный на пользователя, прослушавшего, как минимум, институтский курс высшей математики, что делает использование подобных средств школьниками невозможным.

Разрабатываемая программа позволит школьникам проверить свои знания при изучении указанной темы.

2. ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ

Программа разрабатывается на основе учебного плана кафедры «Прикладная информатика и дизайн», утвержденного на заседании кафедры 30.08.2013, протокол № 1.

3. НАЗНАЧЕНИЕ

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

4.ТРЕБОВАНИЯ К ПРОГРАММЕ ИЛИ ПРОГРАММНОМУ ИЗДЕЛИЮ

4.1.Требования к функциональным характеристикам

4.1.1. Программа должна обеспечивать возможность выполнения следующих функций:

- ввод аналитического представления функции одной переменной и длительное хранение его в системе;

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

УТВЕРЖДАЮ

Зав кафедрой _____

«___» _____201_ г.

ПРОГРАММА ПОСТРОЕНИЯ ТАБЛИЦ И ГРАФИКОВ ФУНКЦИЙ

Руководитель _____

Исполнитель _____

ФИО, группа

201_

Рисунок 1.1 – Образец титульного листа для технического задания на учебную программу

– ввод и изменение интервала определения функции;
– ввод и корректировку шага аргумента;
– построение таблицы значений функции на заданном интервале иди изображение графика функции на заданном интервале при условии, что на указанном интервале она не имеет точек разрыва.

4.1.2. Исходные данные:

– аналитическое задание функции;

– интервал определения функции;

– шаг изменения аргумента, определяющий количество точек на интервале.

4.2. Требования к надежности

4.2.1.Предусмотреть контроль вводимой информации.

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

4.3. Требования к составу и параметрам технических средств

4.3.1.Система должна работать на IBM совместимых персональных компьютерах.

4.3.2.Минимальная конфигурация:

– тип процессора .....Pentium и выше;

– объем оперативного запоминающего устройств .....32 Мб и более.

4.4. Требования к информационной и программной совместимости

Система должна работать под управлением семейства операционных систем Win 32 (Windows 95, Windows 98, Windows 2000, Windows NT и т. п.).

5. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

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

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

5.3.В состав сопровождающей документации должны входить:

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

5.3.2. Руководство пользователя.

5.3.3. Тексты программ.

Пример технического задания 2

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

1. ВВЕДЕНИЕ

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

2. ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ

Система разрабатывается на основании приказа декана факультета и в соответствии с планом мероприятий по совершенствованию учебного процесса на 2013-2014 учебный год.

3. НАЗНАЧЕНИЕ

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

4. ТРЕБОВАНИЯ К ПРОГРАММЕ ИЛИ ПРОГРАММНОМУ ИЗДЕЛИЮ

4.1. Требования к функциональным характеристикам.

4.1.1. Система должна обеспечивать возможность выполнения следующих функций:

- инициализацию системы (ввод списков групп, перечней изучаемых дисциплин в
- соответствии с учебными планами и т. п.);
- ввод и коррекцию текущей информации о ходе сдачи сессии конкретными
- студентами;
- хранение информации об успеваемости в течение времени обучения студента;
- получение сведений о текущем состоянии сдачи сессии студентами.

4.1.2. Исходные данные:

- списки студентов учебных групп;
- учебные планы кафедр - перечень предметов и контрольных мероприятий по каждому предмету;

- расписания сессий;
- текущие сведения о сдаче сессии каждым студентом.

4.1.3. Результаты:

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

4.2. Требования к надежности

4.2.1.Предусмотреть контроль вводимой информации.

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

4.2.3.Обеспечить целостность хранимой информации.

4.3. Требования к составу и параметрам технических средств

4.3.1.Система должна работать на IBM совместимых персональных компьютерах.

4.3.2.Минимальная конфигурация:

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

4.4. Требования к информационной и программной совместимости

Система должна работать под управлением семейства операционных систем Win 32 (Windows 95, Windows 98, Windows 2000, Windows NT и т. п.).

5. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

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

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

5.3.В состав сопровождающей документации должны входить:

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

5.3.2.Руководство системного программиста.

5.3.3.Руководство пользователя.

5.3.4.Графическая часть на трех листах формата А1:

5.3.4.1.Схема структурная программной системы.

5.3.4.2.Диаграмма компонентов данных.

5.3.4.3.Формы интерфейса пользователя.

Аппаратура и материалы. Для выполнения лабораторной работы необходим персональный компьютер со следующими характеристиками: процессор Intel с тактовой частотой 2000 МГц и выше; оперативная память – не менее 128 Мбайт; свободное дисковое пространство – не менее 800 Мбайт; устройство для чтения компакт-дисков; монитор типа Super VGA (число цветов – 256) с диагональю не менее 17 ″. **Программное обеспечение** – операционная система WINDOWS 98 / NT / ME / 2000 / XP/ Vista/ 7, Microsoft Word.

Указания по технике безопасности. Правила техники безопасности при выполнении лабораторной работы совпадают с общепринятыми для пользователей персональных компьютеров. Студентам запрещается самостоятельно производить ремонт персонального компьютера, установку и удаление программного обеспечения. В случае неисправности персонального компьютера необходимо сообщить об этом обслуживающему персоналу лаборатории (оператору, администратору). Необходимо соблюдать правила техники безопасности при работе с электрооборудованием, не касаться электрических розеток металлическими предметами; рабочее место пользователя персонального компьютера должно содержаться в чистоте; не разрешается возле персонального компьютера принимать пищу, напитки.

Методика и порядок выполнения работы

Составить ТЗ на информационную систему в соответствии с ГОСТ 19.104-78. Для оформления отчета можно использовать любой текстовой редактор. Варианты заданий приведены в таблице 1.1. Ваш вариант - последняя цифра в номере зачетной книжки.

Таблица 1.1 – Варианты заданий для разработки ТЗ

№ вари-анта	Тема	Описание предметной области
1	Система поддержки управления библиотекой	Система поддержки управления библиотекой должна обеспечивать операции над данными о читателях (добавление, удаление и изменение). В регистрационном списке читателей хранятся следующие сведения: фамилия, имя и отчество читателя; номер его читательского билета и дата выдачи билета,

		дата последней перерегистрации. Наряду с регистрационным списком системой должен поддерживаться каталог библиотеки, где хранится информация о книгах (наименованиях): название, список авторов, библиотечный шифр, год и место издания, название издательства, общее количество экземпляров книги в библиотеке и количество экземпляров, доступных в текущий момент. Система обеспечивает добавление, удаление и изменение данных каталога, а также поиск книг в каталоге на основании введенного шифра или названия книги или фамилии автора. Читатели имеют доступ только к каталогу книг (они могут осуществлять в нем только поиск и просмотр). В системе поддерживается реестр экземпляров всех книг библиотеки. Каждый экземпляр имеет свой уникальный идентификационный номер, вообще говоря, не совпадающий с библиотечным шифром. В системе осуществляется регистрация взятых и возвращенных читателем книг. Про каждый выданный экземпляр в реестре хранится запись о том, кому и когда была выдана книга, и когда она должна быть возвращена. При возврате книги в записи делается пометка, о том, что данный экземпляр находится в наличии и указывается, какой читатель пользовался этой книгой последним. Если экземпляр приходит в негодность, запись реестра о нем удаляется. Если от поставщиков приходят новые книги, записи о них добавляются в реестр и каталог. При любом обращении читателя в библиотеку сначала осуществляется проверка, не является ли он нарушителем правил пользования. Нарушителем считается тот читатель, который не вернул по истечении срока какую-либо книгу. Нарушители библиотекой не обслуживаются, до тех пор не вернут книги и не заплатят штраф. Перерегистрация читателей проходит раз в два года. Она необходима для поддержания списка читателей в актуальном состоянии. Если какой-либо читатель пропускает перерегистрацию, то по истечении полугода с момента перерегистрации его читательский билет аннулируется, сведения о нем удаляются из системы. Система должна выдавать библиотекарям следующую справочную информацию: – какие книги были выданы за данный промежуток времени; – какие книги были возвращены за данный промежуток времени; – какие книги находятся у данного читателя; – имеется ли в наличии некоторая книга.
2	Система складского учета	На складе для каждого товара фиксируется место хранения (определенная полка), количество товара и его наименование. Разные товары имеют разные единицы измерения: штуки, килограммы, коробки, бутылки и др. Система учета товаров должна обеспечивать добавление информации о новом товаре, изменение или удаление информации об имеющемся товаре, хранение (добавление, изменение и удаление) информации о

		поставщиках и покупателях, включающей в себя название фирмы, ее адрес и телефон. В системе учитывается приход товаров от поставщиков. В каждом приходе товаров могут содержаться несколько позиций, в каждой позиции указываются наименование товара и его количество. После оформления прихода товара в системе количество товара в инвентаризационной описи соответственно увеличивается. Товар со склада отпускается покупателям по расходным накладным. В каждой накладной могут содержаться несколько позиций, в каждой позиции указываются наименование товара и его количество. После оформления расхода товара в системе количество товара в инвентаризационной описи соответственно уменьшается. Система учета по требованию пользователя формирует и выдает на печать следующую справочную информацию: – список всех товаров; – инвентаризационную опись товаров, имеющихся в наличии; – список товаров, количество которых необходимо пополнить; – список товаров, поставляемых данным поставщиком; – все позиции в каком-либо приходе товара; – все позиции в какой-либо расходной накладной. Система осуществляет поиск информации о клиенте или поставщике по части названия фирмы. Это необходимо, чтобы работники склада могли связаться с фирмой по какому-либо вопросу.
3	Каталог Web ресурсов	В каталоге хранится следующая информация о ресурсах: название ресурса, уникальный адрес ресурса (URL), раздел каталога, в котором содержится ресурс, список ключевых слов, краткое описание, дата последнего обновления, контактная информация. Доступ пользователей к каталогу осуществляется при помощи браузера. Пользователи каталога могут добавлять новые ресурсы, информация о которых не была внесена ранее. Ресурсы в каталоге классифицируются по разделам. Полный список ресурсов каждого раздела должен быть доступен пользователям. Количество ресурсов в разделе может быть большим, поэтому пользователь может выбрать количество, отображаемое на одной странице, например 25, тогда на первой странице раздела отображается список из первых 25 ресурсов, на второй – следующие 25 и т. Д. Ресурсы в списке могут быть упорядочены по дате обновления или по названиям (по алфавиту). Пользователям каталога должны быть предоставлены возможности по поиску ресурсов в каталоге. Поиск осуществляется по ключевым словам. Если пользователь не доволен результатами поиска, он может уточнить запрос (осуществить поиск среди результатов предыдущего поиска). Должна быть возможность выдавать результаты поиска в разной форме (вывод всей информации о ресурсах или частичной). Пользователь может отсортировать результаты поиска по

		релевантности (соответствию ключевым словам из запроса) или по дате обновления. Поскольку содержание ресурсов Интернет со временем изменяется необходимо следить за датой последнего обновления, периодически опрашивая Web-сайты, URL которых хранятся в каталоге.
4	Система для ввода информации при приеме сотрудника на работу	Подразделение/кафедра выбирается путем позиционирования на определенной (нужной) строке. Подразделения меняются сравнительно часто и поэтому их названия и коды хранятся в соответствующих справочниках. Должности также выбираются в соответствии со штатным расписанием. Если соответствующие должности уже заняты, то ввод не может быть осуществлен. Каждому сотруднику в соответствии с имеющимися ограничениями должен быть присвоен тарифный разряд. Наряду с другими сведениями, вводится информация о знании иностранных языков. Языки выбираются из списка. Выбор может быть множественным (т.е. сотрудник может владеть более чем одним языком). Все выбранные позиции должны быть видны; пользователь может корректировать свой выбор перед тем, как окончательно занести данные в БД. Кроме названия языка, фиксируется еще и степень владения языком. Запрещается принимать на работу лиц пенсионного возраста. При приеме на преподавательскую должность образование должно быть высшим. При приеме на должность выше ассистента научно-педагогический стаж должен быть больше 3 лет. Научно-педагогический стаж не может быть больше общего стажа.
5	Технические ВУЗы	Представляют сведения о: – профессиях, по которым ведется обучение в ВУЗах страны (региона или края); – самих этих ВУЗах и времени начала подготовки по разным специальностям; – учебных планах и программах Для анализа состояния дел необходимо получить следующие оперативно обновляемые данные: – перечни информационных профессий, с указанием учебных заведений, ведущих подготовку специалистов по данным профессиям; сведения об информационных профессиях, подготовка по которым начата в текущем учебном году; – сводные списки учебных заведений, а также списки только тех учебных заведений, которые объявили прием по данной специальности в текущем году; – справочные данные по каждой информационной профессии (название, наличие государственного общеобразовательного стандарта; учебные планы подготовки; учебные программы; требования к теоретической и практической подготовке специалистов).
6	Медицинские ВУЗы	Представляют сведения о: – профессиях, по которым ведется обучение в ВУЗах края; – самих этих ВУЗах и времени начала подготовки по

		разным специальностям; – учебных планах и программах Для анализа состояния дел необходимо получить следующие оперативно обновляемые данные: – перечни медицинских профессий, с указанием учебных заведений, ведущих подготовку специалистов по данным профессиям; сведения о медицинских профессиях, подготовка по которым начата в текущем учебном году; – сводные списки учебных заведений, а также списки только тех учебных заведений, которые объявили прием по данной специальности в текущем году; – справочные данные по каждой медицинской профессии (название, наличие государственного общеобразовательного стандарта; учебные планы подготовки; учебные программы; требования к теоретической и практической подготовке специалистов).
7	Регистратура клиники	Цель клиники — оказание медицинских услуг по различным направлениям. В ней существуют следующие подразделения: – терапевтическое отделение; – педиатрическое отделение; – травматологическое отделение; – хирургическое отделение; – кабинет окулиста; – гинекологический кабинет; – «Семейный врач»; – кабинет нетрадиционной медицины; – кабинет компьютерной диагностики; – процедурный кабинет. В этих подразделениях работают врачи соответствующих специальностей, а также младший медицинский персонал. В регистратуре клиники ведется учет пациентов клиники. На каждого пациента заводится медицинская карта с индивидуальным номером и личными данными пациентов. У каждого специалиста свой график работы, определяющий время приема в течение рабочей недели. Любой желающий может записаться на прием в удобное для него время. Запись на прием регистрируется. Посещение или непосещение пациентом врача также фиксируется. Если клиент не явился на прием, то работник регистратуры может связаться с ним, выяснить причину и назначить время нового приема. Результаты визита пациента к специалисту — поставленный диагноз, назначенные анализы, лечебные назначения — подлежат регистрации. Ежедневно работник регистратуры составляет расписание приемов для каждого работающего в этот день специалиста. Ежемесячно готовятся отчеты о выполненном объеме работ различных специалистов. Раз в год составляется статистический отчёт о заболеваемости.
8	Касса автовокзала	Автовокзал осуществляет на договорной основе обслуживание пригородных, междугородных и международных автобусных рейсов, выполняемых различными перевозчиками. По требованию пассажира кассир может забронировать места на

		требуемом рейсе на указанную дату. Забронированные места должны быть выкуплены не менее чем за одни сутки до отправления рейса. При покупке билетов пассажир указывает пункт назначения, желаемый рейс, дату выезда, требуемое количество проездных и багажных мест. Кассир обязан оформить проездные и багажные документы, если имеются свободные места на указанные рейс и дату. Билет может быть продан не ранее чем за 14 суток до отправления автобуса. Если пассажир желает сдать билеты, то при сдаче за 1 сутки до отправления автобуса возвращается полная их стоимость. При сдаче билетов менее чем за сутки возвращается 50% стоимости. При неявке пассажира ко времени посадки в автобус стоимость билетов не возвращается. В случае утери билета его стоимость не возвращается, замещающий документ в виде справки не выдается, посадка пассажира на автобус не производится.
9	Web-сайт авиа-компании	Web-сайт авиакомпании позволяет пользователям: – узнать о выполнении рейсов текущего дня; – запросить информацию о расписании рейсов, стоимости билетов и наличии мест; – забронировать билеты. Система бронирования хранит заявки клиентов. Оформляя заявку, клиент указывает: тип билета (в одну сторону или "туда и обратно"); 1 или 2 номера рейсов и 1 или 2 даты вылета (в зависимости от типа билета); класс обслуживания (v.i.p. или эконом); количество мест; признак использования премиальных очков для бесплатного перелета или повышения класса обслуживания. Клиенту высвечивается тариф и общая стоимость заказанных билетов, запрашивается подтверждение брони. После окончания оформления заявки информация передается в базу наличия билетов и количество доступных билетов уменьшается. Клиент самостоятельно выкупает забронированные билеты в какой-либо из касс авиакомпании. Когда билеты выкуплены (информация об этом приходит из маркетинговой базы данных), заявка удаляется. Клиент может аннулировать заявку не позднее трех суток до вылета, при этом в БД наличия билетов делаются соответствующие изменения.
10	Гостиница	Гостиница предоставляет номера клиентам на определенный срок. Каждый номер характеризуется вместимостью, комфортностью (люкс, полулюкс, обычный) и ценой. Клиентами являются различные лица, о которых необходимо собирать определенную информацию (фамилия, имя, отчество и некоторый комментарий). Сдача номера клиенту производится при наличии свободных мест в номерах, подходящих клиенту по указанным выше параметрам. При поселении фиксируется дата поселения. При выезде из гостиницы для каждого места запоминается дата освобождения. Необходимо не только хранить информацию по факту сдачи номера клиенту, но и осуществлять бронирование номеров. Кроме того, для постоянных клиентов, а также для определенных категорий клиентов предусмотрена система скидок. Скидки могут суммироваться.

Задание. Составить техническое задание на разработку программного обеспечения для информационной системы по своему варианту.

Содержание отчета и его форма

Отчет по лабораторной работе должен состоять из:

1) титульного листа (образец приведен в лекции 9);

2) технического задания.

Вопросы для защиты работы

1. Назовите, какой раздел технического задания можно считать основным и почему.
2. Какую информацию должны содержать остальные разделы?
3. В чем основная сложность разработки технического задания?

Лабораторная работа № 2

РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ДИАГРАММ

Цель и содержание работы: изучить построение функциональных диаграмм, а также стандарты IDEF().

Формируемые компетенции: ПК-2, ПК-4, ПК-8

Теоретическое обоснование

Методология IDEF0 [1] является следующим этапом развития хорошо известного графического языка описания функциональных систем SADT, рассмотренного в 9.3. Термин SADT (Structured Analysis and Design Technique) обозначает технологию структурного анализа и проектирования. Термин IDEF обозначает Icam DEFinition - нотацию ICAM, которая является основной частью программы ICAM (Integrated Computer-Aided Manufacturing - интегрированная компьютеризация производства), которая проводилась по инициативе ВВС США. Методологии функционального моделирования семейства IDEF стандартизированы. С их помощью можно эффективно отображать и анализировать модели деятельности широкого спектра сложных систем.

Основной принцип методологии IDEF – это представление любой изучаемой системы в виде набора взаимодействующих и взаимосвязанных блоков, отображающих процессы, операции, действия, происходящие в изучаемой системе. В IDEF0 все, что происходит в системе и ее элементах, называется функциями. Каждой функции ставится в соответствие блок. На диаграмме IDEF0 блок представляет собой прямоугольник. Интерфейсы, посредством которых блок взаимодействует с другими блоками или с внешней по отношению к моделируемой системе средой, представляются стрелками, входящими в блок или выходящими из него. Входящие стрелки показывают, какие условия должны быть одновременно выполнены, чтобы функция, описываемая блоком, осуществилась. Диаграммы IDEF0 основаны на простой графике блоков и стрелок, метки для описания блоков и стрелок выполняются на естественном языке. Последовательная декомпозиция диаграмм строится по иерархическому принципу, при котором на верхнем уровне отображаются основные функции, а затем происходит их детализация и уточнение.

Разработка модели в IDEF0 представляет собой пошаговую, итеративную процедуру. На каждом шаге итерации разработчик предлагает вариант модели, который подвергают обсуждению, рецензированию и последующему редактированию, после чего цикл повторяется.

Основные определения (понятия) методологии и языка IDEF0.

1. Блок: прямоугольник, содержащий имя и номер и используемый для описания функции.

2. Ветвление: разделение стрелки на два или большее число сегментов.

3. Внутренняя стрелка: входная, управляющая или выходная стрелка, концы которой связывают источник и потребителя, являющиеся блоками одной диаграммы. Отличается от граничной стрелки.

4. Входная стрелка: класс стрелок, которые отображают вход IDEF0-блока, то есть данные или материальные объекты, которые преобразуются функцией в выход. Входные стрелки связываются с левой стороной блока IDEF0.

5. Выходная стрелка: класс стрелок, которые отображают выход IDEF0-блока, то есть данные или материальные объекты, произведенные функцией. Выходные стрелки связываются с правой стороной блока IDEF0.

6. Глоссарий: список определений для ключевых слов, фраз и аббревиатур, связанных с узлами, блоками, стрелками или с моделью IDEF0 в целом.

7. Граничная стрелка: стрелка, один из концов которой связан с источником или потребителем, а другой не присоединен ни к какому блоку на диаграмме. Отображает связь диаграммы с другими блоками системы и отличается от внутренней стрелки.

8. Декомпозиция: разделение моделируемой функции на функции – компоненты.

9. Дерево узлов: представление отношений между родительскими и дочерними узлами модели IDEF0 в форме древовидного графа.

10. Диаграмма A-0: специальный вид (контекстной) диаграммы IDEF0, состоящей из одного блока, описывающего функцию верхнего уровня, ее входы, выходы, управления, и механизмы, вместе с формулировками цели модели и точки зрения, с которой строится модель.

11. Диаграмма: часть модели, описывающая декомпозицию блока.

12. Диаграмма-иллюстрация (FEO): графическое описание, используемое, для сообщения специфических фактов о диаграмме IDEF0. При построении диаграмм FEO можно не придерживаться правила IDEF0.

13. Дочерний блок: блок на дочерней (порожденной) диаграмме.

14. Дочерняя диаграмма: диаграмма, детализирующая родительский (порождающий) блок.

15. Имя блока: глагол или глагольный оборот, помещенный внутри блока
и описывающий моделируемую функцию.

16. Интерфейс: разделяющая граница, через которую проходят данные или материальные объекты; соединение между двумя или большим числом компонентов модели, передающее данные или материальные объекты от одного компонента к другому.

17. Код ICOM: аббревиатура (Input - Вход, Control - Управление, Output - Выход, Mechanism – Механизм), код, обеспечивающий соответствие граничных стрелок дочерней диаграммы со стрелками родительского блока; используется для ссылок.

18. Контекст: окружающая среда, в которой действует функция (или комплект функций на диаграмме).

19. Контекстная диаграмма: диаграмма, имеющая узловой номер A-n ($n \geq 0$), которая представляет контекст модели, Диаграмма A-0, состоящая из одного блока, является необходимой (обязательной) контекстной диаграммой; диаграммы с узловыми номерами A-1, A-2,... - дополнительные контекстные диаграммы.

20. Метка стрелки: существительное или оборот существительного, связанные со стрелкой или сегментом стрелки и определяющие их значение.

21. Модель IDEF0: графическое описание системы, разработанное с определенной целью и с выбранной точки зрения. Комплект одной или более диаграмм IDEF0, которые изображают функции системы с помощью графики, текста и глоссария.

22. Номер блока: число (0 – 6), помещаемое в правом нижнем углу блока и однозначно идентифицирующее блок на диаграмме.

23. Перечень узлов: список, часто ступенчатый, показывающий узлы модели IDEF0 в упорядоченном виде. Имеет то же значение и содержание, что и дерево узлов (п. 9).

24. Примечание к модели: текстовый комментарий, являющийся частью диаграммы IDEF0 и используемый для записи факта, не нашедшего графического изображения.

25. Родительская диаграмма: диаграмма, которая содержит родительский блок.

26. Родительский блок: блок, который подробно описывается дочерней диаграммой.

27. Связывание/развязывание: объединение значений стрелок в составное значение (связывание в «пучок»), или разделение значений стрелок (развязывание «пучка»), выраженные синтаксисом слияния или ветвления стрелок.

28. Сегмент стрелки: сегмент линии, который начинается или заканчивается на стороне блока, в точке ветвления или слияния, или на границе (несвязанный конец стрелки).

29. Семантика: значение синтаксических компонентов языка.

30. Синтаксис: Структурные компоненты или характеристики языка и правила, которые определяют отношения между ними.

31. Слияние: объединение двух или большего числа сегментов стрелок в один сегмент.

32. С-номер: номер, создаваемый в хронологическом порядке и используемый для идентификации диаграммы и прослеживания ее истории; может быть использован в качестве ссылочного выражения при определении конкретной версии диаграммы.

33. Стрелка: направленная линия, состоящая из одного или нескольких сегментов, которая моделирует открытый канал или канал, передающий данные или материальные объекты от источника (начальная точка стрелки), к потребителю (конечная точка с «наконечником»). Имеется 4 класса стрелок: входная стрелка, выходная стрелка, управляющая стрелка, стрелка механизма (включает стрелку вызова).

34. Стрелка вызова: вид стрелки механизма, который обозначает обращение из блока данной модели (или части модели) к блоку другой модели (или другой части той же модели) и обеспечивает связь между моделями или между разными частями одной модели.

35. Стрелка механизма: класс стрелок, которые отображают механизмы IDEF0, то есть средства, используемые для выполнения функции; включает специальный случай стрелки вызова. Стрелки механизмов связываются с нижней стороной блока IDEF0.

36. Стрелка, помещенная в туннель (туннельная стрелка): стрелка (со специальной нотацией), не удовлетворяющая обычному требованию, согласно которому каждая стрелка на дочерней диаграмме должна соответствовать стрелкам на родительской диаграмме.

37. Текст: любой текстовый (не графический) комментарий к графической диаграмме IDEF0.

38. Тильда: небольшая ломаная (волнистая) линия, используемая для соединения метки с конкретным сегментом стрелки или примечания модели с компонентом диаграммы.

39. Точка зрения: указание на должностное лицо или подразделение организации, с позиции которого разрабатывается модель.

40. Узел: блок, порождающий дочерние блоки; родительский блок.

41. Узловая ссылка: код, присвоенный диаграмме, для ее идентификации и определения положения в иерархии модели; формируется из сокращенного имени модели и узлового номера диаграммы с дополнительными расширениями.

42. Узловой номер диаграммы: часть узловой ссылки диаграммы, которая соответствует номеру родительского блока.

43. Узловой номер: код, присвоенный блоку и определяющий его положение в иерархии модели; может быть использован в качестве подробного ссылочного выражения.

44. Управляющая стрелка: класс стрелок, которые в IDEF0 отображают управления, то есть условия, при выполнении которых выход блока будет правильным. Данные или

объекты, моделируемые как управления, могут преобразовываться функцией, создающей соответствующий выход. Управляющие стрелки связываются с верхней стороной блока IDEF0.

45. Функция: деятельность, процесс или преобразование (моделируемые блоком IDEF0), идентифицируемое глаголом или глагольной формой, которая описывает, что должно быть выполнено.

46. Цель: краткая формулировка причины создания модели.

В основе методологии IDEF0 лежат четыре основных понятия, описанные в разделе 13 лекций:

- функциональный блок (Activity Box);
- интерфейсная дуга (Arrow) или стрелка;
- декомпозиция (Decomposition);
- глоссарий (Glossary).

Функциональный блок показан на рисунке 2.1. Внутри каждого блока размещается его имя и номер. Имя должно быть активным глаголом или глагольным оборотом, описывающим функцию. Номер блока размещается в правом нижнем углу. Номера блоков используются для их идентификации на диаграмме и в соответствующем тексте.

Рисунок 2.1 – Блок

Стрелка (дуга) формируется из одного или более отрезков прямых линий и наконечника на одном конце. Отрезки стрелок могут быть прямыми или ломаными; в последнем случае горизонтальные и вертикальные отрезки стрелки сопрягаются дугами, имеющими угол 90º. Правила построения стрелок показаны на рисунке 9.2.

Концы стрелок должны касаться внешней границы функционального блока, но не должны пересекать ее. Стрелки должны присоединяться к блоку на его сторонах, а не в углах.

Каждая сторона функционального блока имеет стандартное значение и однозначно определяет роль присоединенной стрелки. Стрелки, входящие в левую сторону блока обозначают входы. Входы преобразуются или расходуются функцией, чтобы создать то, что появится на ее выходе. Стрелки, входящие в блок сверху, соответствуют управлению. Стрелки, выходящие справа, обозначают выходы, т.е. данные или материальные объекты, произведенные функцией.

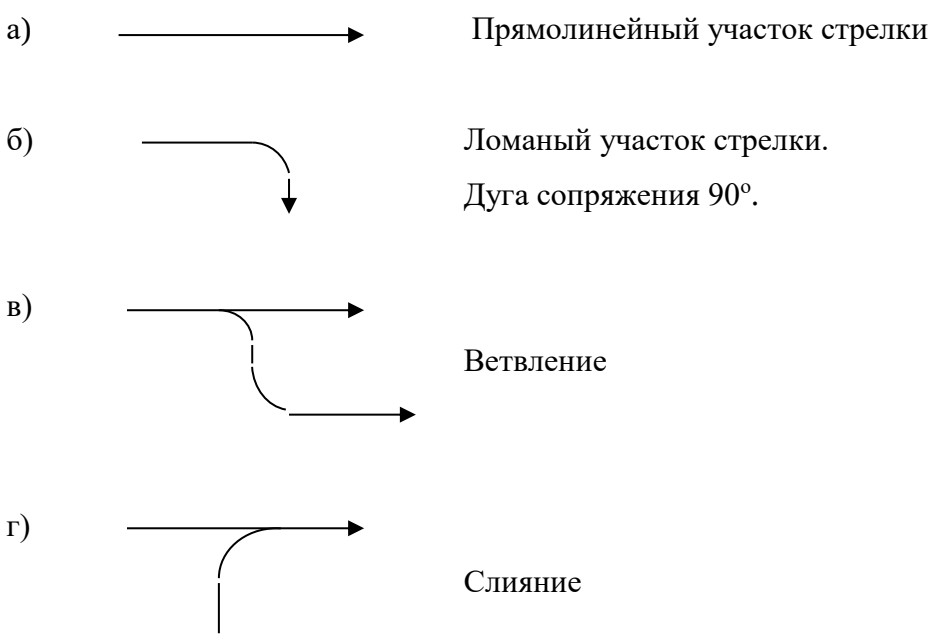

Рисунок 2.2 – Построение стрелок

Стрелки, подключенные к нижней стороне блока, представляют механизмы. Стрелки, направленные вверх, идентифицируют средства, поддерживающие выполнение функции. Другие средства могут наследоваться из родительского блока. Стрелки механизма, направленные вниз, являются стрелками вызова. Стрелки вызова обозначают обращение из данной модели или из данной части модели к блоку, входящему в состав другой модели или другой части модели, обеспечивая их связь, т.е. разные модели или разные части одной и той же модели могут совместно использовать один и тот же элемент (блок). Стандартное расположение стрелок показано на рисунке 9.3.

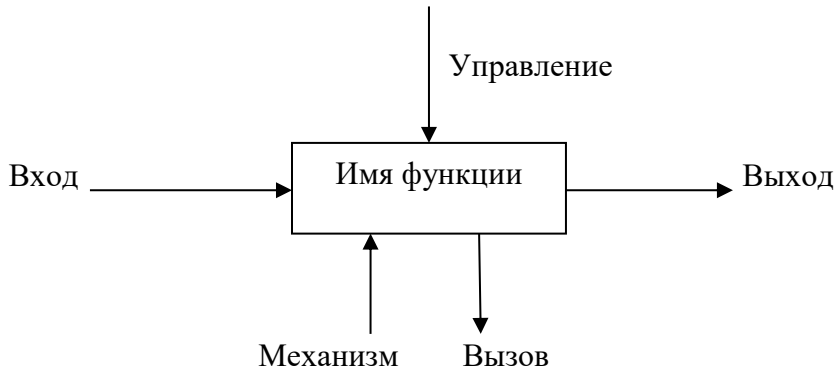

Рисунок 2.3 – Стандартное расположение стрелок

Примеры имен функций (функциональных блоков):

– производить детали;

– планировать ресурсы;
– наблюдать за выполнением;
– проектировать систему;
– эксплуатировать;
– разработать чертежи;
– изготовить деталь;
– проверять деталь.

Стрелки идентифицируют данные или материальные объекты, необходимые для выполнения функции или производимые ею. Каждая стрелка должна быть помечена существительным или оборотом существительного, например:

– спецификации;
– отчет об испытаниях;
– бюджет;
– конструкторские требования;
– конструкция детали;
– директива;
– инженер-конструктор
– плата в сборе;
– требования.

В метках стрелок не должны использоваться такие общеупотребительные термины, как: функция, вход, управление, выход, механизм, вызов.

Каждая модель должна иметь контекстную диаграмму верхнего уровня, на которой объект моделирования представлен единственным блоком с граничными стрелками. Эта диаграмма называется А-0. Стрелки на этой диаграмме отображают связи объекта моделирования с окружающей средой. Поскольку единственный блок представляет весь объект, его имя – общее для всего проекта. Это же справедливо и для всех стрелок диаграммы, поскольку они представляют полный комплект внешних интерфейсов объекта. Диаграмма А-0 устанавливает область моделирования и ее границу. Пример диаграммы А-0 показан на рисунке 2.4.

Контекстная диаграмма А-0 также должна содержать краткие утверждения, определяющие точку зрения должностного лица или подразделения, с позиций которого создается модель, и цель, для достижения которой ее разрабатывают. Изменение точки зрения, приводит к рассмотрению других аспектов объекта. Аспекты, важные с одной точки зрения, могут не появиться в модели, разрабатываемой с другой

точки зрения на тот же самый объект. Формулировка цели выражает причину создания модели, т.е. содержит перечень вопросов, на которые должна отвечать модель, что в значительной мере определяет ее структуру.

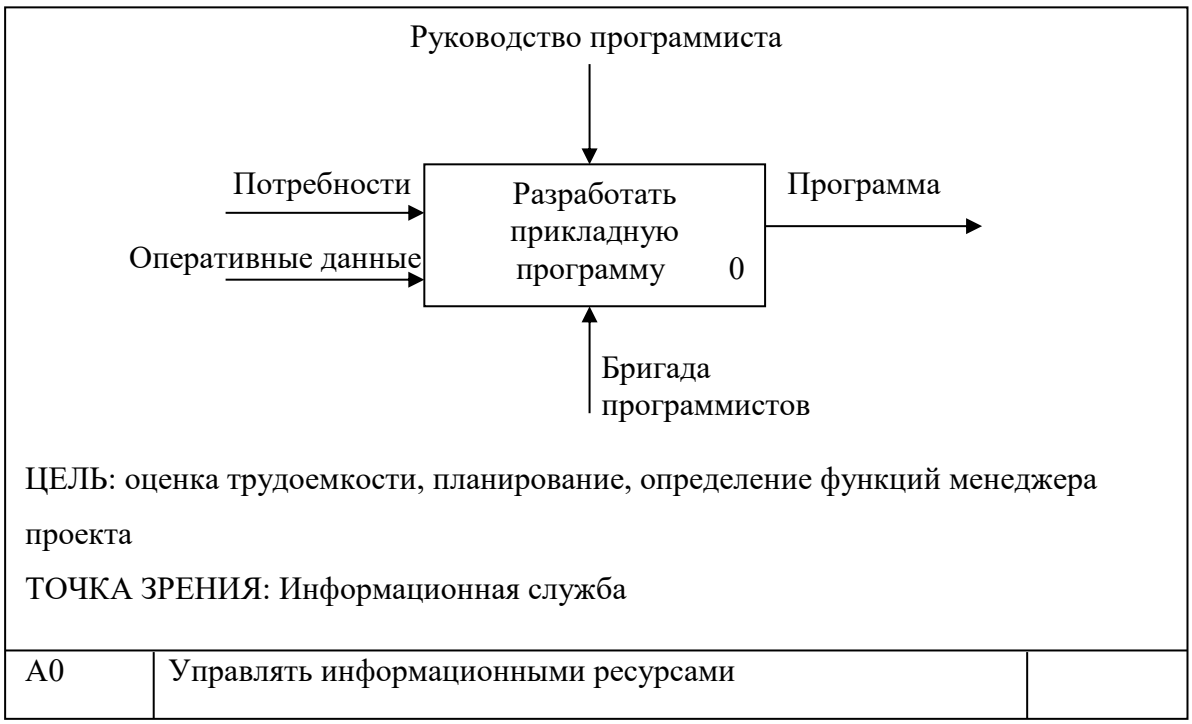

Рисунок 2.4 – Контекстная диаграмма

Наиболее важные свойства объекта обычно выявляются на верхних уровнях иерархии. По мере декомпозиции функции (функционального блока) верхнего уровня и разбиения на подфункции, эти свойства уточняются. Каждая подфункция может быть представлена декомпозицией на элементы следующего уровня, и так происходит до тех пор, пока не будет получена релевантная структура, позволяющая ответить на вопросы, сформулированные в цели моделирования. Каждая подфункция моделируется отдельным блоком. Каждый родительский блок подробно описывается дочерней диаграммой на более низком уровне. Все дочерние диаграммы не должны выходить за пределы области контекстной диаграммы верхнего уровня.

Дочерняя диаграмма

Функция, представленная на контекстной диаграмме верхнего уровня, может быть разложена на основные подфункции путем создания дочерней диаграммы. В свою очередь, каждая из этих подфункций может быть разложена на составные части путем создания дочерней диаграммы следующего, более низкого уровня. Каждая дочерняя диаграмма содержит дочерние блоки и стрелки, обеспечивающие дополнительную детализацию родительского блока.

Дочерняя диаграмма, создаваемая при декомпозиции, охватывает ту же область, что и родительский блок, но описывает ее более подробно. Поэтому дочернюю диаграмму можно рассматривать как вложенную в свой родительский блок. Эта структура показана на рисунке 2.5.

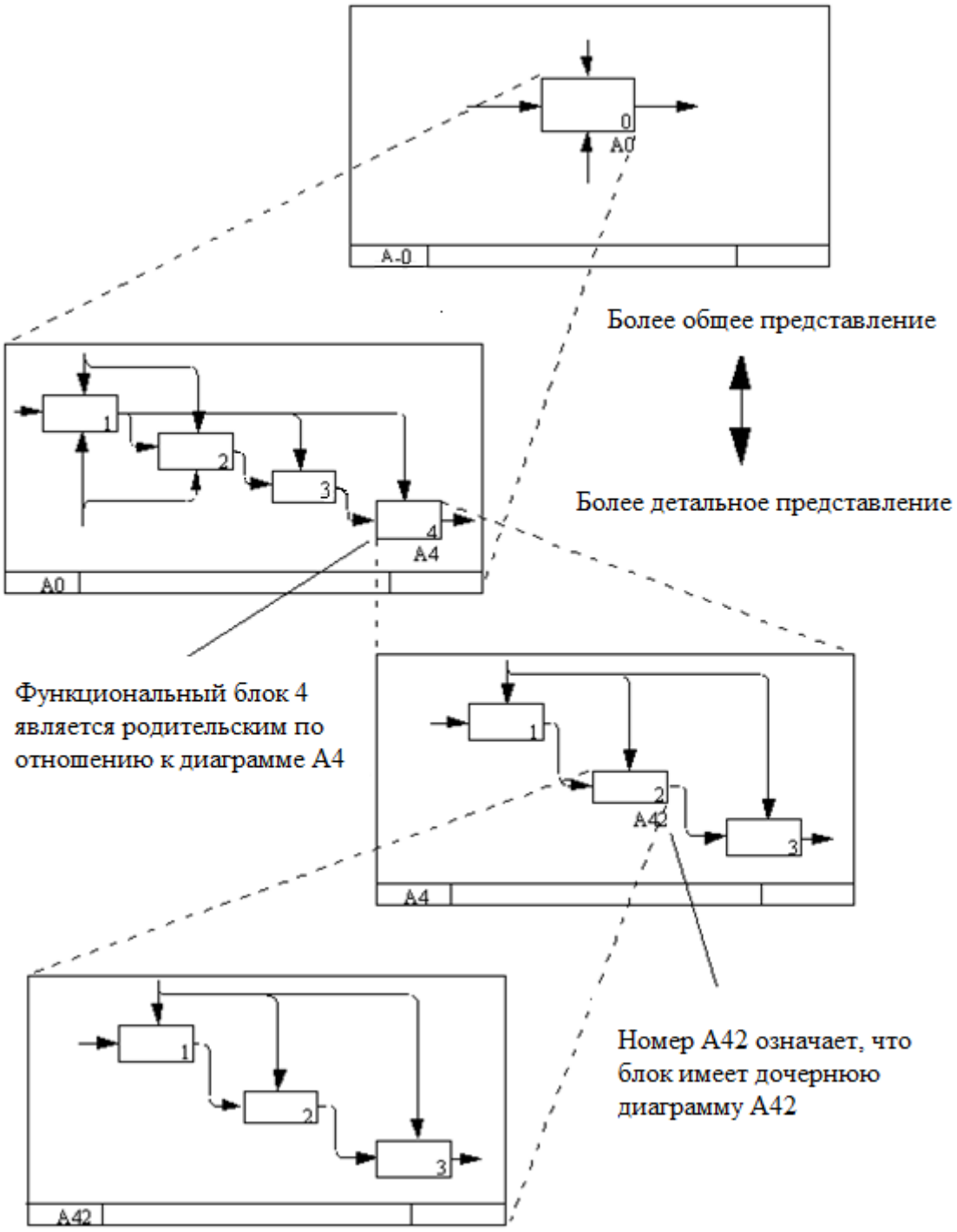

Рисунок 2.5 - Декомпозиция функциональных блоков

Родительская диаграмма

Родительской называется диаграмма, которая содержит один или более родительских блоков. Каждая не контекстная диаграмма является также дочерней

диаграммой, поскольку, по определению, она подробно описывает некоторый родительский блок. Таким образом, любая диаграмма может быть как родительской диаграммой (содержать родительские блоки), так и дочерней (подробно описывать собственный родительский блок). Аналогично, блок может быть как родительским (подробно описываться дочерней диаграммой) так и дочерним (появляющимся на дочерней диаграмме).

В методологии IDEF0 существует 6 (шесть) типов отношений между блоками в пределах одной диаграммы:

- доминирование;
- управление;
- выход - вход;
- обратная связь по управлению;
- обратная связь по входу;
- выход – механизм.

Доминирование определяется взаимным расположением блоков на диаграмме. Предполагается, что блоки, расположенные на диаграмме выше и левее, доминируют над блоками, расположенными ниже и правее. Доминирование подразумевает влияние, которое один блок оказывает на другие блоки диаграммы. Остальные пять отношений описывают связи между блоками и изображаются соответствующими стрелками.

Стрелки, помещенные в «туннель»

Туннель - круглые скобки в начале и|или окончании стрелки. Туннельные стрелки означают, что данные, выраженные этими стрелками, не рассматриваются на родительской диаграмме и|или на дочерней диаграмме. Стрелка, помещенная в туннель там, где она присоединяется к блоку, означает, что данные, выраженные этой стрелкой, не обязательны на следующем уровне декомпозиции. Стрелка, помещаемая в туннель на свободном конце, означает, что выраженные ею данные отсутствуют на родительской диаграмме. Туннель используется, чтобы не загромождать диаграмму.

Правила построения диаграмм

1. В составе модели должна присутствовать контекстная диаграмма А-0, которая содержит только один блок.

2. Блоки на диаграмме должны располагаться по диагонали – от левого верхнего угла диаграммы до правого нижнего в порядке присвоенных номеров. Блоки на диаграмме, расположенные вверху слева доминируют над блоками, расположенными внизу справа. Расположение блоков на листе диаграммы отражает авторское понимание

доминирования. Таким образом, топология диаграммы показывает, какие функции оказывают большее влияние на остальные.

3. Не контекстные диаграммы должны содержать не менее трех и не более шести блоков. Эти ограничения поддерживают сложность диаграмм на уровне, доступном для чтения, понимания и использования.

4. Каждый блок не контекстной диаграммы получает номер, помещаемый в правом нижнем углу; порядок нумерации - от верхнего левого к нижнему правому блоку (номера от 1 до 6).

5. Каждый блок, подвергнутый декомпозиции, должен иметь ссылку на дочернюю диаграмму; ссылка (например, узловой номер, С-номер или номер страницы) помещается под правым нижним углом блока.

6. Имена блоков (выполняемых функций) и метки стрелок должны быть уникальными. Если метки стрелок совпадают, это значит, что стрелки отображают тождественные данные.

7. При наличии стрелок со сложной топологией целесообразно повторить метку для удобства ее идентификации.

8. Следует обеспечить максимальное расстояние между блоками и поворотами стрелок, а также между блоками и пересечениями стрелок для облегчения чтения диаграммы. Одновременно уменьшается вероятность перепутать две разные стрелки.

9. Блоки всегда должны иметь хотя бы одну управляющую и одну выходную стрелку, но могут не иметь входных стрелок.

10. Если одни и те же данные служат и для управления, и для входа, вычерчивается только стрелка управления. Этим подчеркивается управляющий характер данных и уменьшается сложность диаграммы.

11. Максимально увеличенное расстояние между параллельными стрелками облегчает размещения меток, их чтение и позволяет проследить пути стрелок.

12. Стрелки связываются (сливаются), если они представляют сходные данные и их источник не указан на диаграмме.

13. Обратные связи по управлению должны быть показаны стрелками, расположенными сверху и над блоком. Обратные связи по входу должны быть показаны стрелками, расположенными внизу и под блоком. Так же показываются обратные связи посредством механизма. Таким образом обеспечивается показ обратной связи при минимальном числе линий и пересечений.

Методика и порядок выполнения работы

Изучить теоретическое обоснование и материал лекций. По согласованию с

преподавателем выбрать предметную область для построения функциональной диаграммы. Для построения диаграммы можно использовать средства текстового процессора Ms Word или любой графический редактор.

Задание 1. Составить контекстную функциональную диаграмму для выбранной предметной области.

Задание 1. Произвести декомпозицию функциональных блоков для диаграммы, построенной по заданию 1. Получить декомпозицию с количеством блоков менее шести, но более трех.

Содержание отчета и его форма

Отчет по лабораторной работе должен состоять из:

1) названия лабораторной работы;

2) функциональной контекстной диаграммы, построенной при выполнении задания 1.

3) функциональной диаграммы, содержащей дочерние и родительские блоки, построенной при выполнении задания 2.

Вопросы для защиты работы:

1. Перечислите основные компоненты функциональной диаграммы.

2. Какие диаграммы называются контекстными?

3. Какие диаграммы называются родительскими?

4. Какие диаграммы называются дочерними?

5. Каким образом на функциональных диаграммах отображается доминирование одних функциональных блоков над другими?

Лабораторная работа № 3

ДИАГРАММЫ ПРЕЦЕДЕНТОВ (ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ)

Цель и содержание работы: изучить порядок построения диаграммы прецедентов с помощью ArgoUML.

Формируемые компетенции: ПК-2, ПК-4, ПК-8

Теоретическое обоснование

В последнее десятилетие прошлого тысячелетия, объектно-ориентированная (ОО) технология, наконец, завершила превращение из лабораторного инструмента 1960-х в основную парадигму программного обеспечения. Путь развития ОО технологии был длинным и трудным в первую очередь потому, что для ее применения был необходим значительный сдвиг в мышлении программистов, дизайнеров, разработчиков и других лиц, участвующих в жизненном цикле разработки программного обеспечения. Основной предпосылкой, лежащей в основе ОО технологии, является моделирование. ОО технология разработана и реализована по существу, как моделирование реального мира с помощью программного обеспечения артефактов. Эта посылка дает такие мощные результаты благодаря своей простоте. При анализе, разработке и внедрении ОО систем могут быть использованы идеи, изложенные на одном и том же языке. Это позволяет применять моделирование при разработке и испытаниях вместо того, чтобы в первую очередь на самом деле строить систему. Эта особенность в сочетании с возможностью проектировать системы на очень высоком уровне благодаря использованию опыта практиков позволила осуществить разработку и успешную реализацию более сложных систем, чем это было возможно ранее.

Принятие унифицированного языка моделирования (UML) в качестве стандартного языка для описания объектно-ориентированного подхода продолжило продвижение ОО технологии в мейнстрим.

Программа ArgoUML была задумана как инструмент описания окружающей среды для использования при анализе и проектировании объектно-ориентированных программных систем. В этом смысле программа ArgoUML похожа на многие коммерческие CASE-средства для моделирования программных систем. Но ArgoUML имеет ряд очень важных отличий многих из этих инструментов:

- ArgoUML опирается на исследования в области когнитивной психологии и включает в себя ряд функций, которые способствуют повышению

производительности за счет поддержки когнитивных (познавательных) потребностей разработчиков объектно-ориентированного программного обеспечения.

- ArgoUML поддерживает открытые стандарты - UML, XMI, SVG, OCL и другие. Поэтому ArgoUML продолжает развиваться и имеет преимущество перед коммерческими CASE-средствами.
- ArgoUML является 100% приложением Java. Это позволяет ArgoUML работать на всех платформах, для которых доступна платформа Java2.
- ArgoUML является программным продуктом с открытым исходным кодом. Открытый код гарантирует, что разработчики и исследователи программного обеспечения нового поколения теперь будут иметь проверенную основу, которая позволит им управлять развитием и эволюцией CASE- технологий.

ArgoUML является предметно-ориентированной средой разработки, что обеспечивает когнитивную поддержку объектно-ориентированного проектирования. ArgoUML обеспечивает те же функции автоматизации, что и коммерческие CASE-средства, но она ориентирована на поддержку когнитивных (познавательных) потребностей разработчиков. Эти познавательные потребности описаны в тех когнитивных теориях:

1) рефлексия-в-действии (reflection-in-action);

2) оппортунистический дизайн (opportunistic design);

3) понимание и решение проблем (comprehension and problem solving).

Первый выпуск ArgoUML стал доступен в 1998 году. С настоящее время программа ArgoUML стала популярной в образовательной и коммерческой сферах. Узнать больше о проекте Арго можно, используя ссылки http://argouml.tigris.org и http://argouml.tigris.org/servlets/ProjectMailingListList

Для установки можно использовать файл ArgoUML-0.34-setup.exe, который предоставляется вместе с электронной версией методических указаний к лабораторным работам. Программа ArgoUML работает на операционных системах:

- Windows 7 x64
- Windows 7
- Windows Vista x64
- Windows Vista
- Windows 2003
- Windows XP x64

- Windows XP
- Windows 2000
- Mac OS X .

После загрузки программы ArgoUML на экране монитора появляется главное окно, показанное на рисунке 3.1. В верхней части экрана находится *строка меню.* Если при установке в качестве рабочего языка был выбран русский, то наименования команд в строке меню будет таким же, как на рисунке. Меню содержит интуитивно понятные команды, например, в меню Файл можно сохранить проект или открыть другой проект.

По умолчанию главное окно программы разделено на 4 части или панели. Размеры панелей можно изменять, перетаскивая влево, вправо, вверх и вниз границу между ними. В нижней части окна находится строка *состояния.* По часовой стрелке от верхнего левого угла находятся навигационная панель, панель редактирования, область сведений (панель деталей) и «To-Do» области.

1. Верхний левый угол. Навигационная панель показывает древовидную структуру всех классов, интерфейсов, типов данных и объектов. Эта структура может быть адаптирована к потребностям пользователя путем фильтрации объектов, которые можно перетаскивать на диаграммы.

2. В правом верхнем углу находится панель редактирования, которая показывает текущую диаграмму. Объекты на диаграмму можно перетаскивать из навигационной панели, или создавать с помощью команд меню, при наведении курсора на выбранный объект.

3. Внизу слева на панели «To do» выводится список всех выполненных пользователем действий для рассматриваемой модели.

4. Внизу справа на панели деталей содержится подробная информация о выбранном объекте (актер, связь, прецедент).

Команды меню «Файл» представлены в таблице 3.1.

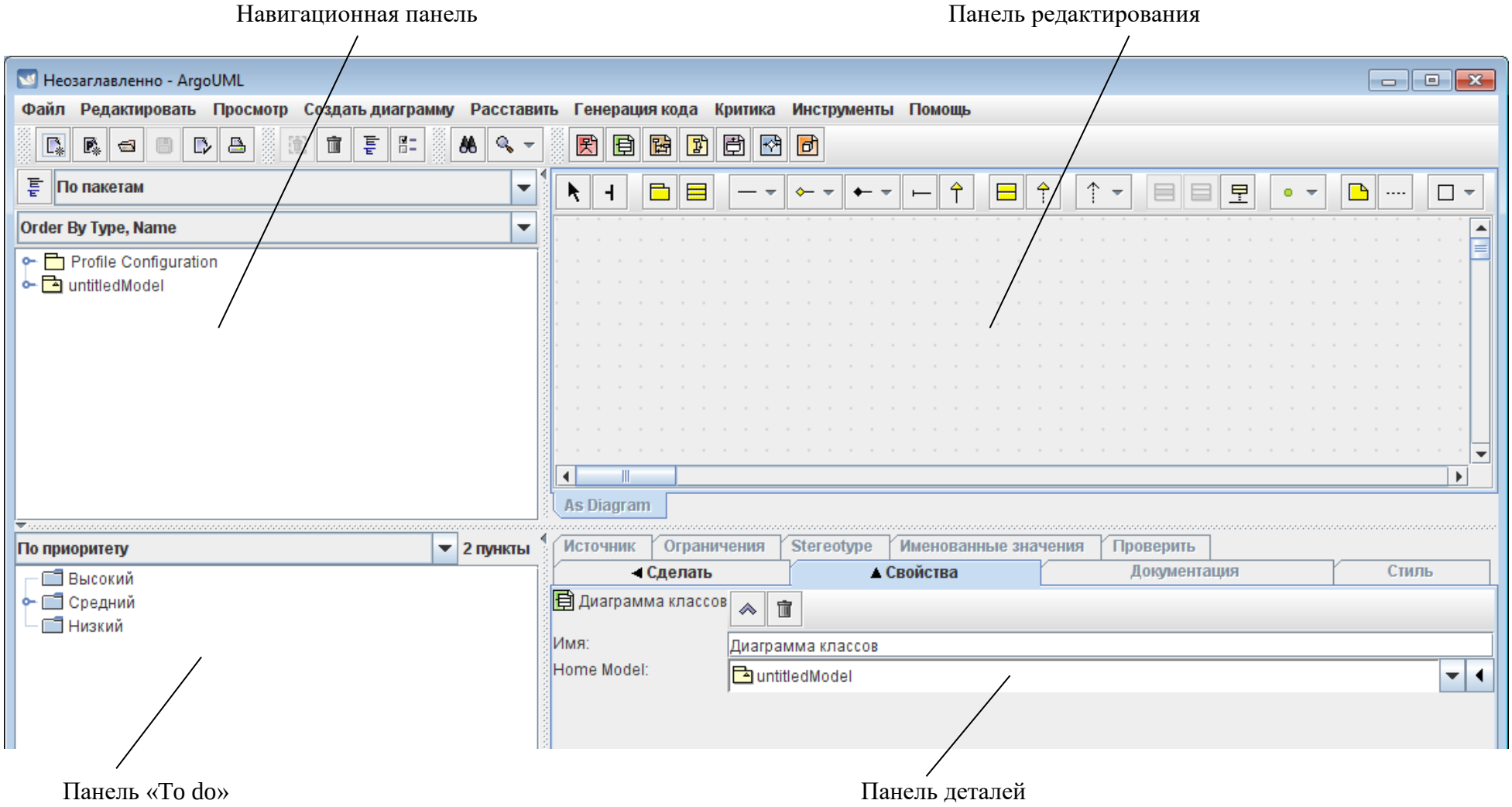

Рисунок 3.1 – Интерфейс пользователя ArgoUML

Таблица 3.1 - Команды меню «Файл»

Команда	Описание
	Создать новый проект
	Открыть проект
	Сохранить проект
	Установка параметров страницы: размера, ориентации, других опций
	Стандартное диалоговое окно для выбора параметров печати для текущей диаграммы

Команды меню «Редактировать» (The Edit Menu) представлены в таблице 3.2. Это меню позволяет пользователю выбирать элементы и размещать их на текущей панели редактирования, удалять элементы и управлять пользовательскими установками.

Таблица 3.2 - Команды меню «Редактировать»

Команда	Описание
`Select All` (Ctrl-A)	Выбор всех элементов модели на текущую панель.
 `Navigate Back` (Вернуться назад)	Перемещение к предыдущему элементу. Если его нет, то команда не активна.
 `Navigate Forward` (Продвинуться вперед)	Отмена команды  . Перемещение к следующему элементу. Если его нет, то команда не активна.
`Invert Selection`	Инвертирует текущее выделение на текущей панели. Все выбранные и не выбранные элементы на текущей диаграмме меняются местами
	Удаление выбранного элемента из текущей диаграммы. В модели элемент остается.
 (Ctrl-Delete)	Полное удаление выбранного элемента из модели. Если удаляемый элемент присутствует на других диаграммах, то появляется диалоговое окно (рисунок 3.2).
	Конфигурация (Сконфигурировать перспективы)
 `Settings...` (Установки)	Открывает диалоговое окно для выбора параметров установки (рисунок 3.3).

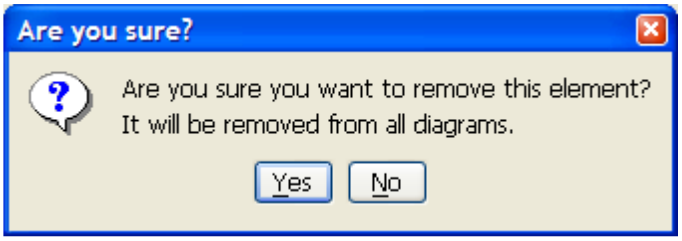

Рисунок 3.2 – Диалоговое окно для согласования полного удаления элемента из всех диаграмм

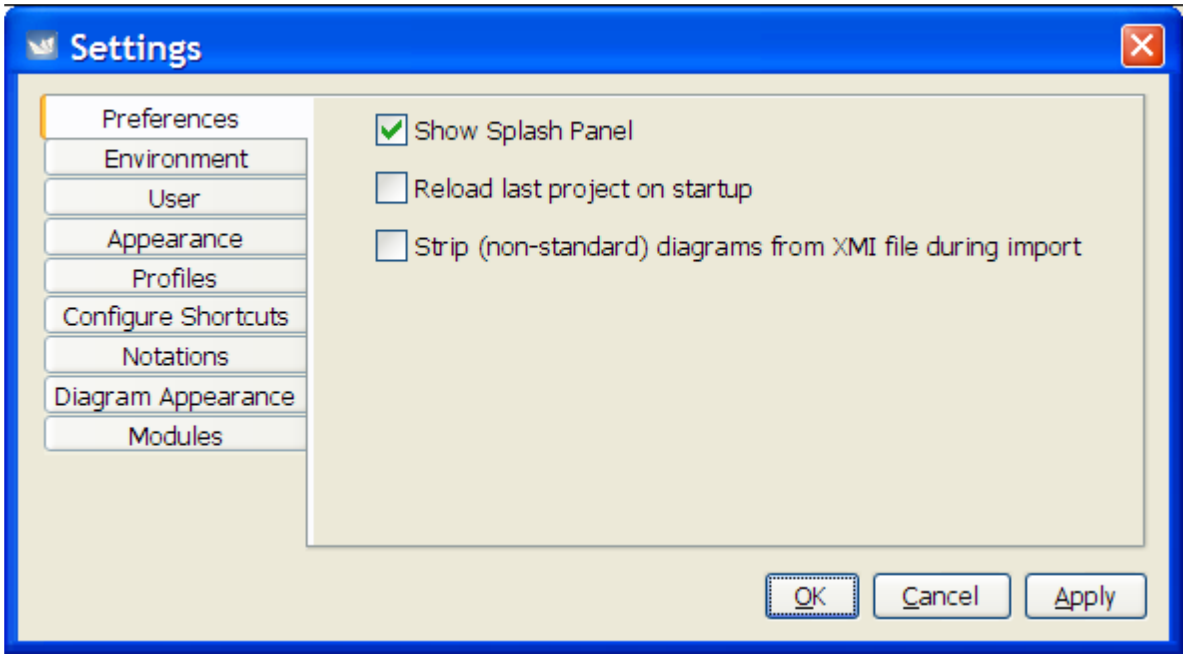

Рисунок 3.3 – Диалоговое окно для выбора параметров установки

Команды меню «Просмотр» (The View Menu) используется для просмотра различных диаграмм. Они представлены в таблице 3.3.

Таблица 3.3 - Команды меню «Просмотр»

Команда	Описание
Goto Diagram... (Перейти к диаграмме)	Открывает немодальное диалоговое окно, в котором перечисляются все диаграммы текущего проекта
Find... (Найти…)	Открывает немодальное диалоговое окно для поиска
Zoom (Масштаб)	Изменяет масштаб. Подменю содержит команды: • `Zoom Out`. Shortcut (Ctrl-Minus). Уменьшает размер и увеличивает обзор. • `Zoom Reset`. Возвращает размер к установленному по умолчанию (т.е. 100%). • `Zoom In`. Shortcut (Ctrl-Plus). Увеличивает масштаб.
Adjust Grid (Настройка решетки)	Позволяет регулировать параметры решетки на панели регулирования: • Линии `16`: решетка из сплошных линий с размером клеток 16 пикселей. • Линии `8`: решетка из сплошных линий с размером клеток 8 пикселей. • Точки `16`: решетка из точек с размером клеток 16 пикселей (по умолчанию). • Точки `32`: решетка из точек с размером клеток 32 пикселя. • Решетка отсутствует.

Adjust Snap (Регулировка оснастки)	Позволяет регулировать привязку к сетке на расстоянии в пикселях: • `Snap 4` • `Snap 8` (по умолчанию). • `Snap 16` • `Snap 32`
Adjust PageBreaks	Настройка разрыва страниц
Toolbars	Позволяет скрывать или выводить на экран команды. По умолчанию все команды показаны.
XML Dump	Выводит контент текущего проекта в XML формате.

Меню «Создать диаграмму» (The Create Menu) используется для создания различных UML диаграмм. Список диаграмм выводится, если нажать на команду «Создать диаграмму». Они представлены в таблице 3.4.

Таблица 3.4 - Команды меню «Создать диаграмму»

Команда	Описание
	Команда для создания диаграммы прецедентов (вариантов использования)
	Команда для создания диаграммы классов
	Команда для создания диаграммы последовательностей
	Команда для создания диаграммы кооперации (collaboration)
	Команда для создания диаграммы деятельности (Activity Diagram)
	Команда для создания диаграммы размещения (Deployment Diagram)

Меню «Расставить» используется для управления размещением элементов на рабочей панели. Команды меню «Расставить» представлены в таблице 3.5.

Таблица 3.5 - Команды меню «Расставить»

Команда	Описание
Выровнять (Align)	Команда для выравнивания имеет подменю: • Выравнивание по верхней границе. • Выравнивание по нижнему краю • Выравнивание по правому краю • Выравнивание по левому краю • Выравнивание по центру. Все выбранные элементы размещаются симметрично относительно вертикальной линии, проходящей через центры элементов • Все выбранные элементы размещаются симметрично относительно горизонтальной линии, проходящей через центры элементов • Выравнивание по сетке. Все вершины выбранных элементов и их

	правые края выравниваются по границам сетки
Распределить (Distribute)	Команда для управления размещением элементов по горизонтали и вертикали имеет подменю: • Размещение элементов, при котором крайний элемент слева и крайний элемент справа не перемещаются, а находящиеся между ними элементы перемещаются таким образом, что расстояние между краями элементов вдоль горизонтали устанавливается одинаковым; • Размещение элементов, при котором крайний элемент слева и крайний элемент справа не перемещаются, а находящиеся между ними элементы перемещаются таким образом, что расстояние между центрами элементов вдоль горизонтали устанавливается одинаковым; • Размещение элементов, при котором крайний элемент сверху и крайний элемент снизу не перемещаются, а находящиеся между ними элементы перемещаются таким образом, что расстояние между краями элементов вдоль вертикали устанавливается одинаковым; • Размещение элементов, при котором крайний элемент сверху и крайний элемент снизу не перемещаются, а находящиеся между ними элементы перемещаются таким образом, что расстояние между центрами элементов вдоль вертикали устанавливается одинаковым.
Переложить (Reorder)	Команда для управления упорядочиванием элементов (на переднем или заднем плане) имеет подменю: •`Forward` , выбранный элемент перемещается вперед (на передний план) относительно соседнего; •`Backward`, выбранный элемент перемещается назад (на задний план) относительно соседнего; •`To Front`, выбранный элемент перемещается вперед (на передний план) по отношению ко всем элементам; •`To Back`, выбранный элемент перемещается назад (на задний план) по отношению ко всем элементам.
Size To Fit Contents	Команда устанавливает минимальный размер текстовых надписей таким образом, чтобы они поместились внутри элементов на диаграмме

Панель «To-do» расположена в нижней части главного окна слева (рисунок 9.1).

Щелчок справа от команды «Средний» выводит сообщение о том, что задания имеют средний приоритет (рисунок 3.4)

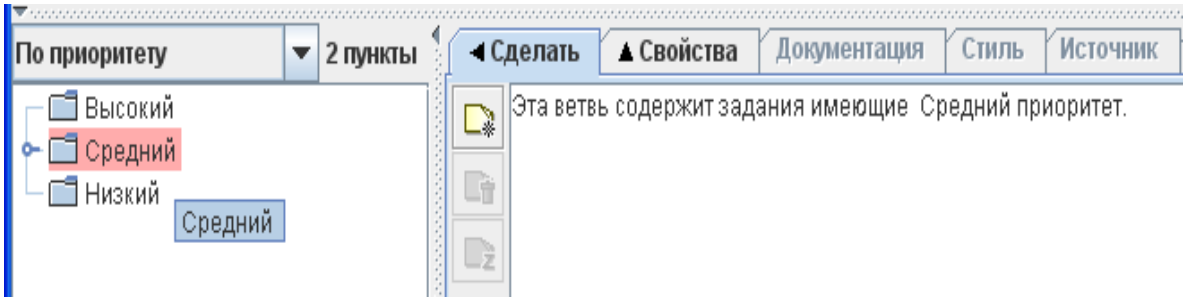

Рисунок 3.4 – Панель «To-do» (слева) и панель деталей (справа)

Щелчок слева от элемента «Средний» в окрестности ключа выводит два элемента:

Add Elements to Package untitledModel (Добавить элементы в пакет неозаглавленной модели) и Revise Package Name untitledModel (Переименовать пакет неозаглавленной модели). Они показаны на рисунке 3.5.

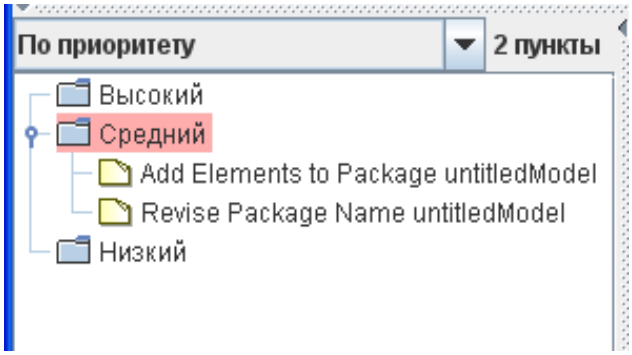

Рисунок 3.5 – Панель «To-do»

Чтобы дать имя новому еще неозаглавленному пакету необходимо щелкнуть на элементе Revise Package Name untitledModel на панели «To-do», а затем выбрать команду «Следующий» на панели деталей. В результате на панели деталей появится диалоговое окно (рисунок 3.6), в котором в области «Имя:» можно ввести название пакета. Чтобы сохранить введенное имя необходимо нажать клавишу «Закончить» на панели деталей.

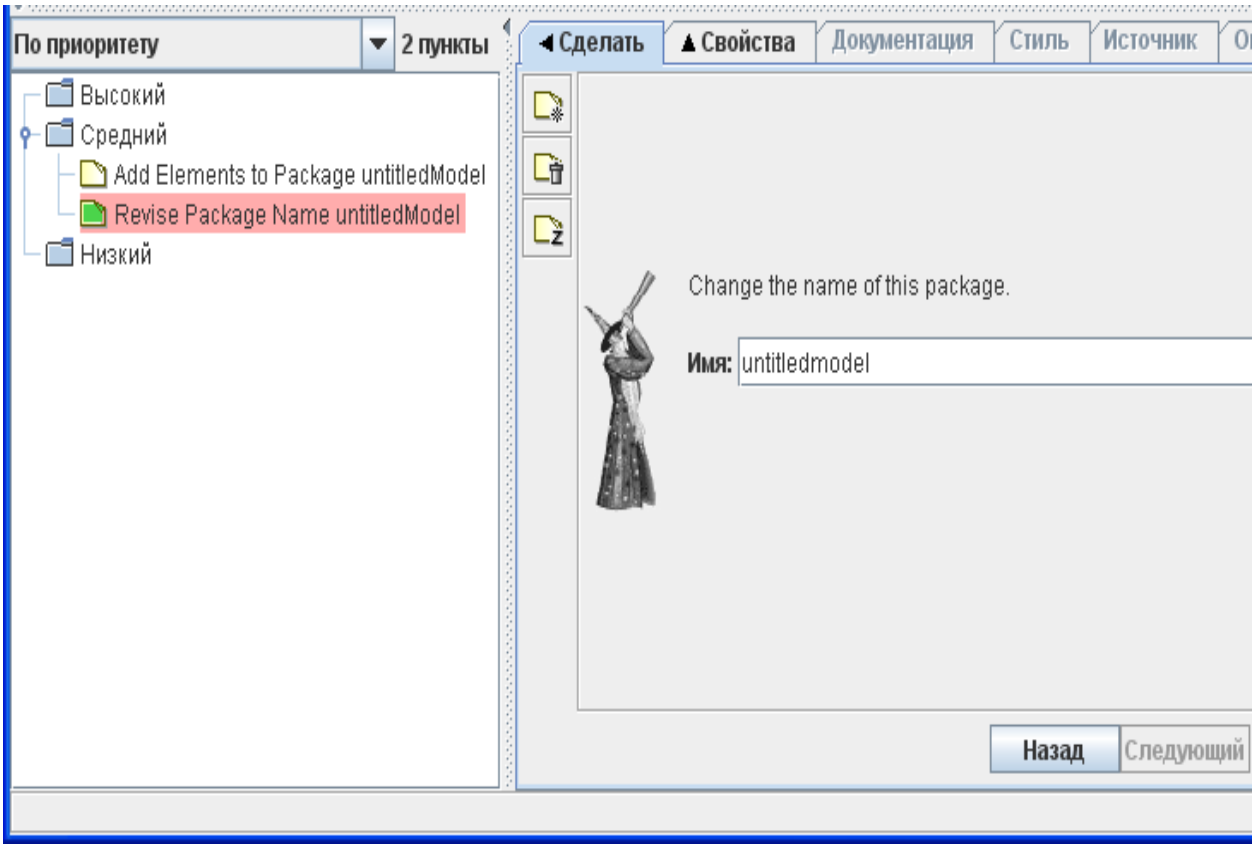

Рисунок 3.6 – Диалоговое окно, предназначенное для ввода имени нового пакета

Термин «Use Cases» применительно к диаграммам UML переводится на русский

язык как «Прецеденты» или «Варианты использования». В литературе на русском языке встречаются оба названия, соответствующие «Use Cases».

После загрузки программы ArgoUML выводится главное окно, имеющее вид, показанный на рисунке 3.1, где на панели редактирования размещены инструменты для построения диаграммы классов. Чтобы перейти к инструментам для создания диаграммы прецедентов необходимо среди команд меню «Создать диаграмму» (рисунок 3.4) выбрать команду для создания диаграммы прецедентов (вариантов использования). При этом на панели редактирования появятся инструменты диаграммы прецедентов (вариантов использования), показанные на рисунке 3.7.

Рисунок 3.7 – Панель инструментов диаграммы прецедентов

По умолчанию однократное нажатие на кнопку с изображением инструмента приводит к его однократному использованию. Для многократного использования инструмента используется двойной щелчок или клавиша CTRL одновременно с однократным щелчком. Инструменты диаграммы прецедентов описаны в таблице 3.6.

Таблица 3.6 - Инструменты диаграммы прецедентов

Инструмент	Описание
 Select	Отменяет выбор инструмента для многократного использования
 Brooom	(Метла). Перемещает сразу несколько элементов на диаграмме. Используется, чтобы быстро выстроить элементы диаграммы вдоль выбранной линии. Для включения можно использовать клавишу ALT.
	Актер. После однократного нажатия на кнопку можно добавить одного актера. После двойного щелчка или использования CTRL одновременно с однократным щелчком можно добавлять элемент на диаграмму многократно.
	Прецедент (вариант использования).
	Точка расширения. Активна только при выборе прецедента. Можно отредактировать двойным щелчком и использованием клавиатуры.
	Инструмент для добавления комментария к элементу диаграммы.
	Инструмент для связывания комментария с элементом диаграммы.

Инструмент для рисования показан на рисунке 3.8. При нажатии на кнопку появляется всплывающее меню, содержащее пиктограммы инструментов, описанные в таблице 3.7.

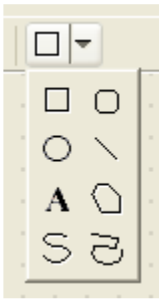

Рисунок 3.8 – Всплывающее меню для рисования

Таблица 3.7 - Инструменты для рисования диаграммы прецедентов

Инструмент	Описание
	Прямоугольник.
	Прямоугольник с закругленными краями.
	Овал.
	Линия.
	Текст.
	Многоугольник.
	Открытый сплайн. Последнюю точку отмечает двойной щелчок.
	Множество линий, проведенных через выбранные точки. Линии могут пересекаться.

Виды ассоциаций, поддерживаемые ArgoUML, показаны на рисунке 3.9. Выбрать вид ассоциации с указанием направления или без указания (простая ассоциация, агрегация или композиция) можно после нажатия на кнопку, расположенную справа от символа ассоциации «–». Если затем нажать на одну из пиктограмм, показанных на рисунке 16.3 и не соответствующих символу «–», то выбранный символ появится рядом с кнопкой выбора вместо символа ассоциации «–». Тогда на диаграмме прецедентов будет отображаться тот вид ассоциации, который выбран. Линию следует проводить от зависимого элемента к

актеру. Инструменты диаграммы прецедентов для выбора ассоциаций описаны в таблице 3.8.

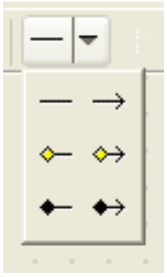

Рисунок 3.9 – Инструмент для выбора типа ассоциации ArgoUML

Таблица 3.8 - Инструменты для выбора ассоциаций диаграммы прецедентов

Инструмент	Описание
	Зависимость
	Обобщение. Проводится от ребенка к родителю.
	Расширение. Проводится от дополнительного элемента к основному.
	Включение

Для построения диаграммы прецедентов рассмотрим некоторые виды работы, выполняемой деканатом. Изобразим на диаграмме актера, щелкнув один раз на изображении актера на панели инструментов, а затем на панели редактирования. Получим изображение актера внутри прямоугольника – рисунок 3.10. Сразу, не перемещая указатель мыши за пределы прямоугольника, дадим актеру название: «деканат», набрав его на клавиатуре. Созданный на панели редактирования элемент (в данном случае актер) появляется и в древовидной структуре навигационной панели.

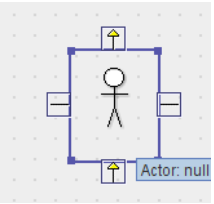

Рисунок 3.10 – Добавление актера на диаграмму прецедентов

Затем добавим прецеденты (варианты использования) и ассоциации. Эти элементы добавляются так же, как и актеры с помощью панели инструментов (рисунок 3.7). Полученная диаграмма прецедентов показана на рисунке 3.11.

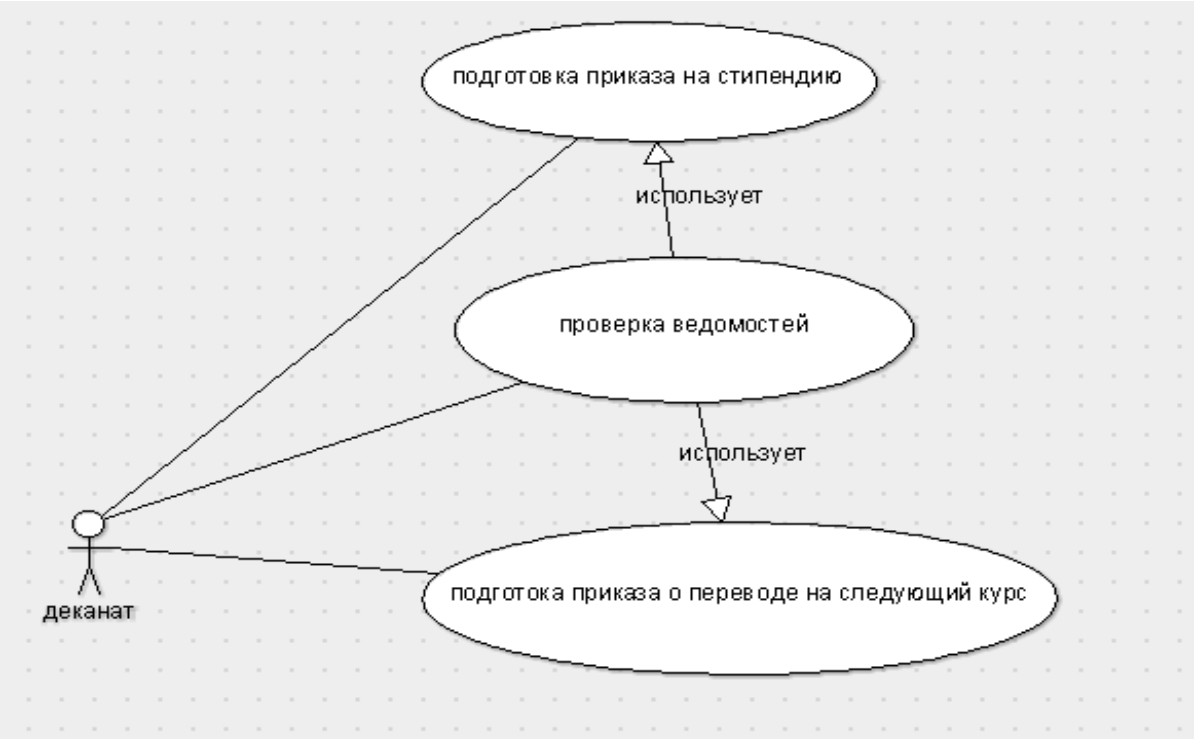

Рисунок 3.11 – Диаграмма прецедентов

Присваивать имена диаграммам, элементам диаграмм, а также редактировать названия элементов можно в окне «Имя» на панели деталей. Добавлять элементы на панель редактирования можно и без использования панели инструментов диаграммы прецедентов, которая показана на рисунке 3.7. Если подходящие элементы (актеры, прецеденты и ассоциации) были ранее созданы для данной диаграммы или для других диаграмм текущего проекта, то они перечислены на навигационной панели. На текущую диаграмму их можно скопировать, перетаскивая мышью на панель редактирования с навигационной панели. Выделение элемента диаграммы на панели редактирования сопровождается его выделением на навигационной панели и на панели деталей. На рисунке 3.12 на панели редактирования показана диаграмма прецедентов для студента, на которой выделен

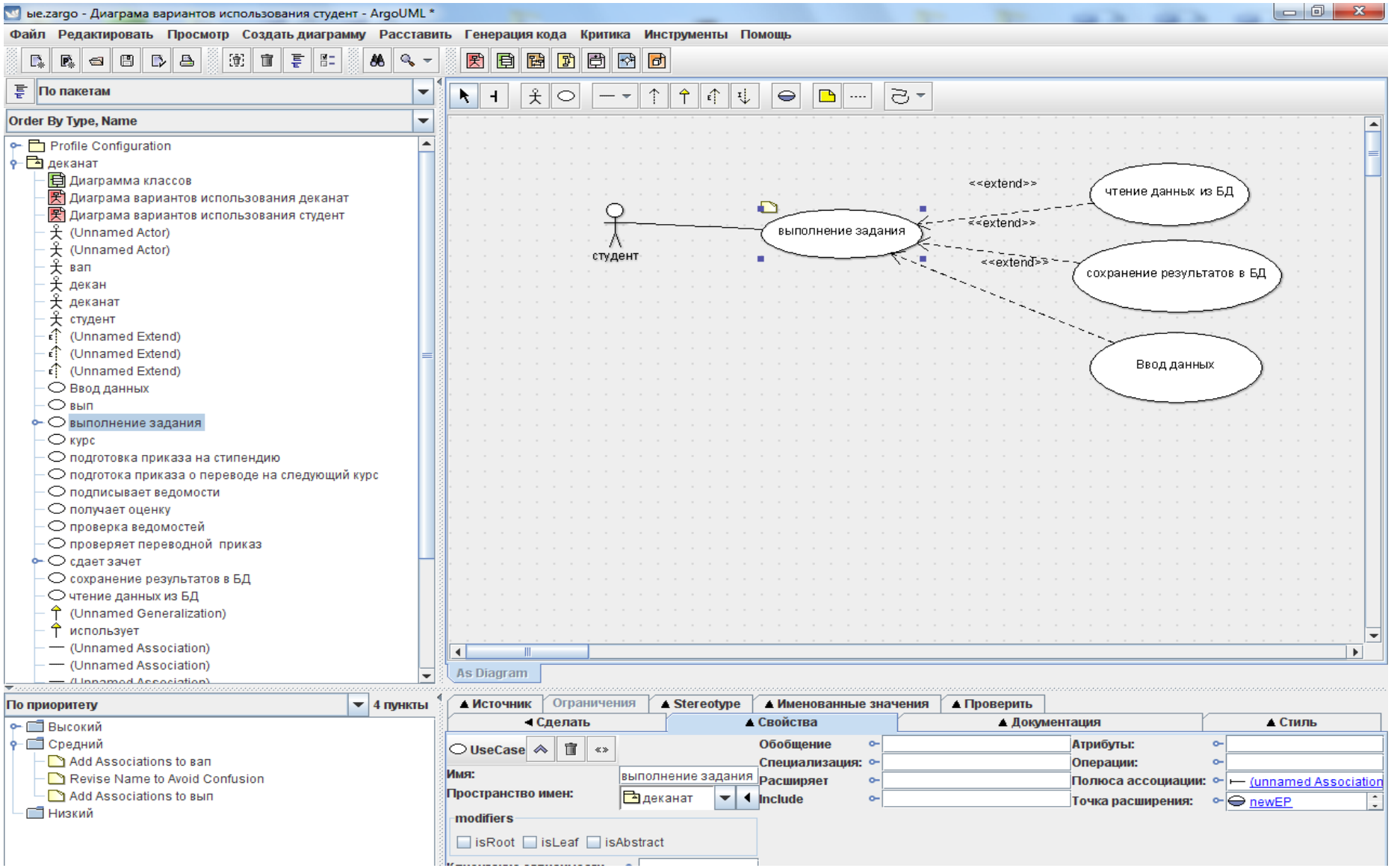

Рисунок 3.12 – Выделение прецедента «выполнение задания» на диаграмме прецедентов

прецедент «выполнение задания». Имя этого элемента одновременно выделено на навигационной панели и отображено в окне «Имя» на панели деталей.

Прецедент «выполнение задания» связан связями «расширение» с прецедентами, которые определяют дополнительные варианты использования.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать диаграмму прецедентов и описание использованных ассоциаций ArgoUML.

Задание 1. Изучить интерфейс программы ArgoUML и ее основные команды.

Задание 2. Изучить принципы объектно-ориентированного представления программных систем по материалам лекций.

Задание 3. Необходимо изучить предметную область, соответствующую вашему варианту задания, которое представлено в таблице 1.1. Выделите актеров (не менее трех) и прецеденты. Для них постройте диаграмму прецедентов.

Вопросы для защиты работы

1. Какие известные стандартные команды реализованы в ArgoUML?
2. Какие диаграммы UML можно построить с помощью ArgoUML?
3. Можно ли с помощью ArgoUML получить описание контента в виде гипертекста?
4. Какие команды реализованы в панели инструментов диаграммы прецедентов ArgoUML?
5. Какие виды ассоциаций можно показать на диаграмме прецедентов с помощью ArgoUML?
6. Каким образом можно изменить имя элемента на диаграмме прецедентов?

Лабораторная работа № 4

ДИАГРАММЫ КЛАССОВ

Цель и содержание работы: изучить порядок построения диаграммы классов с помощью ArgoUML.

Формируемые компетенции: ПК-2, ПК-4, ПК-8

Теоретическое обоснование

Инструменты для построения диаграммы классов (на панели редактирования) показаны на рисунке 4.1. Описание инструментов для построения диаграммы классов приведено в таблице 4.1.

Рисунок 4.1 – Инструменты для построения диаграммы классов

Таблица 4.1 - Инструменты диаграммы классов

Инструмент	Описание
	Пакет
	Класс
	Ассоциация. Может быть двунаправленной.
	Агрегация. Может быть двунаправленной.
	Композиция. Может быть двунаправленной.
	Конец ассоциации. Для добавления другого конца к уже существующей ассоциации (с помощью левой кнопки проводится от центра ассоциации к классу или наоборот). Применяется для создания N-арных ассоциаций.
	Обобщение. Проводится от ребенка к родителю.
	Добавление интерфейса на диаграмму. Для удобства, когда указатель мыши находится над выбранным интерфейсом, на нем отображается ручка, за которую можно перетаскивать элемент
	Реализация. Добавляет реализацию между классом и интерфейсом с помощью перемещения левой кнопки мыши. Проводится от класса к интерфейсу.
	Зависимость. Добавляет зависимость между двумя элементами с помощью перемещения левой кнопки мыши. Проводится от

	зависимого элемента. Существует 2 типа зависимости: разрешение (по умолчанию) и использование.
	Атрибут. Элемент активен только, когда выбранный элемент – класс.
	Операция. По умолчанию получает имя newOperation, которое можно редактировать.
	Association Class. Ассоциированный класс.
 Datatype.	Тип данных. Для удобства, когда указатель мыши находится над выбранным элементом, появляются ручки наверху и в основании, по которым можно щелкнуть или тянуть, чтобы сформировать необходимый элемент. Есть 2 доступных элемента - Перечисление и Стереотип.

Рассмотрим пример, связанный с деятельностью деканата. Анализ предметной области приведет нас к выделению следующих классов:

1. Факультет.
2. Кафедра.
3. Курс.
4. Семестр.
5. Куратор.
6. Студент.
7. Секретарь.

Определение атрибутов

Следующий шаг состоит в определении атрибутов (свойств) каждого класса. Например, класс «Студент» может иметь свойства: полное имя, дата рождения и номер паспорта. Класс «Курс» может иметь свойства: Код, объем часов. Каждый атрибут будет иметь свой тип данных, который определяет, как данные будут храниться в нем, строковые (символьные) или числовые.

Определение функций (операций)

Функции (операции) представляют собой действия, которые может выполнить сущность, например: студент может зарегистрироваться на дополнительный курс или на ДПВ (дисциплину по выбору). Каждая функция может иметь несколько входных параметров и возвращать только один объект.

Определение отношений между классами (ассоциации)

Для любой предметной области необходимо проанализировать возможные

отношения между сущностями. Для предметной области «Деканат» существуют следующие отношения между сущностями: в состав факультета входит много кафедр и за каждой кафедрой закреплено более одного курса. Эти отношения можно представить ассоциациями с помощью языка UML.

Ассоциация определяет связь между двумя классами, причем на каждом конце ассоциации определяется кратность. Например, тип отношения факультет – кафедра соответствует связи 1:M, что изображается на диаграмме как "1 .. *

Построение диаграммы классов

Чтобы добавить новый класс нужно нажать на пиктограмму  на панели инструментов, которая показана на рисунке 4.1. После этого указатель мыши покажет фигуру, имеющую форму крестика, которую можно перемещать по панели редактирования. После перемещения курсора (в форме крестика) в то место, где нужно разместить новый класс, необходимо щелкнуть левой кнопкой. После этого на панели редактирования появится фигура, изображающая класс (рисунок 4.2). Красной волнистой линией подчеркнуто место, предназначенное для ввода имени класса. Класс разделен на три части: верхняя часть будет содержать имя класса, средняя часть будет содержать атрибуты класса, нижняя часть – операции (или функции класса).

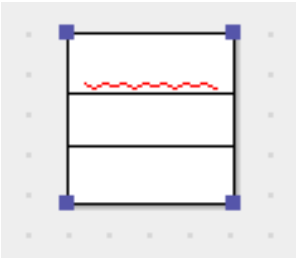

Рисунок 4.2 – Добавление нового класса на диаграмму

Щелкнув указателем на красной волнистой линии можно добавить на диаграмму имя класса. Имена атрибутов можно добавить с помощью в помощью всплывающего меню, показанного на рисунке 4.3. Тип атрибута можно выбрать из ниспадающего списка на панели деталей.

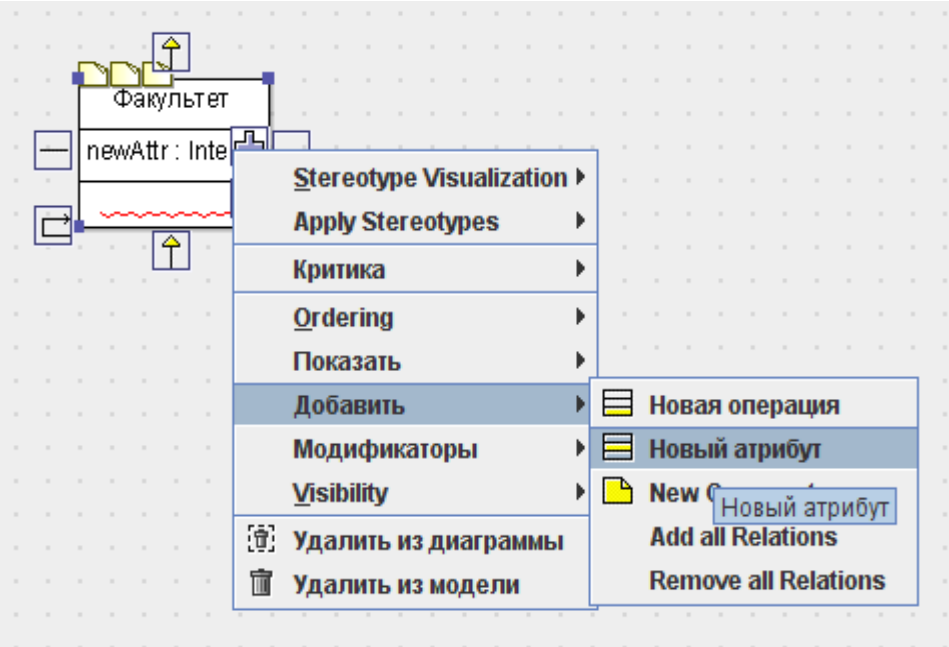

Рисунок 4.3 – Добавление атрибутов класса на диаграмму

Добавление новой операции показано на рисунке 4.4. Имя и тип параметра операции можно указать в круглых скобках.

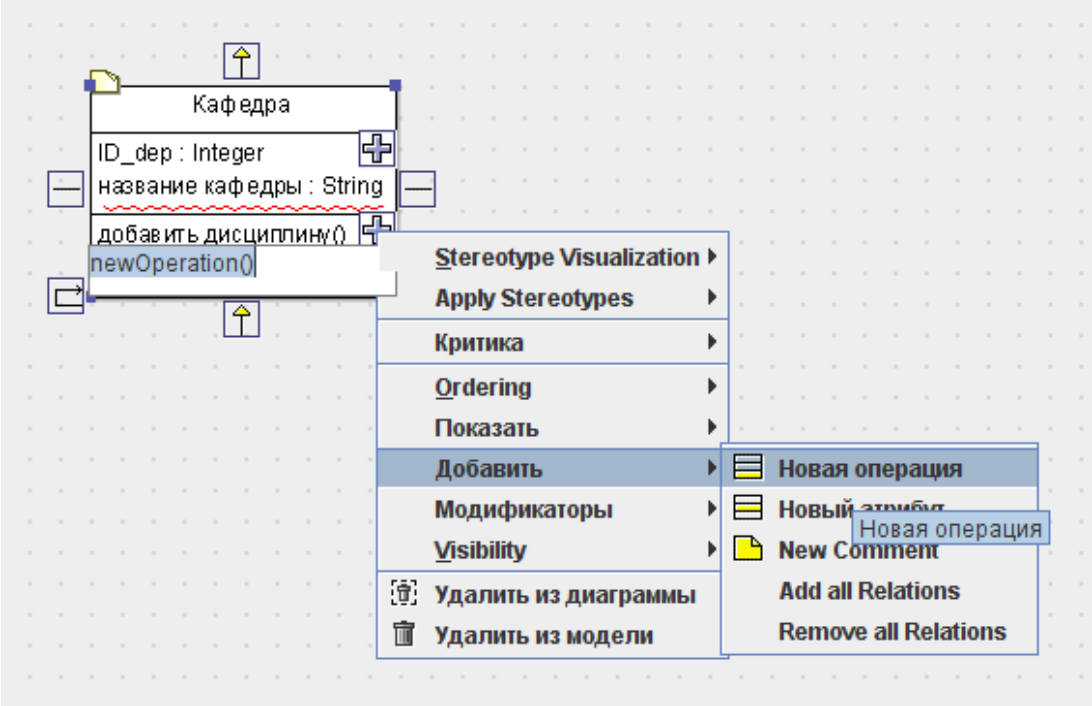

Рисунок 4.4 – Добавление новой операции на диаграмму классов

После создания двух классов можно добавить ассоциацию с команды меню и таких же действий, как и при добавлении ассоциаций на диаграмму прецедентов. Указание кратности (множественности) ассоциации показано на рисунке 4.5. Возможные виды кратности показаны на ниспадающем меню команды «Множественность».

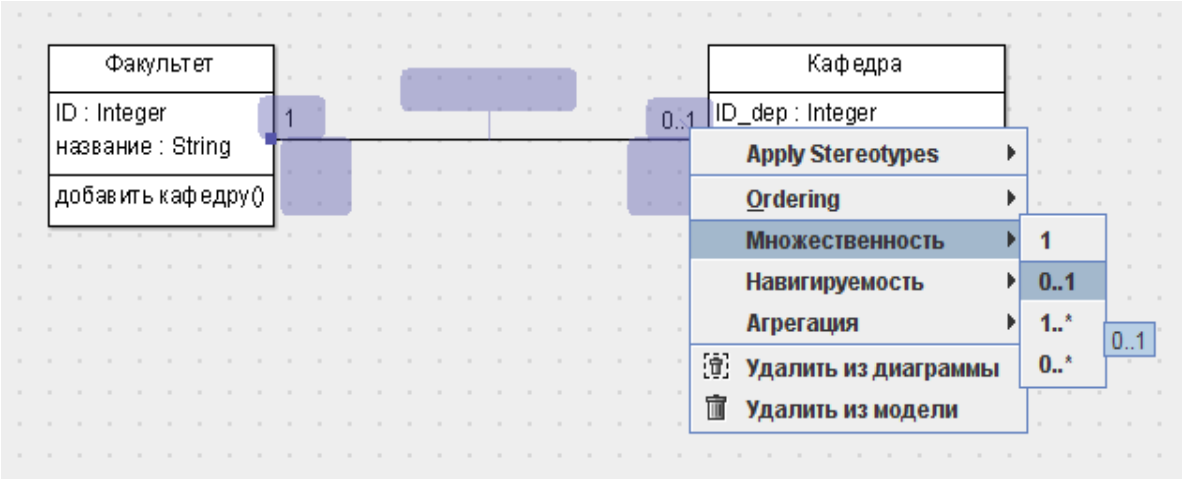

Рисунок 4.5 – Указание кратности ассоциации на диаграмме классов

Чтобы указать кратность ассоциации с другой стороны необходимо подвести курсор к выделенному на границе ассоциации цветному прямоугольнику, дважды щелкнуть левой кнопкой мыши и ввести с клавиатуры. Полученная диаграмма должна выглядеть как на рисунке 4.6.

Рисунок 4. 6 – Диаграмма классов

Задание 1. Необходимо изучить предметную область, соответствующую вашему варианту задания, которое представлено в таблице 1.1. Выделите классы (не менее трех), опишите их атрибуты и операции. Для них постройте диаграмму классов.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать диаграмму классов и описание использованных ассоциаций ArgoUML.

Вопросы для защиты работы

1. Какие команды реализованы в панели инструментов диаграммы классов ArgoUML?
2. Какие виды ассоциаций можно показать на диаграмме классов с помощью ArgoUML?
3. Каким образом можно изменить имя элемента на диаграмме классов?
4. Каким образом можно добавлять атрибуты и операции на диаграмму классов?

Лабораторная работа № 5

ДИАГРАММА ПОСЛЕДОВАТЕЛЬНОСТИ

Цель и содержание работы: изучить порядок построения диаграммы последовательности с помощью ArgoUML.

Формируемые компетенции: ПК-2, ПК-4, ПК-8

Теоретическое обоснование

На диаграмме последовательности изображаются только те объекты, которые непосредственно участвуют во взаимодействии, обмениваясь сообщениями (вызывая методы) в определенной хронологической последовательности. Их используют для моделирования специфических, ограниченных по времени, ситуаций. Диаграммы последовательностей специально выделяют порядок и времена отсылки сообщений объектам. Ключевым моментом для диаграмм последовательности является динамика взаимодействия объектов во времени. В качестве объектов на диаграммах последовательности в ArgoUML можно использовать как актеров, так и экземпляры классов. Внешний вид диаграммы последовательности показан на рисунке 5.1.

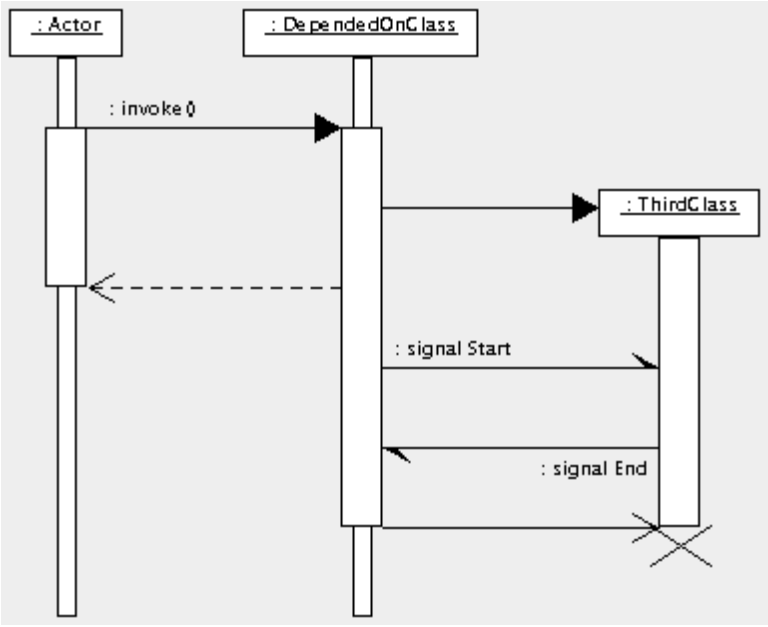

Рисунок 5.1 – Диаграмма последовательности

В UML на диаграмме последовательности крайним слева изображается объект, который является инициатором взаимодействия. Правее изображается другой объект, который непосредственно взаимодействует с первым. Таким образом, все объекты на диаграмме последовательности размещаются по порядку слева направо в соответствии с очередностью или степенью активности при взаимодействии объектов друг с другом.

Графически каждый объект изображается прямоугольником и располагается в верхней части своей линии жизни (вертикальная линия). Внутри прямоугольника записываются имя объекта [:имя класса]. Временная ось направлена сверху вниз. Масштаб на оси времени не указывается, так как диаграмма последовательности моделирует лишь хронологическую упорядоченность взаимодействий типа «раньше-позже». Сообщения, посылаемые от одного объекта к другому, отображаются стрелками с указанием операции и параметров.

Линия жизни служит для обозначения периода времени, в течение которого объект существует в системе и может участвовать во всех ее взаимодействиях. Если объект существует в системе постоянно, то и его линия жизни должна продолжаться по всей плоскости диаграммы последовательности от самой верхней ее части до самой нижней. Линия жизни изображается тонкой (или штриховой) линией, когда объект пассивен, и жирной линией, если объект активен.

Отдельные объекты, выполнившие свою работу в системе, могут быть уничтожены, чтобы освободить занимаемые ими ресурсы. Для таких объектов линия жизни обрывается в момент его уничтожения. Для обозначения момента уничтожения объекта в языке UML используется специальный символ в форме латинской буквы «X». Ниже этого символа линия жизни не изображается, поскольку соответствующего объекта в системе уже нет, и этот объект должен быть исключен из всех последующих взаимодействий.

Не обязательно создавать все объекты диаграммы в начальный момент времени. Отдельные объекты могут создаваться по мере необходимости, экономя ресурсы системы и повышая ее производительность. В этом случае прямоугольник объекта изображается не в верхней части диаграммы последовательности, а в той части, которая соответствует моменту создания объекта. При этом прямоугольник объекта вертикально располагается в том месте диаграммы, которое по оси времени совпадает с моментом его создания в системе. Объект обязательно создается со своей линией жизни.

Панель инструментов для построения диаграммы последовательности показана на рисунке 5.2, а описание отдельных инструментов приведено в таблице 5.1.

Рисунок 5.2 – Инструменты для построения диаграммы последовательности

Таблица 5.1 – Описание инструментов диаграммы последовательности

Инструмент	Описание
	Добавление нового объекта на диаграмму

	Двухсторонние сообщения о вызове процедур, выполнении операций
	Сообщение об отправке сообщения (одностороннее)
	Сообщение о возврате из вызова процедуры. Примером может служить простое сообщение о завершении некоторых вычислений без предоставления результата расчетов объекту-клиенту.
	Действие создания
	Действие уничтожения

В качестве примера для предметной области, связанной с работой деканата, построим диаграмму последовательности для варианта использования «Записать студента на изучение ДПВ» - рисунок 5.3.

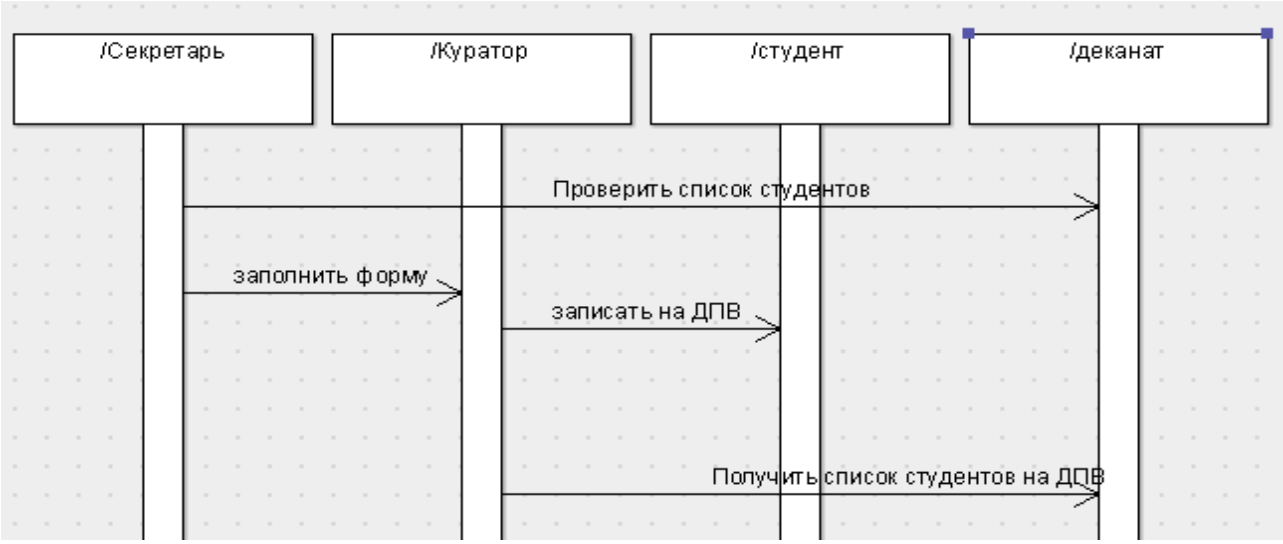

Рисунок 5.3 – Диаграмма последовательности

Задание 1. Необходимо изучить предметную область, соответствующую вашему варианту задания, которое представлено в таблице 1.1. Выделите классы (не менее трех), актеров. Опишите объекты и последовательность их взаимодействия с помощью диаграммы последовательности (для двух вариантов использования).

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать диаграммы последовательности для двух вариантов использования.

Вопросы для защиты работы

1. Какие команды реализованы в панели инструментов диаграммы последовательности ArgoUML?
2. Какие виды сообщений можно показать на диаграмме последовательности?
3. Каким образом отображается линия жизни объекта?

Лабораторная работа № 6

ДИАГРАММЫ СОТРУДНИЧЕСТВА (КООПЕРАЦИИ)

Цель и содержание работы: изучить порядок построения диаграммы сотрудничества с помощью ArgoUML.

Формируемые компетенции: ПК-2, ПК-4, ПК-8

Теоретическое обоснование

Диаграммы последовательности и кооперации (collaboration) отображают взаимодействие элементов моделируемой предметной области. Диаграммы последовательности представляют временные аспекты взаимодействия и хронологическую последовательность действий, а диаграммы кооперации представляют структурные аспекты взаимодействия. На диаграмме кооперации изображаются только те отношения между объектами, которые играют определенные роли во взаимодействии. Представление структуры системы как совокупности взаимодействующих объектов на диаграмме кооперации соответствует статическому описанию системы.

Классификатор (Classifier) – это механизм, описывающий структурные и поведенческие свойства. Классификаторами называются те элементы моделирования, которые могут иметь экземпляры. Структурные свойства классификаторов характеризуются с помощью атрибутов, а поведенческие свойства выражаются в форме операций. Все экземпляры одного классификатора обладают рядом общих свойств. К числу классификаторов относятся классы, интерфейсы, типы данных, сигналы, компоненты, узлы, прецеденты и подсистемы. Несмотря на различие классификаторов между собой в ArgoUML на диаграммах последовательности и кооперации классификаторы изображаются с помощью одной и той же пиктограммы – прямоугольника. Строка текста в прямоугольнике должна иметь следующий формат:

/ <Имя роли классификатора>: <Имя классификатора>

В лабораторной работе 18 классификатор описан как объект.

Кооперация может быть представлена на двух уровнях:

- на уровне спецификации (показывает роли классификаторов и роли ассоциаций в рассматриваемом взаимодействии);
- на уровне примеров (указывает экземпляры и связи, образующие отдельные роли в кооперации).

Диаграмма кооперации уровня спецификации показывает роли, которые играют участвующие во взаимодействии элементы. Элементами кооперации на этом уровне

являются классы и ассоциации, которые обозначают отдельные роли классификаторов и ассоциации между участниками кооперации. Пример кооперации показан на рисунке 6.1.

Рисунок 6.1 – Диаграмма кооперации

Панель инструментов для построения диаграммы кооперации показана на рисунке 6.2, а описание отдельных инструментов приведено в таблице 6.1.

Рисунок 6.2 – Инструменты для построения диаграммы кооперации

Таблица 6.1 – Описание инструментов диаграммы кооперации

Инструмент	Описание
	Роль классификатора
	Роль ассоциации
	Обобщение
	Зависимость
	Добавить сообщение

На рисунке 6.3 показано отношение обобщения между отдельными кооперациями уровня спецификации.

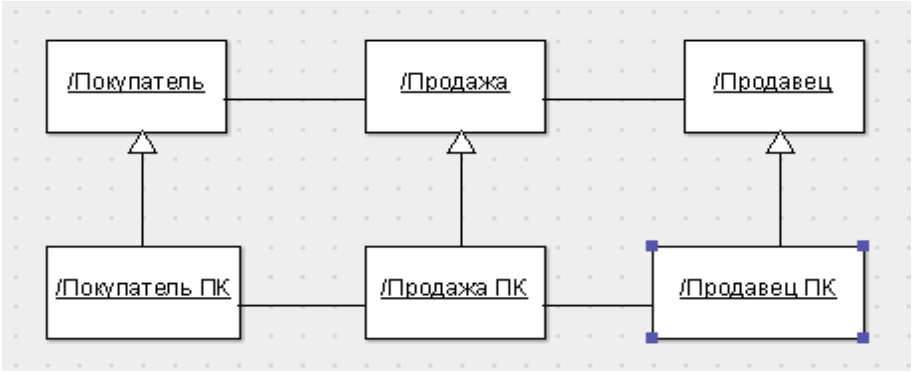

Рисунок 6.3 – Диаграмма кооперации с отношениями обобщения

Отличительной особенностью диаграммы кооперации является присутствие на ней сообщений. Сообщения уже рассматривались ранее при изучении диаграммы последовательности в лабораторной работе 18. Сообщения показаны на рисунках 18.1 и

18.3 направленными стрелками. При построении диаграммы кооперации они имеют некоторые дополнительные семантические особенности. Сообщение на диаграмме кооперации специфицирует коммуникацию между двумя объектами, один из которых передает другому некоторую информацию. При этом первый объект ожидает, что после получения сообщения вторым объектом последует выполнение некоторого действия. Таким образом, именно сообщение является причиной или стимулом для начала выполнения операций, отправки сигналов, создания и уничтожения отдельных объектов. Связь обеспечивает канал для направленной передачи сообщений между объектами от объекта-источника к объекту-получателю. Сообщения в языке UML также специфицируют роли, которые играют объекты - отправитель и получатель сообщения. Сообщения на диаграмме кооперации изображаются помеченными стрелками рядом (выше или ниже) с соответствующей связью или ролью ассоциации. Направление стрелки указывает на получателя сообщения. Сообщениям на диаграмме кооперации в среде ArgoUML автоматически присваиваются номера. На рисунке 6.4 показана диаграмма кооперации с сообщением.

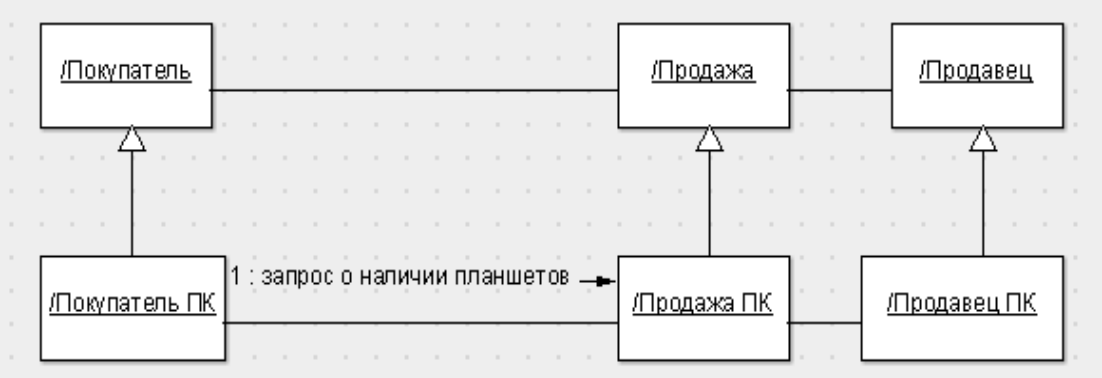

Рисунок 6.4 – Диаграмма кооперации

Задание 1. Построить диаграмму кооперации для той же предметной области, для которой в лабораторной работе 5 была построена диаграмма последовательности. Сравнить диаграммы последовательности и кооперации.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать диаграммы кооперации для двух примеров.

Вопросы для защиты работы

1. Какие команды реализованы в панели инструментов диаграммы кооперации ArgoUML?
2. Какие виды сообщений можно показать на диаграмме кооперации?
3. Что общего у диаграмм последовательности и кооперации?
4. Чем отличаются диаграммы последовательности и кооперации?

Лабораторная работа № 7

ДИАГРАММЫ СОСТОЯНИЙ

Цель и содержание работы: изучить порядок построения диаграммы состояний с помощью ArgoUML.

Формируемые компетенции: ПК-2, ПК-4, ПК-8

Теоретическое обоснование

Диаграммы состояний отображают динамическое поведение сущностей, на основе спецификации их реакции на восприятие некоторых конкретных событий. Системы, которые реагируют на внешние действия, совершаемые пользователями или другими системами, иногда называют реактивными. Если такие действия инициируются в произвольные случайные моменты времени, то поведение модели является асинхронным.

Диаграммы состояний можно использовать для описания поведения не только экземпляров классов (объектов), но и для описания других компонентов моделей, таких как варианты использования, актеры, подсистемы, операции и методы.

Диаграмма состояний представляет собой граф, описывающий поведение некоторого автомата - математической абстракции, используемой для моделирования детерминированного поведения технических объектов или объектов реального мира.

Понятие автомата использовано и в языке UML. Вершинами графа являются состояния и некоторые другие типы элементов автомата (псевдосостояния), которые изображаются соответствующими графическими символами. Дуги графа служат для обозначения переходов из одного состояния в другое состояние. Диаграммы состояний могут быть вложены друг в друга, образуя вложенные диаграммы более детального представления отдельных элементов модели. Переход объекта из состояния в состояние рассматривается как мгновенный. Каждое последующее состояние всегда наступает позже предыдущего. Последовательность состояний автомата охватывает все этапы его жизненного цикла, от создания до уничтожения. Простейший пример диаграммы состояний для технического устройства (телефон, компьютер) показан на рисунке 7.1.

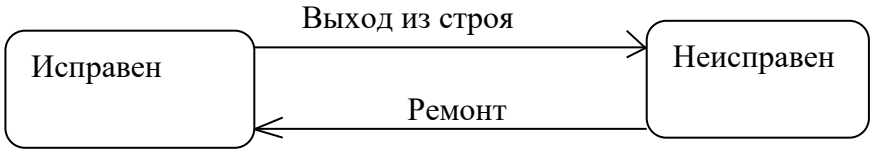

Рисунок 7.1 – Диаграмма состояний технического устройства

Понятие «автомат (state machine)» в языке UML основано на выполнении следующих обязательных условий:

1. Автомат не запоминает историю перемещения из состояния в состояние. Однако в ArgoUML имеются средства, которые дополняют возможности описания предыстории поведения объекта на основе введения в рассмотрение так называемых исторических состояний с помощью инструментов  и  , которые описаны в таблице 7.1.
2. В каждый момент времени автомат может находиться в одном и только в одном из своих состояний. Причем, если не происходит никаких событий, то автомат может находиться в отдельном состоянии как угодно долго.
3. Переходы из одного состояния в другое являются мгновенными, а длительность нахождения автомата в том или ином состоянии, а также время достижения того или иного состояния никак не специфицируются. Поэтому время на диаграмме состояний присутствует в неявном виде. Однако в ArgoUML имеются средства, которые отмечают синхронные состояния с помощью инструмента  , описанного в таблице 7.1.
4. Количество состояний автомата должно быть обязательно конечным.
5. Граф автомата не должен содержать изолированных состояний и переходов. Это условие означает, что для каждого из состояний, кроме начального, должно быть определено предшествующее состояние. Каждый переход должен обязательно соединять два состояния автомата. Допускается переход из состояния в себя, такой переход еще называют "петлей".
6. Автомат не должен содержать конфликтующих переходов, т. е. таких переходов из одного и того же состояния, когда объект одновременно может перейти в два и более последующих состояния (кроме случая параллельных подавтоматов).

Инструменты для построения диаграммы состояний показаны на рисунке 7.2, а их описание приведено в таблице 7.1.

Рисунок 7.2 – Инструменты для построения диаграммы состояний

Таблица 7.1 – Описание инструментов диаграммы состояний

Инструмент	Описание
	Простое состояние

	Композитное (составное) состояние
	Новый переход (New Tansition)
	Состояние синхронизации
 Submachine State	Состояние – ссылка на вложенный автомат
 Stub State	Состояние -заглушка
	Начальное состояние
	Конечное состояние
 Junction	Ветвление. Используется для расщепления одного входящего перехода на несколько исходящих. Выполняется тот переход, для которого в момент перехода сторожевое условие принимает значение «истина». Предопределенное условие, обозначаемое как «else», можно определить только для одного перехода, который выполняется, если для других переходов сторожевые условия принимают значение «ложь».
	Выбор (Choice). Выбор используется для расщепления входящего перехода на несколько исходящих. Выполняется тот переход, для которого в момент перехода сторожевое условие принимает значение «истина». Если для нескольких переходов условие принимает значение «истина», то выбор выполняемого перехода производится случайно. Предопределенное условие, обозначаемое как «else», можно определить только для одного перехода, который выполняется, если для других переходов сторожевые условия принимают значение «ложь».
	Разветвление
	Соединение
	Недавнее историческое состояние (shallow history state)
	Давнее историческое состояние (deep history state). Используется для запоминания всех подсостояний любого уровня вложенности для текущего подавтомата.
	Вызов события как триггера для перехода. Существует 4 типа вызовов: событие вызова; событие изменения; событие сигнала и событие времени (таблица 7.2).
	Сторожевое условие (guard condition) записывается в квадратных скобках после события-триггера и представляет собой логическое выражение. Если сторожевое условие принимает значение "истина", то соответствующий переход может сработать, в результате чего объект перейдет в целевое состояние. Если же сторожевое условие принимает значение "ложь", то переход не может сработать.
	Вызов действия, которое приводит к переходу из одного состояния в другое. Существует 7 видов: действие вызова; действие создания;

	действие уничтожения; действие возврата; действие отправки; неинтерпретируемое действие; новое последовательное действие (таблица 7.3)

Ниспадающее меню для выбора событий вызова появляется при наведении курсора на пиктограмму  . Оно содержит 4 варианта событий вызова, которые описаны в таблице 7.2.

Таблица 7.2 - Инструменты для выбора событий вызова

Инструмент	Описание
	Событие вызова
	Событие изменения
	Событие сигнала
	Событие времени

Ниспадающее меню для выбора варианта вызова действия появляется при наведении курсора на пиктограмму  . Оно содержит 7 вариантов, которые описаны в таблице 7.3.

Таблица 7.3 - Инструменты для выбора вызова действия

Инструмент	Описание
	Действие вызова
	Действие создания
	Действие уничтожения
	Действие возврата
	Действие отправки
	Конечное состояние
	Неинтерпретируемое действие
	Новое последовательное действие

Состояние (state) характеризуется значением таких свойств элементов системы,

которые отражают динамический или функциональный аспект ее поведения. Чтобы добавить новое простое состояние нужно нажать на пиктограмму  на панели инструментов, которая показана на рисунке 7.2. После перемещения курсора на панели редактирования в то место, где нужно разместить новое состояние, необходимо щелкнуть левой кнопкой. После этого на панели редактирования появится фигура, изображающая простое состояние (рисунок 7.3). Красной волнистой линией подчеркнуто место, предназначенное для ввода имени состояния. Состояние разделено горизонтальной линией на две части: верхняя часть будет содержать имя состояния, нижняя часть – список его внутренних действий.

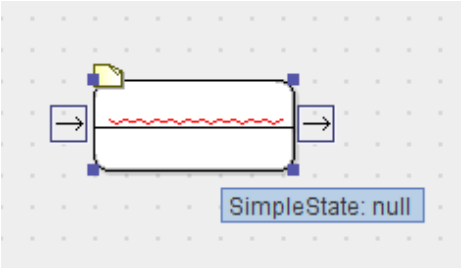

Рисунок 7.3 – Добавление нового простого состояния на диаграмму

После двойного щелчка указателем на красной волнистой линии можно добавить на диаграмму имя состояния. Имена внутренних действий: entry (входное), do (выполняемое), exit (выходное) можно добавить с помощью инструментов «Действие при входе», «Действие при выходе» и «Деятельность выполнения», расположенных на панели деталей, – рисунок 7.4. Тип действий можно выбрать из ниспадающего списка на панели деталей после нажатия на пиктограмму  .

Названия внутренних действий вводятся после двойного щелчка в области списка внутренних действий в нижней части прямоугольника, изображающего простое состояние, на панели редактирования.

Пример диаграммы состояний для примера, описывающего работу электронной почты, представлен на рисунке 7.5.

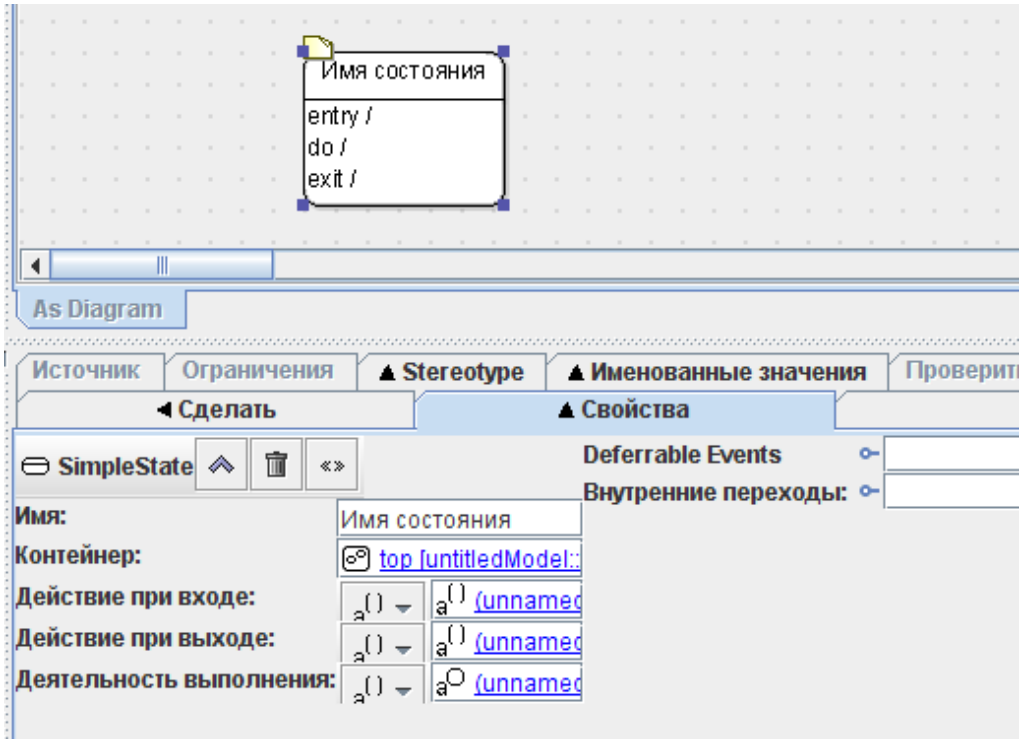

Рисунок 7.4 – Добавление списка внутренних действий простого состояния

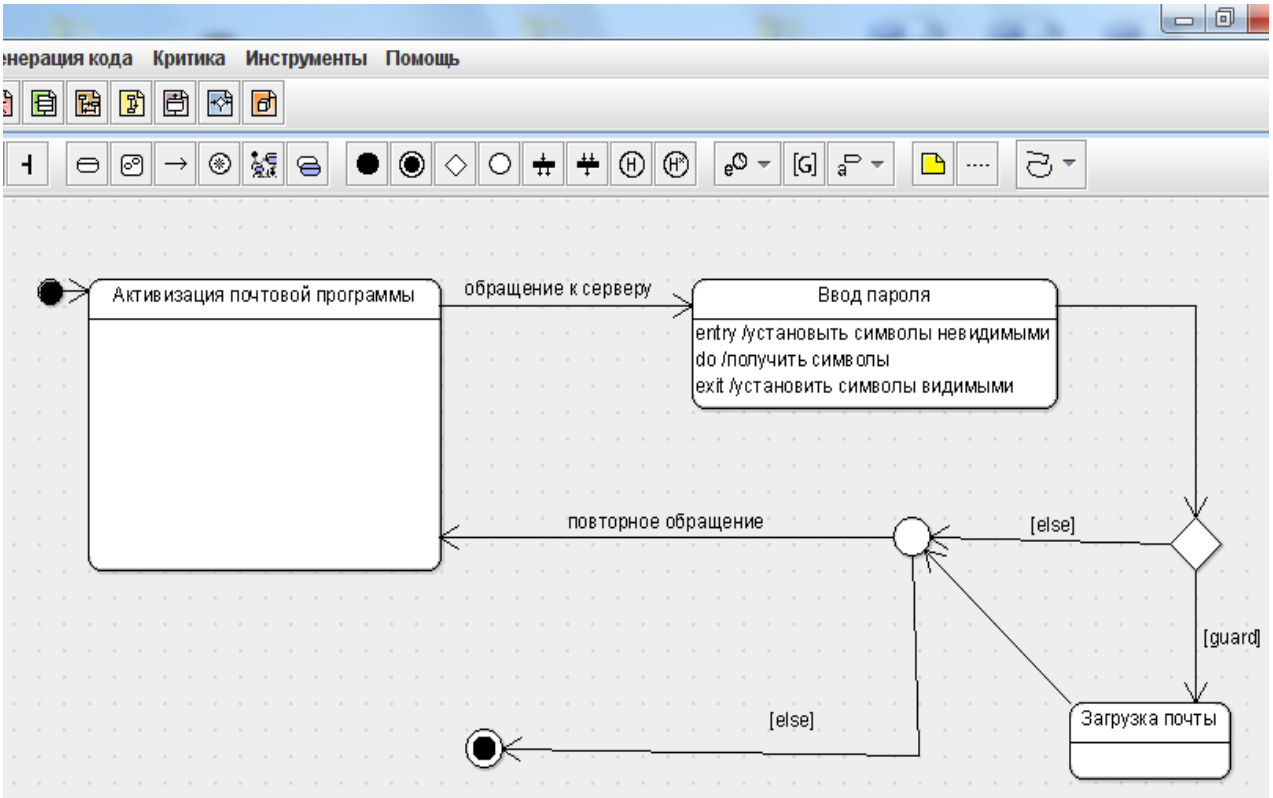

Рисунок 7.5 – Диаграмма состояний

На диаграмме (рисунок 7.5) показано начальное состояние, соответствующее пиктограмме  , три простых состояния «Активизация почтовой программы», «Ввод пароля», «Загрузка почты» и конечное состояние, соответствующее пиктограмме  .

Пиктограммой  на диаграмме изображен инструмент (ветвление), описанный в таблице 7.1. Он использован для расщепления перехода, исходящего от состояния «Ввод пароля», на два перехода. Один из них, направленный к состоянию «Загрузка почты», выполняется, если сторожевое условие [пароль введен верно] принимает значение «истина». Этот переход на диаграмме обозначен [guard]. Предопределенное условие, обозначаемое как «else», выполняется, если сторожевое условие [пароль введен верно] принимает значение «ложь».

Пиктограммой «Выбор»  на диаграмме изображен инструмент, описанный в таблице 7.1. Он использован для выбора либо повторного обращения к почтовой программе, либо для выхода.

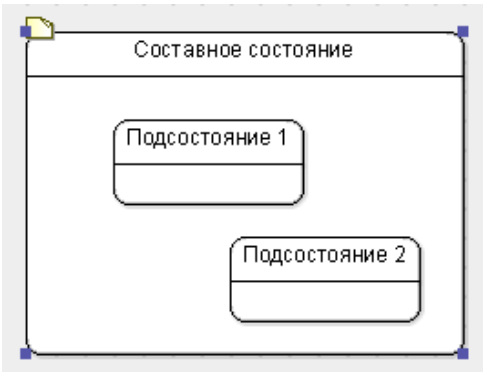

Рисунок 7.6 – Составное состояние

Составное состояние (composite state) создается с помощью инструмента, описанного в таблице 7.1. Это такое состояние, внутри которого находятся вложенные в него подсостояния (substate). Между составным состоянием и подсостояниями имеет место отношение композиции.

Подсостояния могут быть последовательными и параллельными. Чтобы построить на диаграмме параллельные подсостояния, необходимо область внутри составного состояния разделить горизонтальной штриховой линией на горизонтальные полосы, т.е. выделить новый конкурентный участок. Для этого поместив курсор в область, занятую прямоугольником составного состояния, нужно щелкнуть правой кнопкой мыши и в ниспадающем меню (рисунок 7.7) выбрать команду «New Concurrent Region».

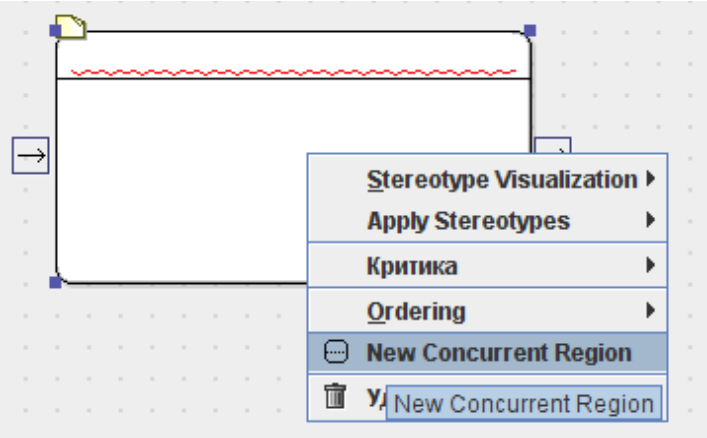

Рисунок 7.7 – Выбор команды «New Concurrent Region»

На рисунке 7.8 область внутри составного состояния разделена штриховой линией на два параллельных процесса.

Параллельные подсостояния (concurrent substates) позволяют специфицировать два и более подавтомата, которые могут выполняться параллельно внутри составного события. Параллельные подсостояния в свою очередь могут состоять из последовательных. Например, Подсостояние 1 и Подсостояние 2 на рисунке 7.8 являются последовательными подсостояниями подавтомата 1. Для подавтомата 2 последовательные подсостояния – это 3 и 4.

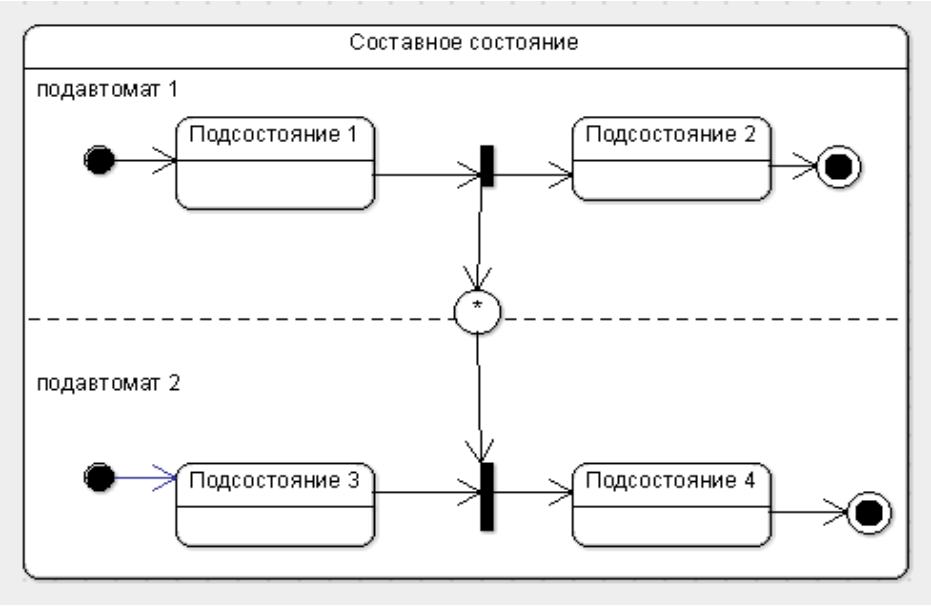

Рисунок 7.8 – Диаграмма состояний, содержащая параллельные подавтоматы

Для учета синхронизации наступления отдельных событий в языке UML имеется специальное псевдосостояние, которое называется синхронизирующим состоянием. На диаграмме показано, что переходы между Подсостоянием 1 и Подсостоянием 2 подавтомата 1 и между Подсостоянием 3 и Подсостоянием 4 подавтомата 2

синхронизированы (символ  на пересечении перехода и штриховой линии).

Задание 1. Необходимо изучить предметную область, соответствующую вашему варианту задания, которое представлено в таблице 1.1. Выделите состояния, характерные для предметной области, проанализируйте возможные переходы между состояниями и события, которые приводят к переходам состояний. Проанализируйте возможность реализации составных состояний с последовательными и параллельными переходами. Постройте диаграмму состояний.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать самостоятельно построенную диаграмму состояний.

Вопросы для защиты работы

1. Какие команды реализованы в панели инструментов диаграммы состояний ArgoUML?
2. Какие виды состояний можно показать на диаграмме состояний с помощью ArgoUML?
3. Каким образом можно разделить переход на несколько исходящих?

Лабораторная работа № 8

ДИАГРАММЫ ДЕЯТЕЛЬНОСТИ

Цель и содержание работы: изучить порядок построения диаграммы деятельности с помощью ArgoUML.

Формируемые компетенции: ПК-2, ПК-4, ПК-8

Теоретическое обоснование

Диаграммы деятельности (activity diagram) отображают особенности алгоритмической и логической реализации выполняемых системой операций. При структурном подходе к разработке информационных систем для отражения алгоритмических решений традиционно использовались блок-схемы алгоритмов.

Под *деятельностью* в данном случае понимают задачу (операцию), которую необходимо выполнить вручную или с помощью средств автоматизации. Каждому варианту использования соответствует своя последовательность задач. В теоретическом плане диаграммы деятельности являются обобщенным представлением алгоритма, реализующего анализируемый вариант использования. На диаграмме деятельность обозначается прямоугольником с закругленными углами (рисунок 8.1, а).

Диаграммы деятельностей позволяют описывать альтернативные и параллельные процессы. Для обозначения альтернативных процессов используют ромб (рисунок 8.1, б), условие указывают над ним слева или справа, а альтернативы «да», «нет» - рядом с соответствующими выходами. С помощью этого же блока можно построить циклический процесс. Множественность активации деятельности обозначают символом «*», помещенным рядом со стрелкой активации деятельности, и при необходимости уточняют надписью вида «для каждой строки». Для обозначения параллельных процессов используют линейки синхронизации (рисунок 8.1, в), причем условие синхронизации можно уточнить, указав его на диаграмме.

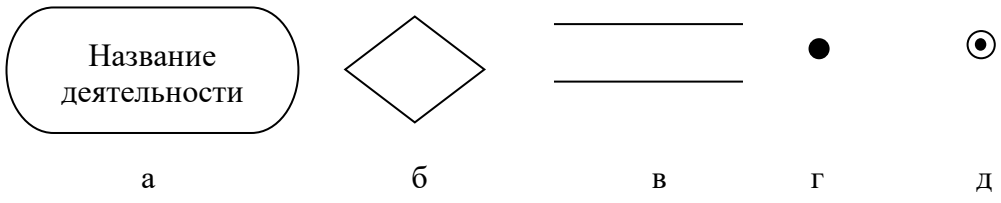

Рисунок 8.4 – Условные обозначения диаграммы деятельности:
а – деятельность; б – выбор; в – линейная синхронизация; г – начало; д – конец

На рисунке 8.2 показано, что «Деятельность 1» и «Деятельность 2» могут

выполняться параллельно. На этапе определения спецификаций имеет смысл уточнять только варианты использования, краткое описание которых недостаточно для понимания сущности решаемых проблем. Диаграммы деятельностей, таким образом, можно использовать вместо описания вариантов использования или как дополнение к ним.

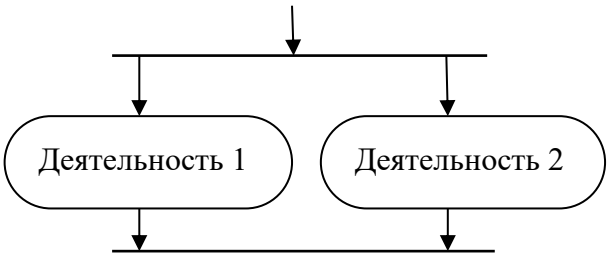

Рисунок 8.2 – Диаграмма деятельности с указанием параллельности процессов

Диаграммы деятельности описывают последовательность выполняемых операций и отображают особенности процедурного и синхронного управления. Их можно считать частным случаем диаграмм состояний. В ArgoUML панель инструментов для построения диаграммы деятельности (рисунок 8.3) имеет сходство с панелью инструментов для построения диаграммы состояний (рисунок 14.1). На обеих диаграммах присутствуют обозначения состояний, переходов, сторожевого условия, ветвления, начального и конечного состояний. Отличие заключается в семантике состояний, которые используются для представления действий. Каждое состояние деятельности соответствует выполнению некоторой элементарной операции, а переход в следующее состояние на диаграмме деятельности происходит после завершения операции в предыдущем состоянии.

Рисунок 8.3 – Панель инструментов для построения диаграммы деятельности

В таблице 8.1 описаны только те инструменты для построения диаграммы деятельности, которые отличаются от инструментов для построения диаграммы состояний.

Таблица 8.1 – Инструменты для построения диаграммы деятельности

Инструмент	Описание
	Новое состояние деятельности
	Дорожки (swimlanes). Применяется для разделения состояния действия на диаграмме деятельности на отдельные группы, которые отделяются друг от друга вертикальными линиями.
	Вызов состояния
	Объект перехода между состояниями

В зависимости от степени детализации диаграммы деятельностей так же, как

диаграммы классов, используют на разных этапах разработки. На этапе анализа требований и уточнения спецификаций диаграммы деятельностей позволяют конкретизировать основные функции разрабатываемого программного обеспечения.

Графически диаграмма деятельности представляется в форме графа деятельности, вершинами которого являются состояния действия, а дугами - переходы от одного состояния действия к другому. Простейшая диаграмма деятельности показана на рисунке 8.4.

Рисунок 8.4 – Диаграмма деятельности

Состояние деятельности изображается фигурой, ограниченной параллельными горизонтальными линиями и боковыми дугами. Для добавления нового состояния деятельности используется инструмент с пиктограммой  .

Для представления параллельных процессов на диаграммах деятельности предусмотрены специальные инструменты: «Разветвление»  и «Соединение»  . На рисунке 8.5 показана диаграмма деятельности, на которой параллельно выполняются процессы, соответствующие третьему и четвертому состояниям действиям.

Другим средством ArgoUML для представления параллельных процессов является

инструмент «Дорожки» (swimlanes)  . Он позволяет моделировать работу предприятий (бизнес-процессы), предполагающую взаимодействие между отделами предприятия.

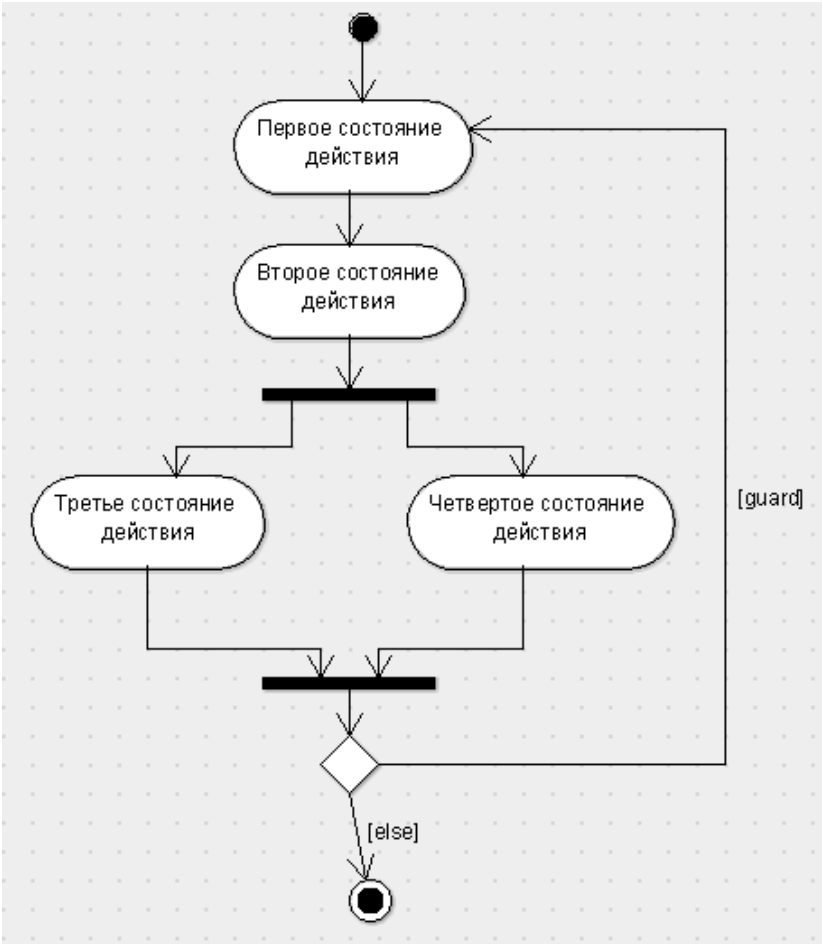

Рисунок 8.5 – Диаграмма деятельности с последовательным и параллельным выполнением операций

На рисунке 8.6 показана диаграмма деятельности, в которой использован инструмент «Дорожки» для моделирования работы двух отделов предприятия.

Инструмент  можно применять многократно для получения необходимого количества дорожек. Дорожки на панели редактирования размещаются рядом и имеют одинаковую длину.

Чтобы добавить надписи на диаграмму деятельности необходимо использовать щелчок правой кнопкой мыши и ниспадающее меню (рисунок 8.7).

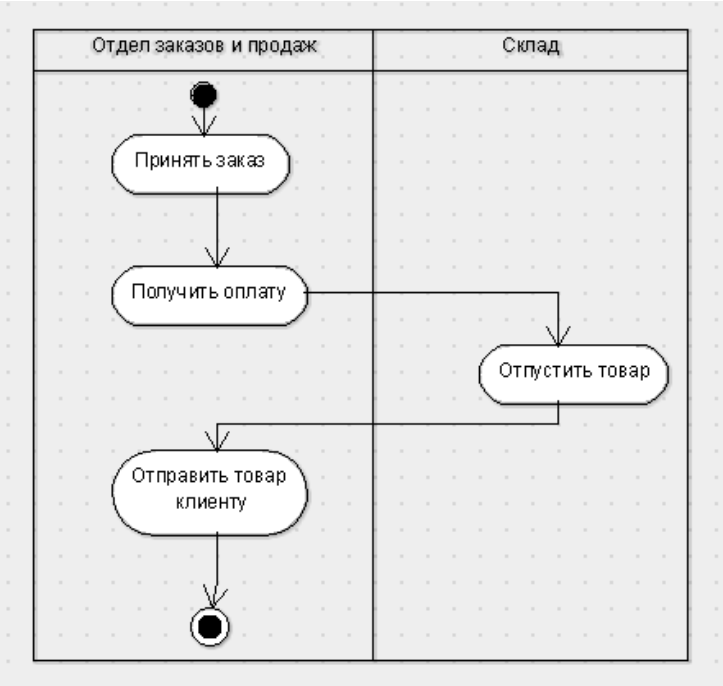

Рисунок 8.6 - Диаграмма деятельности с параллельными дорожками

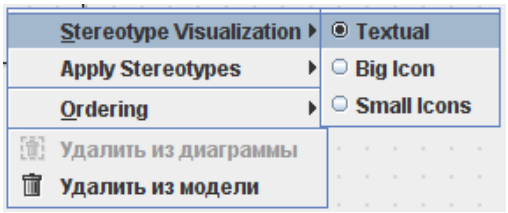

Рисунок 8.7 – Ниспадающее меню для добавления надписей

В общем случае действия на диаграмме деятельности выполняются над теми или иными объектами. Эти объекты либо инициируют выполнение действий, либо определяют некоторый результат этих действий. При этом действия специфицируют вызовы, которые передаются от одного объекта графа деятельности к другому. Объекты можно явно указать на диаграмме деятельности (рисунок 8.8).

Для графического представления объектов используется прямоугольник класса, с тем отличием, что имя объекта подчеркивается. На диаграмме деятельности с дорожками расположение объекта может иметь некоторый дополнительный смысл. А именно, если объект расположен на границе двух дорожек, то это может означать, что переход к следующему состоянию действия в соседней дорожке ассоциирован с готовностью некоторого документа (объект в некотором состоянии). Если же объект целиком расположен внутри дорожки, то и состояние этого объекта целиком определяется действиями данной дорожки.

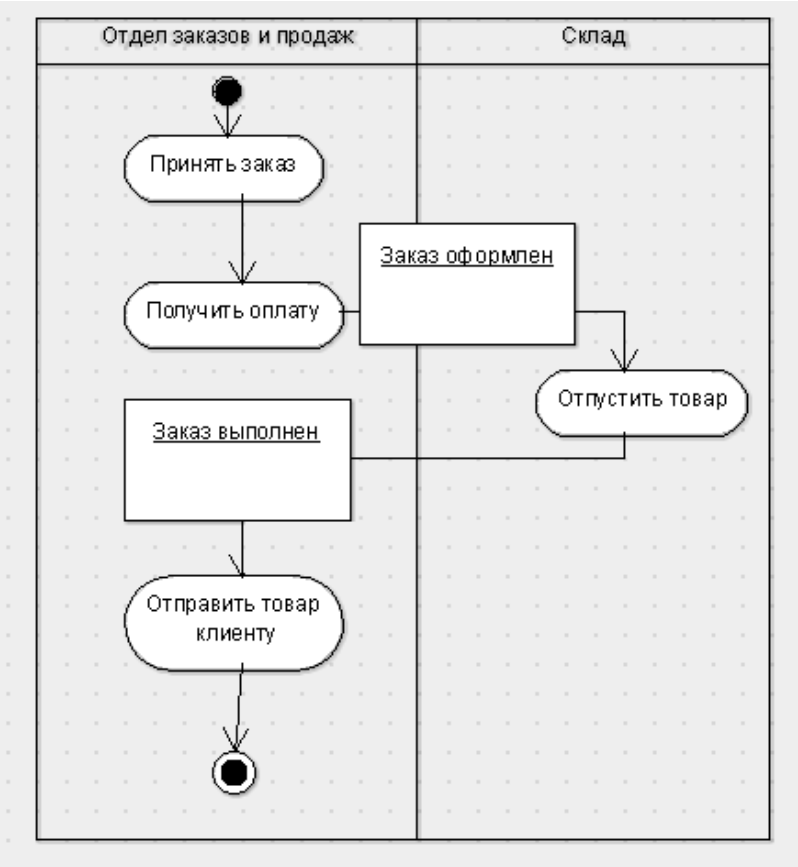

Рисунок 8.8 – Диаграмма деятельности с параллельными дорожками и объектом

Задание 1. Необходимо изучить предметную область, соответствующую вашему варианту задания, которое представлено в таблице 1.1. Выделите действия, характерные для предметной области, проанализируйте возможные переходы между состояниями. Проанализируйте возможность реализации последовательных и параллельных действий. Постройте диаграмму деятельности.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать самостоятельно построенную диаграмму деятельности.

Вопросы для защиты работы

1. Какие команды реализованы в панели инструментов диаграммы деятельности ArgoUML?
2. Какими средствами ArgoUML можно представить последовательные и параллельные операции?
3. Охарактеризуйте инструмент «Дорожки» ArgoUML?

Лабораторная работа № 9

ДИАГРАММЫ РАЗВЕРТЫВАНИЯ

Цель и содержание работы: изучить порядок построения диаграммы развертывания с помощью ArgoUML.

Формируемые компетенции: ПК-2, ПК-4, ПК-8

Теоретическое обоснование

Диаграмма развертывания (Deployment diagram) в UML моделирует *физическое* развертывание артефактов на узлах. Например, чтобы описать web-сайт диаграмма развертывания должна показывать, какие аппаратные компоненты («узлы») существуют (например, web-сервер, сервер базы данных, сервер приложения), какие программные компоненты («артефакты») работают на каждом узле (например, web-приложение, база данных), и как различные части этого комплекса соединяются друг с другом.

Узлы представляются как прямоугольные параллелепипеды с артефактами, расположенными в них, изображенными в виде прямоугольников. Узлы могут иметь подузлы, которые представляются как вложенные прямоугольные параллелепипеды. Один узел диаграммы развертывания может концептуально представлять множество физических узлов, таких как кластер серверов баз данных.

Существует два типа узлов:

1. Узел устройства.
2. Узел среды выполнения.

Узлы устройств — это физические вычислительные ресурсы со своей памятью и сервисами для выполнения программного обеспечения, такие как обычные ПК, мобильные телефоны. Узел среды выполнения — это программный вычислительный ресурс, который работает внутри внешнего узла и который предоставляет собой сервис, выполняющий другие исполняемые программные элементы.

Панель инструментов для построения диаграммы развертывания представлена на рисунке 9.1. Описание инструментов представлено в таблице 9.1.

Рисунок 9.1 – Инструменты для построения диаграммы развертывания

Диаграмма развертывания показывает наличие физических соединений - маршрутов передачи информации между аппаратными устройствами, задействованными в реализации

системы. Диаграмма развертывания предназначена для визуализации элементов и компонентов программы, существующих лишь на этапе ее исполнения (runtime).

Таблица 9.1 – Описание инструментов диаграммы развертывания

Инструмент	Описание
	Узел (узел-тип)
	Узел - экземпляр
	Компонент
	Компонент-экземпляр
	Обобщение (от ребенка к родителю)
	Реализация (от класса к интерфейсу)
	Зависимость (от зависимого элемента)
	Ассоциация (шесть видов ассоциаций представлены на рисунке 10.3)
	Объект
	Связь

Узел (node) представляет собой некоторый физически существующий элемент системы, обладающий некоторым вычислительным ресурсом. В языке UML понятие узла может включать в себя не только вычислительные устройства (процессоры), но и другие механические или электронные устройства, такие как датчики, принтеры, модемы, цифровые камеры, сканеры и манипуляторы. Узлы на диаграмме развертывания могут представляться как в качестве типов (рисунок 9.2, а), так и в качестве экземпляров (рисунок 9.2, б).

а) б)

Рисунок 9.2 – Изображение узлов на диаграмме развертывания

Внутри узлов на диаграмме развертывания могут быть размещены исполняемые компоненты. На рисунке 9.3 показана диаграмма развертывания, на которой внутри узлов

расположены исполняемые компоненты. Так как в ArgoUML нет инструмента для изображения интерфейса, то на диаграмме развертывания показан объект – экземпляр интерфейса.

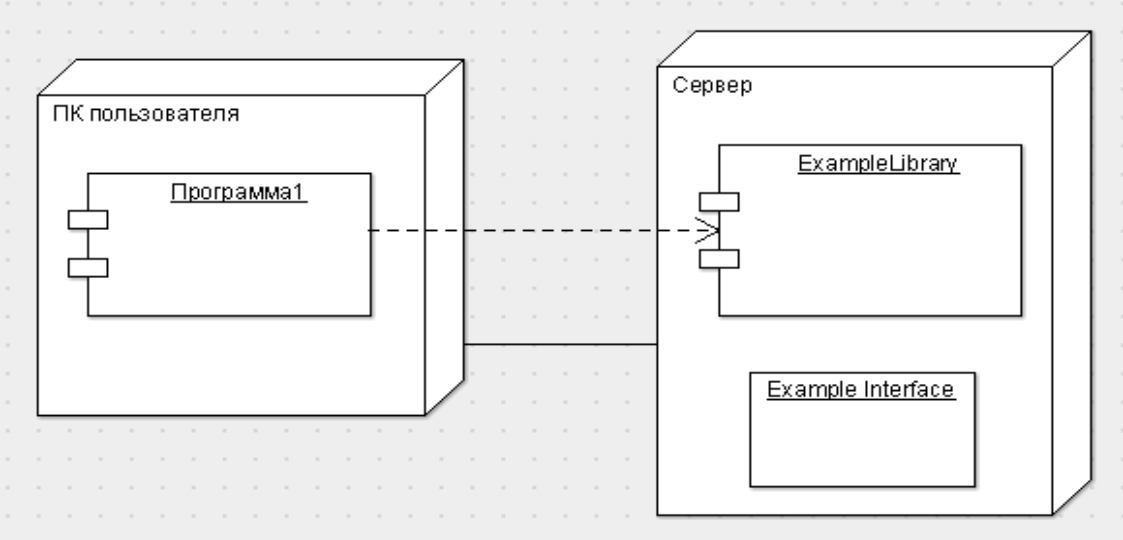

Рисунок 9.3 – Изображение исполняемых компонентов внутри узлов на диаграмме развертывания

В качестве отношений между узлами на диаграммах развертывания можно представить физические соединения между и зависимости между узлами и компонентами. На рисунке 9.4 показана диаграмма развертывания для компьютерной сети. Комментарий добавлен с помощью инструмента  , связь комментария с узлом добавлена с помощью инструмента  .

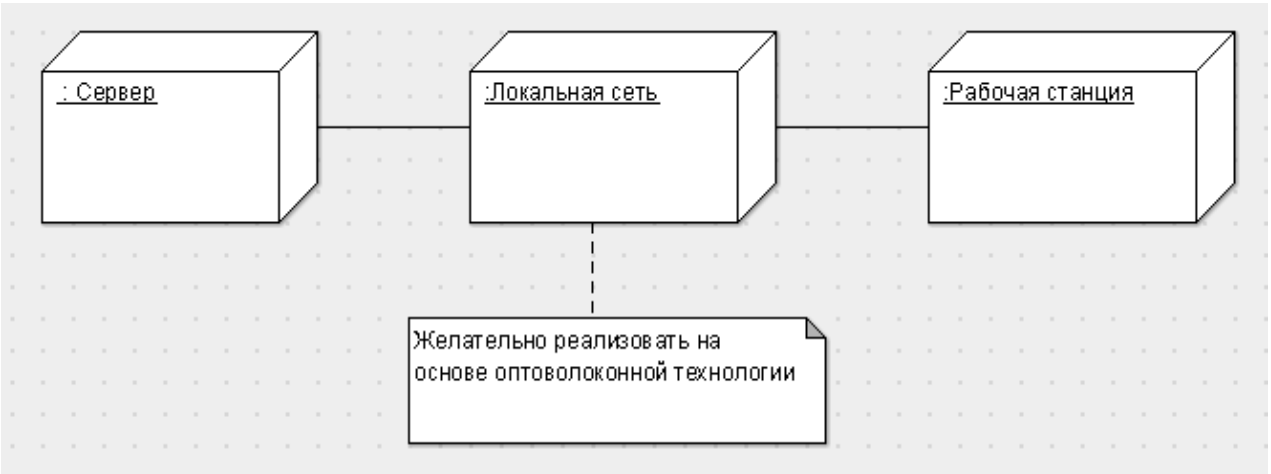

Рисунок 9.4 – Изображение локальной сети на диаграмме развертывания

Если на узле развернуто большое количество компонентов, то их можно показать не внутри узла, а в виде отношения зависимости (рисунок 9.5).

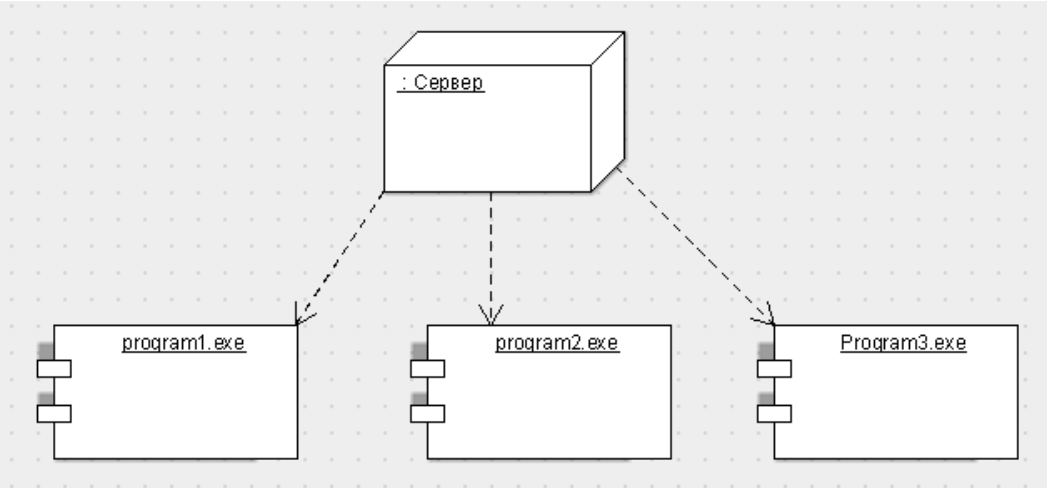

Рисунок 9.5 – Диаграмма развертывания с использованием отношения зависимости

Задание 1. Необходимо изучить предметную область, соответствующую вашему варианту задания, которое представлено в таблице 1.1. Выделите характерные для предметной области компоненты программы, существующие лишь на этапе ее исполнения, проанализируйте, какие аппаратные или программные компоненты можно представить как узлы. Проанализируйте возможность реализации программных или аппаратных решений для моделирования предметной области. Постройте диаграмму развертывания.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать самостоятельно построенную диаграмму развертывания.

Вопросы для защиты работы

1. Какие команды реализованы в панели инструментов диаграммы развертывания ArgoUML?
2. Какими средствами ArgoUML можно представить сложные зависимости между узлами и компонентами?
3. Охарактеризуйте отличие между узлом-типом и узлом-экземпляром в ArgoUML.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Батоврин В.К. Толковый словарь по системной и программной инженерии: учеб. Пособие для ВУЗов. М.: ДМК Пресс, 2012. 280с.

2. Иванова Г.С. Технология программирования: учебник/ М.:КНОРУС, 2011. – 336 с.

3. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на С++: пер. с англ. М.:Бином, СПб.: Невский диалект, 1998.

4. Камаев В.А., Костерин В.В. Технологии программирования: Учебник. М.: Высш.шк., 2005. – 359 с.

5. Смирнова Г.Н., Сорокин А.А., Тельнов Ю.Ф. Проектирование экономических информационных систем/Под ред. Ю.Ф. Тельнова .– М.:Финансы и статистика,2001. –512 с.

6. Липаев В.В. Методы обеспечения качества крупномасштабных программных систем. – М.: СИНТЕГ.– 2003.–510 с.

7. Андон Ф.И., Суслов В.Ю., Коваль Г.И., Коротун Т.М. Основы качества программных систем.–Киев, Академпериодика.– 2002.–502с.

8. Липаев В.В. Надежность программного обеспечения АСУ.–М.: Сов.радио, 1977.–400с.

9. Майерс Г. Надежность программного обеспечения .– М.: Мир, 1980.–360с.

10. http://argouml.tigris.org

11. http://argouml.tigris.org/servlets/ProjectMailingListList

12. Методология функционального моделирования IDEF0/ РД IDEF 0 – 2000/ Издание официальное/ ГОССТАНДАРТ РОССИИ, 2000г.

13. http://www.insapov.ru/idef0-standard-description.html

14. IEEE-CS/ACM Software Engineering Ethics and Professional Practices. http://www.computer.org/tab/seprof/code.htm#Public

3.Леоненков А.В.Самоучитель UML. http://khpi-iip.mipk.kharkiv.edu/library/case/leon